COPPE/UFRJ

OTIMIZAÇÃO COM ENXAME DE PARTÍCULAS E BUSCA BASEADA EM CLASSES APLICADAS AO PROBLEMA DA RECARGA DE UM REATOR NUCLEAR

Anderson Alvarenga de Moura Meneses

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Nuclear, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Nuclear.

Orientador: Roberto Schirru

Rio de Janeiro

Abril/2010

OTIMIZAÇÃO COM ENXAME DE PARTÍCULAS E BUSCA BASEADA EM CLASSES APLICADAS AO PROBLEMA DA RECARGA DE UM REATOR NUCLEAR

Anderson Alvarenga de Moura Meneses

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA NUCLEAR.

Examinada por:

_____
Prof. Roberto Schirru, D.Sc.

_____
Prof. Eduardo Gomes Dutra do Carmo, D.Sc.

_____
Prof. José Antonio Carlos Canedo Medeiros, D.Sc.

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

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

RIO DE JANEIRO, RJ - BRASIL
ABRIL DE 2010

Meneses, Anderson Alvarenga de Moura

Otimização com enxame de partículas e busca baseada em classes aplicadas ao problema da recarga de um reator nuclear / Anderson Alvarenga de Moura Meneses. – Rio de Janeiro: UFRJ/COPPE, 2010.

XI, 111 p.: il.; 29,7 cm.

Orientador: Roberto Schirru

Tese (doutorado) – UFRJ/ COPPE/ Programa de Engenharia Nuclear, 2010.

Referências Bibliográficas: p. 97-105.

1. Recarga do Reator Nuclear. 2. Otimização. 3. Inteligência Artificial. I. Schirru, Roberto. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Nuclear. III. Título.

Agradecimentos

Antes de tudo, agradeço a Deus. Agradeço à minha esposa e companheira Heloísa Meneses. À minha mãe e professora Arynéa Alvarenga Meneses e ao meu pai Antenor de Moura Meneses (*in memoriam*). A toda minha família.

Ao meu orientador Prof. Roberto Schirru e a Luca Maria Gambardella, pelo apoio e pela excelente oportunidade de convívio e aprendizado. Eu me considero privilegiado por ter estudado e trabalhado com estes dois grandes pesquisadores e Mestres.

A todos os funcionários, alunos e amigos que conheci no Laboratório de Monitoração de Processos (LMP), em especial a Marcelo, Alan, Ioná Maghali, Andressa, Kelling, Rafael, Simone, Serginho, Vinícius, Zé Luiz e César. A todos os professores do Programa de Engenharia Nuclear, em especial a Carlos Canedo Medeiros, Fernando Carvalho da Silva, Aquilino Senra Martinez e Eduardo Gomes Dutra. A todos os funcionários do Programa de Engenharia Nuclear, em especial à Tânia e à Jô. Aos professores Cláudio Márcio do Nascimento Abreu Pereira, Regina Cely Barroso e Luís Fernando de Oliveira.

Aos amigos que conheci no Instituto Dalle Molle de Estudos sobre Inteligência Artificial (IDSIA), em especial a Cássio Campos, Paola Rancoita, Alessandro Giusti, Tom Schaul, Nikolaus Mutsanas, Giuseppe Cuccu e Denis Mauá. Aos amigos físicos e professores Augusto Cézar Menezes, Christiano Jorge Gomes Pinheiro e Fabrício José Teixeira Mariano.

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

OTIMIZAÇÃO COM ENXAME DE PARTÍCULAS E BUSCA BASEADA EM CLASSES APLICADAS AO PROBLEMA DA RECARGA DE UM REATOR NUCLEAR

Anderson Alvarenga de Moura Meneses

Abril/2010

Orientador: Roberto Schirru

Programa: Engenharia Nuclear

O problema da recarga de um reator nuclear, ou o problema da Otimização do Gerenciamento de Combustível Intra-Núcleo (OGCIN), é um problema clássico em Engenharia Nuclear, estudado por mais de 40 anos e proeminente devido ao seu alto grau de complexidade. Nesta Tese apresentamos a aplicação da Meta-Heurística de Otimização (MHO) de Otimização com Enxame de Partículas à OGCIN. Também é apresentada a Heurística de Aceitação de Vizinhança Reativa, para uso com MHOs. Finalmente mostramos o desenvolvimento da MHO Busca Baseada em Classes, implementada para a OGCIN. Apresentamos também os resultados destas técnicas, cujos resultados têm qualidade superior a resultados obtidos por outras MHOs consideradas estado-de-arte para este problema.

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

PARTICLE SWARM OPTIMIZATION AND CLASS-BASED SEARCH APPLIED TO THE NUCLEAR REACTOR RELOAD PROBLEM

Anderson Alvarenga de Moura Meneses

April/2010

Advisor: Roberto Schirru

Department: Nuclear Engineering

The nuclear reactor reload problem or the In-Core Fuel Management Optimization (ICFMO) is a classical problem in Nuclear Engineering, studied for more than 40 years and prominent due to its high degree of complexity. In this thesis we present the application of the Optimization Metaheuristic (OMH) of Particle Swarm Optimization to the ICFMO. The Reactive Neighbourhood Acceptance Heuristic is also presented, to be used with OMHs. Finally we demonstrate the development of the Class Based Search OMH, implemented for the ICFMO. We also present the results of these techniques, whose results have superior quality in relation the ones obtained by other OMHs considered the state-of-art for this problem.

Índice

Lista de Acrônimos e Abreviações

ACO - Ant Colony Optimization

ANC - Advanced Nodal Code

AP - Assignment Problem

BBC - Busca Baseada em Classes

BE - Binary Exchange

BOC - Beginning of Cycle

DE - Differential Evolution

DEPP - Dia Efetivo a Plena Potência

EC - Elemento Combustível

GA - Genetic Algorithm

HAVR - Heurística de Aceitação de Vizinhança Reativa

IA - Inteligência Artificial

MHO - Meta-Heurística de Otimização

NPP - Nuclear Power Plant

OGCIN - Otimização do Gerenciamento de Combustível Intra-Núcleo

PBIL - Population-Based Incremental Learning

PC - Padrão de Carregamento

PM - Programação Matemática

PSO - Particle Swarm Optimization

PSORK - Particle Swarm Optimization with Random Keys

PWR - Pressurized Water Reactor

RNA - Rede Neural Artificial

RS - Random Search

SA - Simulated Annealing

SBC - Sistema Baseado em Conhecimento

TAP - Task Assignment Problem

TS - Tabu Search

TSP - Traveling Salesman Problem

VQ - Veneno Queimável

Capítulo 1

Introdução

O problema de Engenharia Nuclear de Otimização do Gerenciamento de Combustível Intra-Núcleo (OGCIN), ou otimização do projeto de Padrões de Carregamento (PCs), ou ainda problema da recarga de um reator nuclear, são denominações para o problema de otimização associado à operação de recarga de combustível em um reator nuclear, que é um problema estudado por mais de quatro décadas. Por exemplo, na operação de recarga de um Reator a Água Pressurizada (Pressurized Water Reactor, PWR), ao final de um ciclo de operação, aproximadamente um terço dos Elementos Combustíveis (ECs) são substituídos. Isto se deve a uma série de motivos, como o aumento da concentração de produtos de fissão nuclear, resultantes das reações, conforme o passar do tempo, influenciando diretamente a distribuição de potência no núcleo. Os ECs novos, juntamente com os ECs remanescentes, devem formar um novo PC, destinado ao novo ciclo de operação da planta. Assim, a fim de otimizar a queima de um novo PC é necessária, então, a alocação dos ECs novos, bem como a reorganização das posições dos antigos, de acordo com restrições de segurança, para prover então o funcionamento com o máximo aproveitamento do combustível nuclear.

Desta forma, o objetivo do OGCIN é determinar PCs para produção a plena potência dentro de margens de segurança adequadas [1]. Em outras palavras, a busca consiste em determinar PCs que maximizem a vida útil do combustível nuclear, permitindo o aumento do ciclo de queima do combustível, trazendo ganhos operacionais e econômicos relacionados ao funcionamento da usina, sujeito a restrições de segurança e regulatórias [2].

De acordo com STEVENS et al. [3], as principais características da OGCIN são não-linearidade, multi-modalidade, regiões desconexas de soluções possíveis, alta dimensionalidade e riscos de aproximação. A otimização de um reator nuclear também envolve múltiplos objetivos relacionados a economia, segurança e cálculos de Física de Reatores e Termo-Hidráulica.

A OGCIN representa um problema combinatório NP-Difícil. Nesse problema de otimização, para *n* ECs, o número total de PCs seria *n!*, sem considerar para o problema a colocação de Venenos Queimável (VQ) ou a orientação dos FAs, como o problema é abordado em [4], o que de fato aumenta a complexidade do problema. Por exemplo para um típico PWR como o da Usina Nuclear de Angra 1, no Estado do Rio de Janeiro, existem aproximadamente $8{,}09 \times 10^{200}$ PCs diferentes, já que o reator desta planta possui 121 ECs. Evidentemente, nem todos estes PCs representam soluções válidas ou possíveis para o problema, seja em termos de segurança ou por questões operacionais, o que faz com que no espaço de busca, as regiões que possuem soluções válidas sejam desconexas [2]. Além disso, para que haja uma distribuição de potência simétrica no núcleo do reator tendo em vista a máxima uniformidade na curva de densidade de potência, utilizam-se modelagens que representam 1/4 ou 1/8 do seu formato original, como é mostrado no esquema que representa a vista superior de um núcleo de reator a água pressurizada, na Figura 1.1, o que acaba por reduzir o espaço de busca. No entanto, ainda assim o número de combinações é muito grande: não é possível avaliar todas as configurações possíveis de PCs, uma vez que o tempo computacional tornaria inviável este processo. Em suma, a complexidade do problema, o grande número de soluções possíveis e o método de avaliação de soluções candidatas fazem este problema proeminente em Engenharia Nuclear.

Outra importante característica da OGCIN é que, além da complexidade própria do problema relativa ao seu caráter combinatório, a avaliação de soluções candidatas é também complexa. Neste sentido, são utilizados códigos de Física de Reatores com a implementação da solução numérica das equações dos modelos de Transporte ou Difusão de nêutrons e cálculos relativos à Termo-Hidráulica. Para achar então o valor da função objetivo são efetuados cálculos numéricos segundo as modelagens existentes para solucionar os sistemas de equações diferenciais de transporte ou difusão de nêutrons, a fim de determinar, a partir daí, a distribuição de potência, a reatividade, a

queima (*burn-up*) e os demais importantes resultados para análises e previsão do funcionamento de um reator nuclear, o que demanda considerável tempo computacional. Assim, vale ressaltar que o tempo necessário para avaliação de uma solução candidata no atual paradigma da computação sequencial de tais códigos é ainda proibitivo, o que torna o problema ainda mais atraente no sentido do desafio de desenvolver novos algoritmos para encontrar soluções próximas a ótimas com o menor número possível de avaliações.

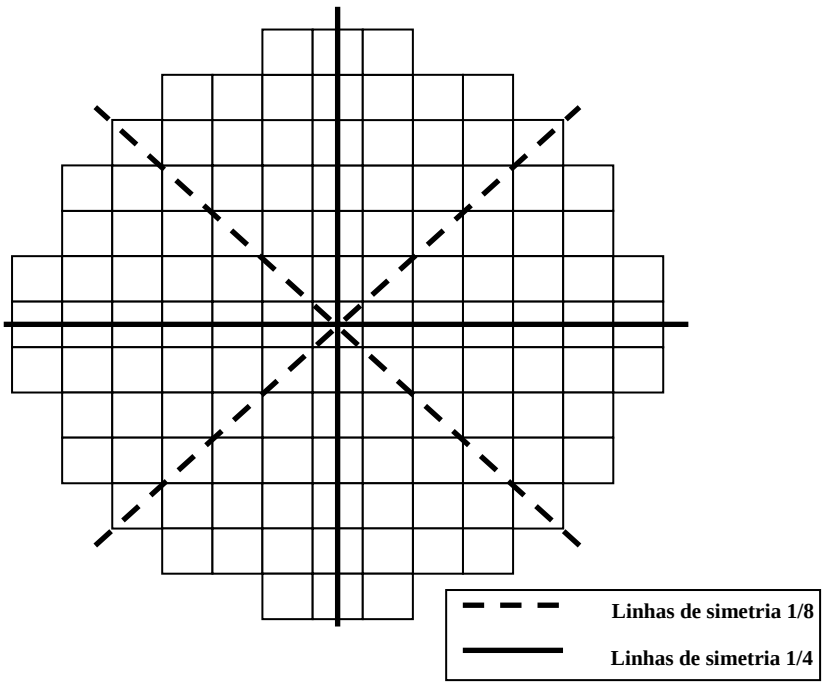


Figura 1.1 – Representação da vista superior do núcleo de um reator com 121 ECs, como o de Angra 1, com suas linhas de simetria.

Diversas técnicas para a OGCIN têm sido aplicadas ao longo dos anos, em alternativa a abordagens clássicas não adequadas a este problema de natureza combinatória. Dois exemplos seminais de abordagens utilizando Programação Matemática (PM) que podem ser citados são Programação Dinâmica, em 1965 [5] e Programação Linear e Quadrática, em 1968 [6]. O desenvolvimento de técnicas de Inteligência Artificial (IA) nas últimas décadas, em especial as chamadas Meta-Heurísticas de Otimização (MHO) [7], e sua aplicação à OGCIN tem levado a melhorias significativas nos resultados obtidos na otimização deste problema, com

destaque para as técnicas de Arrefecimento Simulado (*Simulated Annealing,* SA) [3, 8, 9], Algoritmo Genético (*Genetic Algorithm*, GA) [4, 10, 11, 12], Busca Tabu (*Tabu Search*, TS) [13], Aprendizado Incremental Baseado em Populações (*Population-Based Incremental Learning*, PBIL) [14, 15, 16, 17], e Otimização com Colônias de Formigas (*Ant Colony Optimization*, ACO) [16, 18, 19, 20].

Assim como as MHOs supracitadas, a Otimização com Enxame de Partículas (*Particle Swarm Optimization*, PSO) [21], inicialmente apresentada no ano de 1995 [22], também se mostrou eficiente em diversos problemas contínuos e em diversos ramos de Engenharia. Até 2009, quando surge o 1º. trabalho do PSO à OGCIN [23], exemplos de progressos nas pesquisas sobre PSO realizados em Engenharia Nuclear compreendiam sua aplicação ao projeto de núcleos de reatores nucleares [24] e a identificação de transientes durante a operação de uma usina nuclear [25]. Desenvolvimentos mais recentes englobam o uso do PSO para ajuste de um sistema de inferência neuro-fuzzy para monitoração de sensores [26], bem como modelos de PSO em computação paralela para a solução de problemas nucleares como o projeto de núcleo de reatores nucleares e da OGCIN [27]. O PSO modela uma busca colaborativa, levando em conta aspectos sociais da inteligência. Cada enxame é composto por várias partículas e, durante a busca, cada uma delas é guiada por sua própria experiência, bem como pela experiência do grupo obtida durante a busca, de acordo com o balanceamento entre cognição individual e aprendizado social. Parte desta tese será dedicada a relatar o desenvolvimento da aplicação da técnica PSO ao OGCIN, bem como os resultados obtidos. O modelo de PSO aplicado à OGCIN foi adaptado com o uso de chaves aleatórias (*Random Keys*, RK) [28]. Assim, foi possível mapear o espaço de busca em um espaço de soluções combinatórias que são de fato possíveis para a OGCIN.

Conforme mencionado anteriormente, um dos desafios para o desenvolvimento e implementação de algoritmos para a OGCIN é conseguir realizar a otimização no menor número possível de iterações. Desta forma, as MHO também podem ser utilizadas com Heurísticas para superar os reveses do tempo computacional proibitivo. Heurísticas são critérios ou princípios para decidir quais dentre diversos cursos alternativos são mais efetivos com respeito a algum objetivo. A segunda etapa do trabalho, seguinte ao desenvolvimento e implementação do PSO para o OGCIN, foi o desenvolvimento de

uma nova abordagem para o uso de heurísticas relacionais a serem utilizadas com MHO [29], a que chamamos Heurística de Aceitação de Vizinhança Reativa (HAVR), que foi implementada com a Busca Aleatória (Random Search, RS) e PSO, apresentando melhoras nos resultados tanto quantitativamente, em termos do ciclo de operação quanto em redução do tempo computacional.

Na terceira etapa, investigando ainda a possibilidade de ser realizada uma busca aproximada baseada nas principais características dos ECs, que foi particularmente bem sucedida para o uso da HAVR com MHOs como demonstrado em [29], desenvolvemos finalmente a MHO Busca Baseada em Classes (BBC), que leva em consideração as características ligadas a reatividade e posição relativa no núcleo, sendo mais uma técnica para investigação da OGCIN com resultados melhores que algumas técnicas que representam o estado-de-arte do problema.

Nas subseções 1.1 e 1.2 citaremos nossos trabalhos anteriores àqueles aqui apresentados, bem como explicitaremos os objetivos, a relevância e a inovação dos trabalhos desenvolvidos. Posteriormente, na subseção 1.3 apresentamos uma visão geral dos demais capítulos desta tese.

1.1 O Trabalho Desenvolvido Anteriormente

Antes do trabalho de desenvolvimento e aplicação do PSO à OGCIN citado anteriormente [23], trabalhos preliminares foram publicados relatando a validação de código, solução de instâncias do Problema do Caixeiro Viajante, aplicação do método com o código de Física de Reatores e Termo-Hidráulica Advanced Nodal Code (ANC), estudo de variante do PSO e estudo de confinamentos [30-35] (ver anexos I a VII desta tese).

Em [30], [31] e [32], aplicamos o PSO a problemas combinatórios que são referências no estudo da OGCIN, que são os problemas de caixeiro viajante (*Traveling*

Salesman Problems, TSPs) Oliver30 e ry48p (Rykel48). O número de soluções candidatas destes TSPs e sua não linearidade os fazem semelhantes em complexidade à OGCIN com simetria de ¼ de núcleo. Assim, os TSPs foram usados para uma análise prévia e validação da nova técnica, já que as avaliações da OGCIN são feitas com códigos de física de reatores com considerável custo computacional.

Em [33], para comparação prévia dos resultados do PSO com os resultados obtidos por um especialista em recarga fizemos o acoplamento ao código de Física de Reatores ANC, assim como feito em trabalho anterior com o GA [36]. Também foram realizados diversos testes com rotinas em dll (*dynamic link libraries;* bibliotecas de vínculo dinâmico) contendo as funções de geração de números aleatórios do Fortran e da linguagem C. A obtenção de resultados qualitativamente similares em ambas as linguagens de programação e com diferentes rotinas de geração de números aleatórios também demonstraram a robustez do método independente da plataforma de execução. Pôde ser observado que o PSO atingiu resultados comparáveis ao AG e ao especialista. Assim, os resultados reforçaram a possibilidade de a metaheurística Particle Swarm Optimization (PSORK), a ser discutida oportunamente, se tornar outra importante metaheurística de otimização.

Em [34], implementamos e testamos o algoritmo de *Guaranteed Convergence* (Convergência Garantida) ao problema combinatório cuja prova de convergência é demonstrada para o PSO em problemas contínuos [37-39], apresentando resultados para os TSPs Oliver30 e ry48p. Já em [35], apresentamos um estudo sobre o confinamento do PSO para os TSPs supracitados bem como resultados da aplicação de confinamentos para a OGCIN.

De uma maneira geral, os estudos realizados anteriormente foram fundamentais para a validação de toda a metodologia e bem como para a sedimentação de conhecimento relativo à OGCIN. Também foi possível verificar variantes do PSO, ou seja, possíveis complementações que podem ter utilidade para melhorias no desempenho deste algoritmo na OGCIN. Os resumos de tais publicações se encontram nos anexos a esta Tese.

1.2. Objetivos, Relevância e Inovação desta Tese

Esta tese tem o objetivo principal de mostrar o desenvolvimento e os resultados obtidos com: (i) um modelo de PSO específico para a OGCIN; (ii) uma nova abordagem para o uso de heurísticas na OGCIN; e (iii) uma nova MHO a ser aplicada à OGCIN, a BBC, citada anteriormente.

A relevância do trabalho desenvolvido reside no fato de que o mesmo apresenta novos e eficientes métodos de solução para a OGCIN, que representa um problema clássico e importante em Engenharia Nuclear. Como veremos adiante, os resultados alcançados mostram não somente a viabilidade de uso das técnicas desenvolvidas, mas também sua eficiência e eficácia em relação a resultados obtidos por outros métodos. Deve ser lembrado, ainda, que melhorias nos métodos de busca aplicados à OGCIN representa em termos práticos grande vantagem econômica, como será detalhado adiante. De fato, os resultados do trabalho desenvolvido apontam para o aprimoramento da qualidade na OGCIN.

Esta tese também possui conteúdo inovador no senso que, primeiramente, mostra os resultados da primeira aplicação do PSO à OGCIN, acrescido de um estudo detalhado do balanceamento dos parâmetros do algoritmo para este problema. Em segundo lugar, traz uma nova abordagem para o estudo de Heurísticas a serem usadas em conjunto com MHOs, reduzindo o custo computacional e em alguns casos melhorando resultados qualitativa e quantitativamente. Finalmente, com o desenvolvimento da BBC, uma nova MHO desenvolvida especificamente para o problema da OGCIN, foram atingidos resultados melhores que outras MHOs usadas para a OGCIN do ciclo 7 de Angra 1.

1.3 Visão Geral dos Capítulos da Tese

No capítulo 2, apresentamos formalmente a OGCIN e suas características, bem como discutimos sua formulação matemática e correlação com o TSP e o problema de atribuição (*Assignment Problem*, AP).

Discutimos no capítulo 3 as MHOs aplicadas à OGCIN, com destaque para o PSO e a Busca Baseada em Classes.

O capítulo 4 descreve os resultados da aplicação do PSO a duas instâncias do PCV (Oliver30 e Rykel48).

O capítulo 5 traz os resultados dos testes realizados por ocasião da aplicação do PSO à OGCIN, bem como sua análise e comentários.

No capítulo 6 apresentamos os resultados da HAVR implementada com RS e PSO. A exemplo do capítulo anterior, também analisamos e comentamos os resultados.

Os resultados da BBC aplicada à OGCIN são mostrados no capítulo 7 desta tese, juntamente com sua análise.

Finalmente, as conclusões desta tese são mostradas no capítulo 8, juntamente com as propostas de trabalhos futuros.

Capítulo 2

Otimização do Gerenciamento de Combustível Intra-Núcleo

Este capítulo é destinado à apresentação do problema da OGCIN, que é o problema combinatório referente à operação de recarga de ECs no núcleo do Reator de uma Usina Nuclear. Apresentamos seus aspectos teóricos, sua formulação matemática, bem como sua relação com os parâmetros de otimização fornecidos pelo simulador de operação da Usina de Angra 1, o RECNOD [12, 36].

2.1. Aspectos Teóricos

Após o período chamado ciclo de operação, não é possível manter a usina nuclear trabalhando com a potência nominal prevista nas especificações técnicas. Neste momento, é necessário o desligamento da usina para o processo de recarga, no qual uma parte do combustível nuclear é substituída por combustível novo. A OGCIN consiste em encontrar o PC de ECs, entre antigos e novos, que satisfaça as restrições do problema e forneça a maior duração de ciclo possível, dentro da estratégia de planejamento do ciclo de vida de combustível nuclear. A organização dos ECs em uma configuração de núcleo leva, portanto, a um problema de otimização combinatória, com uma função objetivo elaborada de acordo com métodos e critérios específicos da Engenharia Nuclear, envolvendo os conhecimentos científicos de Neutrônica e Termohidráulica.

Considerando a demanda planejada de potência, sujeito a restrições termohidráulicas relacionadas com a segurança, o objetivo básico da OGCIN é otimizar a utilização do combustível nuclear, em termos da duração do ciclo de operação, ou seja, de modo que a queima do combustível com a configuração encontrada forneça o maior número de dias de queima de acordo com as especificações técnicas da usina, medido em Dias Efetivos a Plena Potência (DEPP). O objetivo da OGCIN é determinar os PCs para produção de plena potência dentro de margens adequadas de segurança [1].

Através dos anos, o conhecimento de especialistas tem sido fundamental para a otimização. No entanto, técnicas de Inteligência Artificial, em especial as MHOs, que serão abordadas no capítulo 3, têm sido utilizadas com considerável sucesso como mencionado anteriormente na OGCIN, incluindo o GA [4, 10-12, 36], ACO [16, 18-20] e PBIL [14-17], pois trata-se de um problema com grande complexidade.

A complexidade da OGCIN é caracterizada, entre outros fatores, pelo grande número de soluções candidatas do problema, regiões desconexas de soluções possíveis, riscos relativos a aproximações [3]. Assim, a estratégia utilizada para avaliar e comparar as soluções candidatas na metaheurística PSORK, que será vista oportunamente, foi a maximização do ciclo do combustível nuclear considerando a Concentração de Boro fornecida por um código de Física de Reatores, neste caso, o RECNOD, dada uma configuração de ECs no núcleo de um Reator. A Concentração de Boro possui uma relação direta com o número de DEPPs do ciclo de uma Usina Nuclear. Como 4ppm de Boro correspondem a aproximadamente 1 DEPP [36], configurações de núcleo que maximizam a Concentração de Boro permitem ciclos maiores, ou seja, maiores concentrações indicam que a usina nuclear operará uma quantidade maior de dias com tais configurações. Por exemplo, 1 DEPP equivale a um retorno econômico da ordem de centenas de milhares de dólares norte-americanos na operação de uma usina nuclear PWR.

A OGCIN pode ser formulada de diversas maneiras. Por exemplo, pode ser formulada para uma única usina ou para uma comunidade de plantas [40]. O problema pode também ser formulado para um único ciclo de operação, quando é considerado somente um período de intervalo entre dois desligamentos (*shut-downs*) sucessivos, ou multi-ciclo, quando mais de um período de intervalo é considerado. Por exemplo, o

sistema SIMAN/X-IMAGE [3] foi projetado para apoiar otimizações de recargas multi-ciclo ou de ciclo único.

Uma outra abordagem é considerar além das posições dos ECs, sua orientação e a presença de VQ [4], ou uma abordagem baseada na OGCIN mas para otimizar somente o VQ a ser utilizado [41], ou ainda para buscar as melhores posições de ECs e VQ [42]. Não obstante, também é possível a busca pelo melhor PC possível, sem preocupação com VQ e orientação [12, 14-20, 36].

A OGCIN possui múltiplos objetivos com relação a economia, procedimentos operacionais e restrições regulatórias [2]. Assim, é possível buscar soluções para a OGCIN dentro de um arcabouço multi-critério com diversos (e possivelmente conflitantes) objetivos, buscando as melhores soluções que vão pertencer a uma superfície de *compromisso* (*trade-off*), a chamada superfície de Pareto. Também é possível agregar os objetivos em apenas uma função objetivo, como a função descrita em [43], ou a função de *fitness* descrita em [17]. Como nestes exemplos, uma única função objetivo, contendo ambos os critérios relativos à duração do ciclo e à segurança, é considerada no nosso trabalho.

O grupo de técnicas usadas na OGCIN engloba otimização manual, PM, MHOs e Sistemas Baseados em Conhecimento (SBC). De fato, essas abordagens levam a três categorias de ferramentas computadorizadas para a OGCIN: pacotes de projeto manual, sistemas especialistas e pacotes de otimização [44]. Como mencionado anteriormente, aplicações de métodos de PM foram feitas em 1965, com Programação Dinâmica [5], e em 1968, com Programação Linear e Quadrática [6]. SBCs também tem sido aplicados à OGCIN e um dos primeiros usos de regras lógicas para a formação de PCs pode ser vista em [40].

A OGCIN é um problema do *mundo-real* (*real-world problem*) com complexidade na busca de soluções próximas a ótimas bem como uma função de avaliação altamente complexa, consistindo em códigos neutrônicos baseados na solução numérica de métodos de Física de Reatores. Diversas tentativas de contornar o alto custo computacional das avaliações de soluções candidatas foram feitos, por exemplo o uso de Redes Neurais Artificiais (RNAs) como aproximadores universais para realizar a

avaliação dos PCs substituindo os códigos de Física de Reatores, com um custo menor na fase de otimização. Neste sentido, RNAs foram usadas com GAs para a OGCIN de um PWR [45] e reatores avançados resfriados a gás [46]. O projeto de um método de busca para a OGCIN deve levar em consideração que o tempo necessário para avaliar uma única solução candidata é proibitivo, direcionando esforços para um menor número de avaliações durante o processo de otimização.

Em suma, a OGCIN é um problema complexo de Engenharia Nuclear, cujos objetivos estão relacionados a economia, segurança e aspectos regulatórios. Sua complexidade se deve não somente a suas características combinatórias e não-polinomiais, mas também à complexidade da função de avaliação das soluções candidatas, necessitando de processos de otimização com um menor número de avaliações. Isto será particularmente importante no capítulo 3, quando da apresentação da HAVR e da BBC. A seção a seguir mostra a formulação matemática da OGCIN utilizada neste trabalho.

2.2 Formulação Matemática

Formalmente, para uma única planta e para a otimização de um único ciclo, sem considerar a orientação dos ECs ou o posicionamento de VQ no núcleo, e para a otimização do ciclo de operação, sujeito a restrições de segurança, a OGCIN pode ser formulada como uma variação do AP [47], com similaridades nas restrições de posicionamento, embora com diferenças óbvias na avaliação de soluções candidatas: dado um conjunto S de m ECs, um conjunto D de m posições, o problema da OGCIN é atribuir cada EC $i \in$ S para uma e somente uma posição $j \in$ D de tal maneira que cada posição é ocupada por algum EC. Dadas as variáveis de decisão

$$x_{ij} = \begin{cases} 1 & para\ \ i\ atribuído\ \ a\ \ j, \\ 0 & caso\ \ contrário, \end{cases} \tag{2.1}$$

e a função objetivo relacionada com o ciclo de operação escrita como

$$\text{maximizar (ou minimizar) } F_{relacionada\ ao\ ciclo}(\tilde{p}), \tag{2.2}$$

onde F é uma função relacionada ao ciclo de operação (uma função que poderia ser maximizada ou minimizada, dependendo da escolha da função), e $\tilde{p}$ sendo um vetor representando uma permutação de ECs, as restrições de posicionamento relacionadas com cada EC são estabelecidas de acordo com a atribução de cada EC a exatamente uma posição e que cada posição seja ocupada por algum EC. Tais restrições de posicionamento são dadas por

$$\sum_{j=1}^{m} x_{ij} = 1 \quad i = 1, 2, ..., m \qquad \text{e} \tag{2.3}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, 2, ..., m \qquad . \tag{2.4}$$

As restrições de segurança podem ser representadas pela função G relacionada à segurança

$$G_{relacionada\ a\ seguranca}(\tilde{p}) < G_0 \,. \tag{2.5}$$

Ambos os valores para *F* e *G* durante o processo de otimização são obtidos com códigos de Física de Reatores tendo o PC representado por $\tilde{p}$ como entrada.

MACHADO e SCHIRRU [18] aplicaram ACO [48] à OGCIN, modelando o problema com base no TSP. Este e outros trabalhos [12, 14, 15, 23] usam TSPs como problemas de referência para testes de MHOs antes de aplicá-las diretamente à OGCIN. Isso leva a uma rápida discussão sobre os modelos de OGCIN, o AP e o TSP. Por exemplo, vamos considerar a permutação $\tilde{p}_1 = [2\ 1\ 4\ 3]$, representando o posicionamento de 4 ECs em um reator abstrato, cada um dos quatro em uma posição do núcleo, com o EC 2 na primeira posição, o EC 1 na segunda posição etc. De acordo com a formulação da OGCIN com base no AP, as variáveis de decisão para a permutação podem representadas pela matriz

$$X = [x_{ij}] = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{2.6}$$

representando que o EC *i* é atribuído à posição *j*. Com base na formulaçãp do TSP em [48], que considera *i* e *j* como duas cidades sucessivas, e de acordo com [18], uma solução candidata para a OGCIN pode ser representada como uma seqüência de ECs sucessivos *i* and *j* em uma linha hipotética no núcleo. A matriz Y com as variáveis de decisão y_{ij} para o PC abstrato representado por $\tilde{p}_1$ é então

$$Y = [y_{ij}] = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.7}$$

Ou seja, a seqüência dada em $\tilde{p}_1$ implica que o EC 2 é seguido pelo EC 1, o EC 1 é seguido pelo EC 3 e assim por diante, sobre a linha hipotética dentro do núcleo.

Embora as formulações para a OGCIN baseadas no AP e no TSP resultem em diferentes matrizes de variáveis de decisão, ambas são equivalentes. De acordo com [49], o TSP é um caso especial da formulação de Koopsman-Beckmann e portanto "o TSP de n cidades é exatamente equivalente a AP n×n" com a restrição de que as permutações do TSP devem ser cíclicas. A vantagem de este ponto ficar claro é que qualquer técnica aplicada ao TSP ou ao AP pode ser equivalentemente adaptada para a OGCIN, como tem sido feito para MHOs como o GA, ACO e PSO, que serão discutidas no próximo capítulo.

2.3 Simulação do Ciclo 7 da Usina de Angra 1 com o Código de Física de Reatores RECNOD

A Usina Nuclear de Angra 1 é um PWR localizado no estado do Rio de Janeiro, no sudeste do Brasil, cujo núcleo é composto de 121 ECs. Uma representação de sua vista superior pode ser vista na Figura 2.1. Conforme mencionado, tal modelo para o núcleo de Angra 1 fornece aproximadamente 121! ($\approx 8{,}09\times10^{200}$) soluções candidatas. No entanto, já que a distribuição de potência precisa ser simétrica no núcleo do reator esta característca pode ser usada para reduzir a complexidade da OGCIN. Já que existem dois eixos principais dividindo o núcleo (eixos de simetria de ¼ de núcleo), podemos dividir o núcleo e otimizar as combinações de aproximadamente apenas ¼ de seus ECs, como pode ser visto na figura 2.2.

Assim, utilizando tal simetria, passamos a ter 37 ECs e, mantendo o elemento central fixo como no veremos adiante para o modelo utilizado, o número de combinações possíveis cai para aproximamente 36! ($\approx 3{,}7\times10^{41}$).

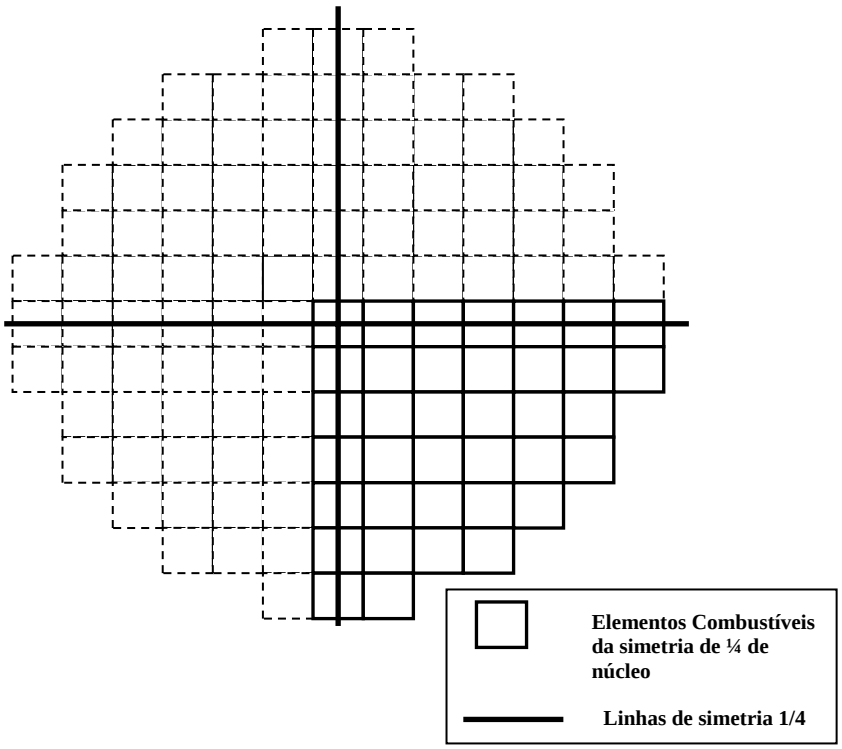


Figura 2.1 – Representação da vista superior do núcleo de um reator com 121 ECs, como o de Angra 1, com destaque para o esquema de simetria de ¼ de núcleo.

Utilizando os eixos secundários (eixos de 1/8 de núcleo) também mostrados na figura 1.1, passamos a ter a 21 ECs como mostra a figura 2.2. Desta forma o número combinações possíveis mantendo o elemento central fixo passa a ser 20! ($\approx 2{,}4\times10^{18}$), que ainda assim é um número muito alto. Os ECs que se situam sobre os eixos de simetria apenas podem ser permutados com ECs também dos eixos (tais ECs são chamados *quartetos* pois obviamente, para que se mantenha a simetria, o conjunto de quatro ECs simétricos entre si – que se encontram sobre os eixos de simetria – devem possuir as mesmas características como enriquecimento e tempo no núcleo). Os ECs que não se encontram sobre os eixos de simetria, chamados *octetos*, também só podem ser permutados com ECs nesta mesma condição, pois existem sempre mais sete ECs simétricos, que entrarão em posições simétricas no restante do núcleo para manter a distribuição de potência o mais uniforme possível. Tal restrição coloca o problema com $10!\times10!$ soluções possíveis, o que fornece aproximadamente $1{,}3\times10^{13}$ núcleos a serem avaliados. Chapot [36] descreve outros tipos de situações em que tal simetria é quebrada.

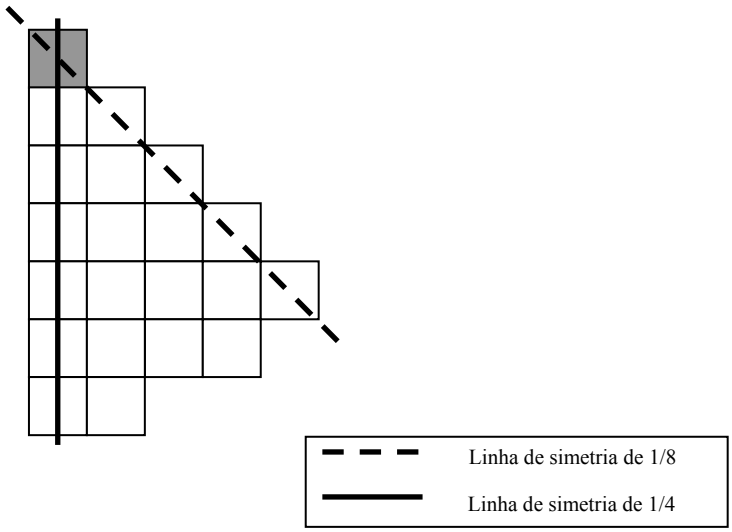


Figura 2.2 – Simetria de 1/8 de núcleo. Excetuando-se o elemento central, em cinza, todos os demais ECs são permutados.

O código de Física de Reatores RECNOD é o simulador para a usina de Angra 1 usado neste trabalho. O desenvolvimento e testes relacionados ao 7º. ciclo da referida usina encontram-se detalhados em [36]. RECNOD é um código nodal com base nos trabalhos descritos por LANGENBUCH et al. [62], LIU et al. [61] e MONTAGNINI et al. [64] e aplicado em diversos trabalhos de otimização [12, 14-20, 36].

Os parâmetros nucleares fornecidos pelo RECNOD são, entre outros, a Máxima Potência Média Relativa (P_{rm}) e a Concentração de Boro (C_B). Tal limitação do código RECNOD se deve ao exposto a seguir [20, 36]. Um código comercial de Física de Reatores como o ANC, utilizado para a OGCIN, possui os seguintes módulos: fluxo-potência-reatividade; queima de combustível; pesquisa de criticalidade com Boro solúvel; modelos de realimentação (com a correção da densidade do moderador e a realimentação Doppler, entre outros); e a reconstrução de densidade de potência pino a pino. No entanto, os módulos de realimentação Termo-Hidráulica e de reconstrução pino a pino não fazem parte do RECNOD.

Assim, o custo computacional de avaliação de uma solução pelo RECNOD é menor, sendo que, para efeitos de estudos de otimização, o uso da P_{rm} não viola as especificações técnicas da Usina de Angra 1 [36]. Para um Fator de Pico de Potência

Radial $F_{XY\ max}$ = *1,435* para Angra 1, os cálculos correspondem a P_{rm} = *1,395*. Assim qualquer PC cuja simulação resulte em $P_{rm} \leq 1{,}395$ estará dentro dos requisitos técnicos de segurança. Caso contrário, o PCs não pode ser aceito como solução (embora algumas de suas características, dependendo do caso, não possam ser aproveitadas no sentido do processo de busca das MHOs, como veremos adiante).

O valor de C_B fornecido pela avaliação do RECNOD é dada no Equilíbrio do Xenônio, outro aspecto que reduz o esforço computacional do processamento, sem prejudicar a validade para propósitos de otimização. CHAPOT [36] demonstrou que é possível extrapolar e predizer o ciclo de operação referente a um dado PC de maneira que 4ppm são equivalentes a 1 DEPP.

Assim, se considerarmos uma função *F* relacionada ao ciclo e *G* relacionada à segurança, de modo que

$$F_{relacionada\ ao\ ciclo}(\tilde{p}) = \frac{1}{C_B}, \tag{2.8}$$

$$G_{relacionada\ a\ segurança}(\tilde{p}) = P_{rm} \tag{2.9}$$

e

$$G_0 = 1{,}395 \tag{2.10}$$

então a OGCIN pode ser formulada como

$$\text{minimizar } \frac{1}{C_B} \tag{2.11}$$

$$\text{sujeito a } P_{rm} \leq 1{,}395 \quad . \tag{2.12}$$

Para as implementações computacionais, tanto a função relativa ao ciclo quanto a restrição relativa à segurança foram agregadas em uma única função de fitness (considerando que os valores de P_{rm} são sempre maiores que o inverso de C_B em nossos experimentos) então temos:

$$Fitness = \begin{cases} \dfrac{1}{C_B}, \; se \; P_{rm} \leq 1{,}395 \\ P_{rm}, \qquad caso \; contrário \end{cases} . \tag{2.13}$$

Em suma, para a OGCIN em nossos estudos de otimização utilizamos o simulador RECNOD, projetado para o ciclo 7 da usina nuclear de Angra 1 [20]. Além disto, é importante frisar que os parâmetros nucleares relevantes para o trabalho aqui relatado são C_B, relacionado ao comprimento do ciclo, e P_{rm}, relacionado a critérios de segurança, agregados em uma única função objetivo, mostrada na equação (2.13).

No próximo capítulo, serão analisadas as MHOs aplicadas à recarga de combustível. Serão vistas com destaque as MHOs PSO e BBC, aplicadas à OGCIN neste trabalho, bem como apresentaremos a HAVR, fundamental para o desenvolvimento da BBC.

Capítulo 3

Metaheurísticas aplicadas à Otimização do Gerenciamento de Combustível Intra-Nucleo

Neste capítulo apresentaremos uma visão geral das MHOs, mostrando a MHO de maior relevância para este trabalho: o PSO. Em seguida, apresentamos a nossa MHO de Busca Baseada em Classes, com a Heurística de Aceitação de Vizinhança Reativa.

3.1 Visão Geral

Junto com as importantes contribuições da PM e dos SBCs relatadas no capítulo anterior, MHOs têm sido aplicadas à OGCIN de maneira bem-sucedida, com relevantes resultados na solução deste problema. MHOs são métodos heurísticos genéricos, ou seja, de baixo acoplamento com especificidades de problemas e características como a memorização das soluções (ou de características de soluções) geradas durante o processo de busca [7].

A aplicação do SA [50] à OGCIN [8, 9] foi um dos primeiros importantes passos em direção ao uso de MHO no projeto de PCs. O SA é um algoritmo de busca local [3] no qual, a partir de um padrão (*incumbent*), uma série de perturbações gera um padrão candidato e o algoritmo decide preservar ou substituir o padrão original. Sua vantagem

sobre outros métodos é sua habilidade de escapar de mínimos locais em problemas multimodais [3].

O GA [51, 52] é uma MHO baseada nos princípios Neo-Darwinianos [36], levando em conta os principais fenômenos evolucionários como mutação, recombinação genética e seleção natural. Poon and Parks [4] apresentaram a primeira aplicação do GA à OGCIN, e os autores concluem que é atrativo usar em vez de SA por causa da característica de o primeiro ser implementado em paralelo, com uma grande performance do GA nas iterações iniciais, que contrasta com um progresso mais consistente do SA a longo prazo durante o processo de otimização. Não obstante, a implementação do GA para a OGCIN em [4] precisava de uma população inicial formada por soluções válidas, e esse procedimento foi também seguido em [53], com "um conjunto de PCs satisfazendo as restrições".

Outra MHO notável, a ACO [48] foi aplicada à OGCIN [18]. Situada no paradigma da Inteligência de Enxame, sua abordagem é diferente do SA, que modela um fenômeno físico, bem como diferente do GA, que modela aspectos evolucionários. Na ACO, os agentes, ou formigas artificiais, trabalham colaborativamente explorando um grafo, buscando soluções ótimas ou próximas a ótimas. Uma abordagem recente para a aplicação de ACO à OGCIN pode ser vista em [19] e [20], com as Redes Conectivas de Colônias de Formigas Artificiais.

Na próxima seção, abordaremos o PSO e sua variante para otimizar problemas combinatórios como a OGCIN.

3.2 Otimização com Enxame de Partículas

3.2.1. Aspectos Teóricos

O foco de parte de nosso trabalho de pesquisa é a utilização da MHO de Otimização por Enxame de Partículas (*Particle Swarm Optimization*, PSO) [21, 22] para a otimização da recarga de combustível em um reator nuclear. O PSO apresenta algumas características tais como a busca utilizando uma população de possíveis soluções com as regras de transição entre uma geração e outra envolvendo probabilidades. O PSO representa a metáfora do aprendizado social, que leva em conta que o indivíduo pode obter vantagem tanto de sua própria experiência quanto a partir do conhecimento adquirido por outro elemento de seu grupo, como será detalhado adiante. É um método que pode ser facilmente implementado, além de usar operadores aritméticos simples, e que mostrou ser eficiente em vários problemas de otimização.

Modelos que simulassem comportamentos sociais, como o de um bando de pássaros ou o de um cardume de peixes, estavam sendo estudados em torno de 1990 e serviram como fonte de inspiração para a elaboração da técnica. Entretanto, o interesse dos pesquisadores girava em torno da estética da coreografia e da descoberta de leis que descrevessem a dinâmica do movimento coletivo daqueles animais, que mesmo estando em grande quantidade, apresentam sincronismo em situações de mudança de direção, espalhamento, reagrupamento etc.

Na sociobiologia vêem-se discussões sobre a vantagem competitiva de se realizar aprendizado individual a partir do conhecimento de outros componentes do grupo [22]. Esta habilidade pode ser observada em muitas espécies, inclusive na humana, havendo, entretanto, uma principal diferença. Enquanto os demais animais realizam este aprendizado para funções instintivas como a defesa contra predadores ou a procura de alimentos e parceiros, os seres humanos também o aplicam nos campos da

abstração cognitiva. De qualquer forma, esse tipo de aprendizado é fundamental para vantagens competitivas e colaborativas evolucionárias.

Outro dado importante é que os modelos de simulação apresentados na época traziam processos locais como base para o movimento coletivo. Através deles, cada componente conseguiria ajustar suas condições de posição e velocidade em relação aos outros do grupo. Isto proporcionou solo fértil para a proposição do modelo de otimização utilizando um enxame de partículas.

No início da década de 90, foi proposto um novo modelo de movimento com base no aprendizado coletivo [22]. Este, porém, apesar de ser similar aos já apresentados, possuía um detalhe que o fazia diferente dos demais. Os pássaros desta modelagem eram atraídos para uma área de pouso. Isto seria possível quando fosse programada uma tendência maior para os pássaros pousarem em vez de permanecerem em vôo. Quando um deles passasse sobre uma determinada área, seria atraído, e os demais, com o aprendizado coletivo e o passar do tempo também se movimentariam com o intuito de aterrissar.

Ora, o que foi observado é que localizar um "poleiro" e mover-se em direção a ele são procedimentos semelhantes aos que diversas técnicas de otimização procuram fazer. Só que neste caso, a capacidade sócio-cognitiva é o fundamento, pois um pássaro que achou uma posição correspondente a uma solução considerada boa sob algum aspecto pode influenciar outros do bando, fazendo com que caminhem em sua direção, havendo sempre a possibilidade de passarem por alguma posição ainda melhor. Isto aumentaria as chances de serem descobertos melhores resultados conforme a evolução do algoritmo [21].

Algumas adaptações, logicamente, tiveram que ser efetuadas. Tratar cada indivíduo como pássaro ou peixe, de acordo com as analogias e metáforas utilizadas, seria um pouco inadequado. Isto poderia levar a supor que cada "agente" (este sim, um termo mais coerente) seria dotado de alguma espécie de capacidade cognitiva. Passou-se a utilizar então o termo *partícula* para caracterizar os membros de uma população que procura soluções, já que certas características como volume, massa e dimensões dos

agentes não são consideradas para a resolução de problemas, nem levadas em conta para o processo de otimização.

Uma partícula é, na verdade, a representação de um objeto que se move ao longo do espaço de busca impulsionado por comparações entre sua posição atual, sua melhor posição atingida até o momento e a posição obtida pela melhor partícula do enxame. Estes termos, devidamente equacionados é que vão proporcionar o cálculo de suas novas posições e velocidades a cada iteração, ou seja, vão direcionar o movimento das partículas. Cada uma das partículas realiza um balanço entre suas capacidades de exploração e de tirar proveito da exploração realizada pelas outras. A exploração é a capacidade de procurar uma boa solução individualmente. Tirar vantagem do sucesso obtido por outro indivíduo é que estaria relacionado ao comportamento de aprendizado social ou coletivo citado anteriormente.

O balanceamento destas características pode fazer grande diferença em um processo de otimização. Pouca capacidade de exploração pode significar convergência prematura para um resultado não necessariamente ótimo global. Pouca capacidade de obter vantagem do sucesso alheio pode fazer com que o enxame perca a habilidade de encontrar uma boa solução, e neste caso, não convergir para um resultado. Ou seja: o sucesso da busca realizada pelo enxame está condicionado ao balanço entre individualidade e coletividade, justamente os traços que compõem o comportamento social.

Seguindo o modelo do PSO, várias aplicações para problemas de otimização em espaços contínuos foram elaboradas. Em [54] pode ser visto um apanhado da técnica com sua utilização em diversos problemas, tais como Funções Multiobjetivo, Minimax, Programação Linear Inteira (*Integer Programming*) e funções com ruído e mudança contínua, mostrando a robustez e a versatilidade da técnica. Para espaços de busca discretos, temos algumas contribuições, como a de KENNEDY e EBERHART [55], na qual é desenvolvida uma versão discreta binária para o algoritmo do PSO onde as partículas apresentam os valores de vetores binários de comprimento n, com as velocidades representando as probabilidades de mudança do valor do *bit*; WANG et al. [56] que apresentam uma aplicação na solução do TSP, com base em Seqüências de Troca (*Swap Sequences*) e Operadores de Troca (*Swap Operators*), mostrando

resultados para um TSP contendo 14 cidades; CLERC [57], com uma forma de adequar o PSO para solucionar um TSP com 17 cidades, usando estratégias específicas; YIN [58], que adaptou o PSO discreto de [55] para determinar aproximações poligonais ótimas de curvas digitais; e SALMAN et al. [59], que aplicaram o PSO ao problema de alocação de tarefas (*Task Assignment Problem*, TAP), simplesmente truncando os números reais obtidos, transformando-os em inteiros. Em particular esta última aplicação e o fato de não ser utilizada a parte decimal dos números encontrados trouxeram importantes contribuições para o desenvolvimento da metodologia apresentada neste trabalho. Vale ressaltar ainda, que em nenhum destes trabalhos houve aplicação do PSO à OGCIN nem aos problemas combinatórios correlatos Oliver30 e ry48p (Rykel48).

Devemos, no entanto, destacar que os algoritmos baseados em PSO que citamos a seguir determinam um critério específico para busca local no TSP. O fato de que heurísticas locais para mudanças na ordem de cidades adjacentes proporcionam relativo sucesso para um algoritmo para o TSP não significa que o mesmo possa ser utilizado diretamente para o OGCIN. Não há uma relação direta pelo seguinte motivo: quando a ordem de duas cidades em um TSP é mudada localmente, o caminho resultante pode ser menor ou maior, mas é sempre possível. No caso da OGCIN, quando dois ECs adjacentes são mudados de posição, isto pode levar a totalmente diferentes distribuições de potência e, portanto, até mesmo configurações de núcleo inválidas, dependendo de sua nova posição no núcleo. Assim, não há prova de que heurísticas locais no PSO poderiam ser bem sucedidas para a OGCIN. E mesmo para provar a afirmativa de que heurísticas locais são bem-sucedidas para o PSO aplicado à OGCIN, é necessário fazer um estudo anterior completo sobre o PSO sem heurísticas locais. Estas são as razões pelas quais nós não adotamos heurísticas nesta etapa do trabalho. No capítulo 5 será apresentada a Heurística de Aceitação de Vizinhança Reativa para a OGCIN e seus resultados.

Neste sentido, nosso trabalho é diferente dos abaixo, que adotam mecanismos de busca locais:

- PSO discreto com operador *delete-crossover* – As equações do PSO são redefinidas considerando as posições como caminhos do TSP e a diferença entre as posições são decompostas em uma ou mais transposições [60]. O operador

delete-crossover é aplicado à melhor posição global (o melhor caminho) como heurística local para melhorar o desempenho do enxame.

- PSO modificado baseado em transformação do espaço – Os autores usam as heurísticas de busca local opt-2 e opt-3 e uma *operação caótica* [61].

Enfim, o algoritmo PSO desenvolvido neste trabalho tem como finalidade possibilitar sua utilização na otimização da recarga de combustível em reatores nucleares, tendo sido especificamente criado com este fim.

3.2.2. Formulação Matemática do PSO

Como veremos adiante, a formulação matemática do PSO representa justamente o que foi mencionado acima, relativo à seu processo de otimização: cada indivíduo é influenciado durante a busca pela sua própria experiência, bem como pela experiência do grupo, balanceando aprendizado individual e cognição social.

Um enxame com P partículas otimiza em um espaço de busca n-dimensional. Cada partícula i tem uma posição $\boldsymbol{x_i^t} = [x_{i1}\ x_{i2}\ \ldots\ x_{in}]$ e uma *velocidade* $\boldsymbol{v_i^t} = [v_{i1}\ v_{i2}\ \ldots\ v_{in}]$ em uma iteração t, através da dimensão j atualizadas de acordo com as equações

$$v_{ij}^{t+1} = w^t v_{ij}^t + c_1 r_1^t\,(pbest_{ij} - x_{ij}^t) + c_2 r_2^t\,(gbest_j - x_{ij}^t) \tag{3.1}$$

$$\text{e} \quad x_{ij}^{t+1} = x_{ij}^t + v_{ij}^{t+1}, \tag{3.2}$$

$$\text{onde} \quad i = 1, 2, \ldots, P \tag{3.3}$$

$$\text{e} \quad j = 1, 2, \ldots, n. \tag{3.4}$$

O peso de inércia w^t pode decrescer linearmente de acordo com a equação

$$w^t = w - \frac{w - w_{\min}}{t_{\max}} t\ , \tag{3.5}$$

onde w é a constante de inércia máxima, t_{max} é o número máximo de iterações. Altos valores de w^t favorecem uma busca global, enquanto baixos valores de w^t favorecem uma busca local.

No 2o. membro da equação (3.1), o primeiro termo representa a influência do movimento da própria particula, agindo como uma memória do comportamento anterior da particular; o 2o. termo representa a cognição individual, onde a partícula i compara sua posição com sua melhor posição $\boldsymbol{pbest_i}$; e o terceiro termo representa o aspecto social da inteligência, baseado em uma comparação entre a posição da partícula e o melhor resultado obtido pelo enxame ***gbest***. A equação (3.2) descreve como as posições são atualizadas. Ambos c_1 e c_2 são constantes de aceleração: c_1 está relacionado à congnição individual enquanto c_2 está relacionado ao aprendizado social; r_1 e r_2 são números aleatórios de uma distribuição uniforme de probabilidades. A inicialização do algoritmo é aleatória, isto é, as posições e velocidade são inicializadas aleatoriamente. O algoritmo é descrito na Fig. 3.1.

```
1. Inicialização
     1.1. Para cada partícula i em uma população de tamanho P:
            1.1.1. Inicializar x_i^t aleatoriamente.
            1.1.2. Inicializar v_i^t aleatoriamente.
            1.1.3. Avaliar a fitness f(x_i).
            1.1.4. Inicializar pbest_i com uma cópia de x_i.
     1.2. Inicializar gbest com uma cópia de x_i com a melhor
fitness.

2. Repetir até que um critério de parada seja satisfeito:
     2.1 Para cada partícula i:
            2.1.1. Atualizar v_i^t e x_i^t de acordo com as
      equações(3.1) e(3.2).
            2.1.2. Avaliar f(x_i^t).
            2.1.3. Se f(pbest_i) < f(x_i^t) então pbest_i ← x_i^t .
            2.1.4. Se f(gbest_i) < f(x_i^t) então gbest_i ← x_i^t .
```

Figura 3.1 – Algoritmo básico do PSO.

As posições $\boldsymbol{x_i^t}$ atualizadas pelas equações (3.1) e (3.2) são avaliadas por uma função de fitness do problema $f(\boldsymbol{x_i^t})$, que, no caso dos estudos aqui realizados, corresponde à equação (2.13). Os vetores de posição $\boldsymbol{gbest} = [gbest_1\ gbest_2\ ...\ gbest_n]$ e $\boldsymbol{pbest_i} = [pbest_{i1}\ pbest_{i2}\ ...\ pbest_{in}]$ são atualizados dependendo da informação adquirida pelo enxame, construindo seu conhecimento no espaço de busca ao longo das iterações.

3.2.3. Particle Swarm Optimization com Random Keys

O modelo de chaves aleatórias (*Random Keys*, RK) [28] mapeia um vetor composto de números reais em uma solução contendo números inteiros não repetidos. Assim, soluções possíveis podem ser formadas em um problema de otimização, por exemplo.

Nos trabalhos desenvolvidos, é usada a abordagem *Single Machine Scheduling Problem* [28]. O mapeamento do vetor $\boldsymbol{S_A}$ = *[0,39 0,12 0,54 0,98 0,41]* resulta no vetor $\boldsymbol{V_A}$ = *[2 1 5 3 4]*, já que 0,12 é o menor número, está na 2a. posição; 0,39 corresponde à 1ª. posição e assim por diante.

SALMAN et al. [59] aplicaram o PSO à otimização do TAP truncando as coordenadas do vetor de posições para obter soluções. Vale lembrar que no TAP é possível a repetição de números inteiros em uma solução candidata. No entanto, já que a repetição de ECs não é permitida na OGCIN, usamos o vetor de posições $\boldsymbol{x_i^t}$ é usado como o vetor de chaves para gerar soluções candidatas possíveis, como é demonstrado na figura 3.2. Além disso, o truncamento para gerar soluções poderia causar distorções ou perda de informação longo do processo de otimização.

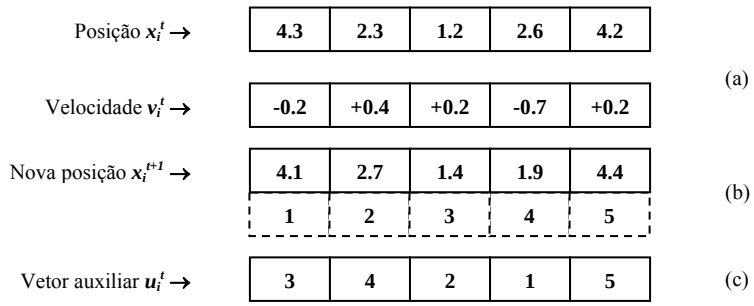


Figura 3.2 – (a) Os vetores posição e velocidade de uma partícula para um espaço de busca de 5 dimensões. (b) Novas posições como chaves aleatórias. (c) Vetor auxiliar contendo uma solução possível, a ser avaliada pela função objetivo do problema.

A principal adaptação é, portanto, a interpretação da posição. Ela representa um conjunto de chaves que permite o mapeamento da informação adquirida ao longo das iterações. Assim, as posições não necessitam ser truncadas ou arredondadas como na aplicação do PSO ao TAP. A informação do vetor de posições é mapeada pelo RK resultando em uma solução possível a ser avaliada. Desta forma, um espaço de busca discreto é obtido, com soluções possíveis para a OGCIN a partir de um espaço real contínuo, onde o PSO consegue resultados consideráveis [54]. Isto significa que, com RK, uma busca em um espaço contínuo real gera soluções para problemas combinatórios onde não é possível a repetição de elementos.

Para a OGCIN, o Particle Swarm Optimization com Random Keys (PSORK) foi modelado com as partículas do enxame tendo uma posição $\boldsymbol{x_i^t}$ em um espaço de busca de 20 dimensões, como mostra a figura 3.3. A cada iteração do algoritmo, estas posições são comparadas à melhor posição obtida pelo enxame até o momento, bem como com a posição obtida pela própria partícula de acordo com as equações (3.1), (3.2), (3.3), (3.4) e (3.5). As posições são usadas como chaves para o mapeamento e desta forma obtém-se o vetor auxiliar $\boldsymbol{u_i}$ com 20 números, de 1 até 20 sem repetição, de acordo com o mapeamento RK mostrado na figura 3.3. O vetor é passado ao código de Física de Reatores RECNOD para avaliação. Desta forma, cada partícula representa um PC possível, candidato a solução. O mapeamento permite obter soluções candidatas para a OGCIN com simetria 1/8 de núcleo a partir de um espaço de busca de 20 dimensões, e o código RECNOD avalia cada um deles, fornecendo C_B and P_{rm} para a função objetivo agregada da equação (2.6).

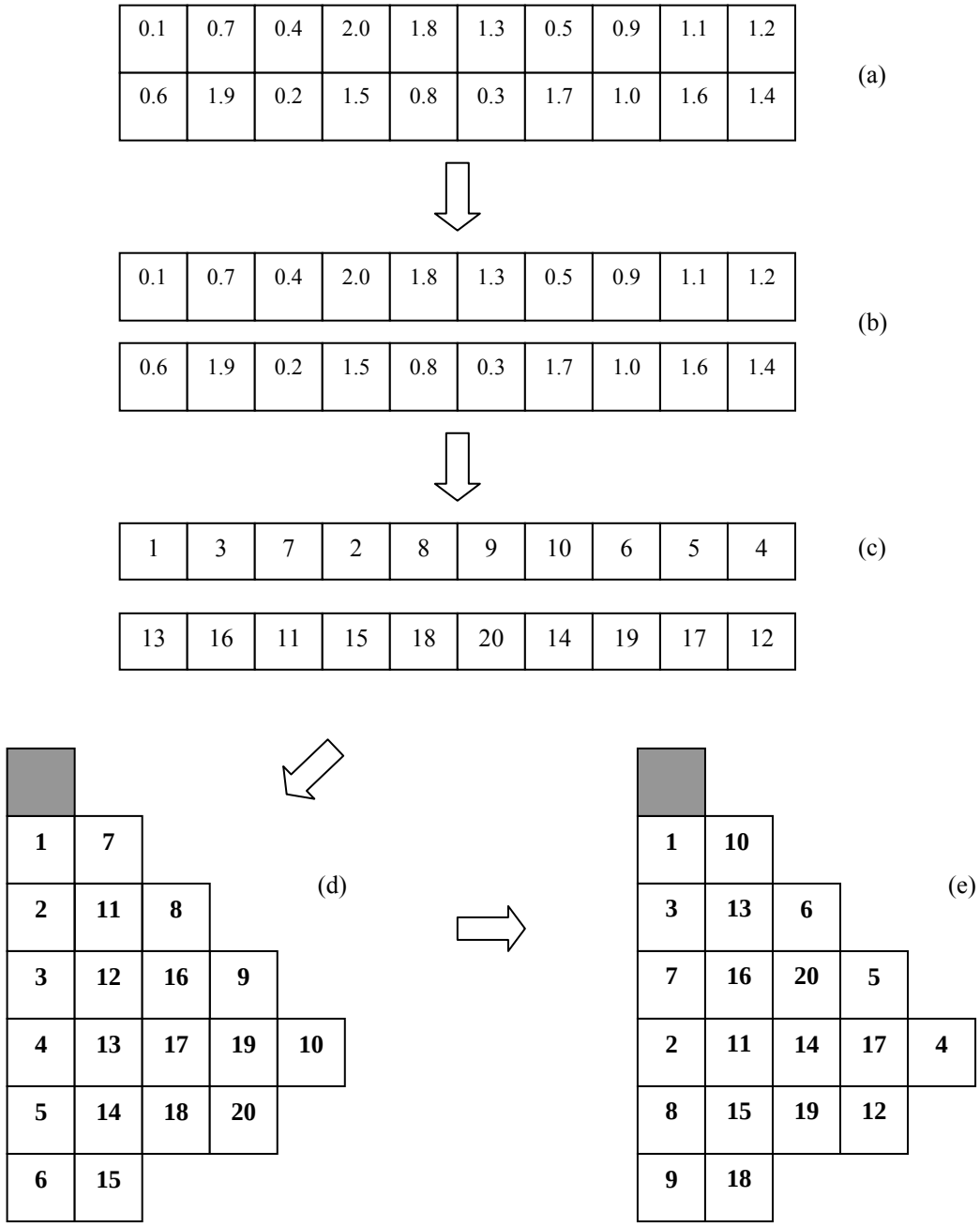


Figura 3.3 – (a) Posição x_i^t da i-ésima particular em um espaço de 20 dimensões reais na iteração *t*. (b) Divisão das chaves em dois grupos. (c) As dez primeiras chaves fazem o mapeamento dos quartetos; as 10 outras chaves fazem o mapeamento dos octetos. (d) A configuração base para a simetria de 1/8. (e) O núcleo resultante em simetria de 1/8 de núcleo, a ser avaliado pelo código de Física de Reatores.

3.3. Heurística de Aceitação de Vizinhança Reativa e Busca Baseada em Classes

Nesta seção discutiremos a HAVR e a BBC. O desenvolvimento conceitual e prático da BBC teve particular motivação no sucesso da abordagem da HAVR como critério de seleção das soluções candidatas a serem avaliadas durante a busca realizada por MHOs. Este é o motivo de as partes conceituais de ambas encontrarem-se nesta mesma seção. Os resultados e comentários sobre a aplicação da HAVR à RS e ao PSO na OGCIN estarão no capítulo 6. Os resultados e comentários sobre a aplicação da BBC na OGCIN estarão no capítulo 7.

3.3.1. Heurística de Aceitação de Vizinhança Reativa

Como discutido anteriormente, uma importante característica da OGCIN é que, além da complexidade do problema, a avaliação das soluções candidatas também é complexa. Para fazer a avaliação, códigos de Física de Reatores com implementações das soluções numéricas das equações dos modelos de Transporte ou Difusão de nêutrons. Desta forma, o tempo para avaliar uma solução candidata para o problema é proibitivo para o processo de otimização.

Um dos objetivos desta tese é relatar o desenvolvimento e resultados da Heurística de Aceitação de Vizinhança Reativa (HAVR). Já que essa abordagem é baseada na classificação de PCs de acordo com os ECs mais reativos, a heurística pode ser usada independentemente do ciclo, modelo de PWR ou método de busca utilizado. Aplicamos a HAVR à RS e ao PSO e comparamos os resultados a outros trabalhos apresentados na literatura.

De acordo com PEARL [65], “heurísticas são critérios, métodos ou princípios para decidir qual dentre vários cursos alternativos de ação promete ser o mais efetivo para atingir algum objetivo”. Vários métodos de solução da OGCIN usam heurísticas

para achar PCs próximos a ótimos, entre os quais, métodos baseados em conhecimento e MHO com heurísticas em sua abordagem.

Considerando os métodos baseados em conhecimento em Engenharia Nuclear, alguns artigos são notáveis. NAFT e SESONSKE [40], por exemplo, usam várias regras lógicas para gerar novos PCs candidatos, ou ainda, as mudanças de posições são restritas de acordo com esse conjunto de regras. SUH e LEVINE [66] usam simples regras diretas de otimização, e os autores introduziram três passos em direção a um projeto de PCs: determinar um PC de início de ciclo (*Beginning of Cycle*, BOC) consistente com uma Distribuição de Potência de Haling, correlacionar ECs disponíveis usando o k_∞ como as variáveis de correlação e otimizar o VQ para seguir a distribuição ótima de Haling. Em um outro trabalho, o protótipo FUELCON descrito por GALPERIN e KIMHY [42] usa algumas regras SE-ENTÃO como "combustível fresco posicionado na periferia do núcleo, i.e., posições com a maior distância do centro do núcleo", entre outras.

Dois exemplos de trabalhos sobre MHOs que usam heurísticas nucleares são descritas por LIN et al. [13] e STEVENS et al. [3]. Na primeira destas referências, um programa de computador foi desenvolvido para a OGCIN, usando um conjunto de regras heurísticas para determinar os PCs candidatos de acordo com o algoritmo da TS. Na segunda referência, heurísticas são usadas de acordo com o arcabouço do SA, reforçando restrições a transições.

STEVENS et al. [3] também comentam que heurísticas geométricas simples limitam a geração de novos padrões de uma maneira direta. Essas heurísticas se referem à colocação de classes específicas de ECs (combustível duplamente queimado, combustível fresco com VQ) em certas posições do núcleo (cantos periféricos, periferia do núcleo). Por outro lado, heurísticas relacionais se referem a posições relativas não levando em consideração uma posição absoluta no núcleo. Como um exemplo, a classe de combustíveis frescos não deve ter nenhum de seus pares em posições adjacentes no interior do núcleo. Ademais, cada candidato gerado tem que satisfazer todos os requisitos com relação às heurísticas.

Estes dois exemplos refletem como as heurísticas têm sido usadas associadas a MHOs. De fato, elas podem ser usadas como heurísticas de construção como definido por LIN et al. [13] para o TS. Ou ainda, um procedimento pode verificar e garantir que uma solução candidata a ser avaliada satisfaça heurísticas, como descrito por STEVENS et al. [3].

Todavia, a proposição aqui apresentada é diferente de ambos exemplos acima. É uma terceira maneira de aplicar uma heurística a uma MHO. Além do mais, também é apropriada ao tempo proibitivo da avaliação de uma solução candidata usando códigos de Física de Reatores.

No primeiro exemplo, quando um conjunto de regras heurísticas é usado para gerar soluções candidatas, um problema que pode ocorrer é que algumas regiões do espaço de busca não serão exploradas. Além disso, de acordo com STEVENS et al. [3], "heurísticas freqüentemente criam regiões disjuntas de soluções possíveis", e no caso da OGCIN, não somente devido à introdução de heurísticas durante a busca, mas de fato também por causa das características do espaço de busca, que é altamente complexo. Assim, embora o uso de heurísticas na construção de soluções possa ser bem-sucedida, não há garantia de uma busca global efetiva nesse caso, o que é essencial para achar soluções próximas a ótimas, e o risco é ainda pior se a proibição leva a um desvio sistemático para as mesmas regiões, induzido pelas regras heurísticas.

No segundo exemplo, quando somente soluções candidatas que satisfazem a todos os requisitos das regras heurísticas são avaliados, não há flexibilidade e talvez algumas das características interessantes de soluções que embora não sejam aceitáveis do ponto de vista de segurança, por exemplo, não podem ser memorizadas durante o processo meta-heurístico de otimização. Embora restrições absolutas também sejam bem-sucedidas, ao fazê-las, impossibilita-se o algoritmo de memorizar características que poderiam ser úteis no processo de otimização.

De maneira contrária, existe a possibilidade de redução da quantidade de avaliações realizadas pelo código de Física de Reatores, dando flexibilidade para a aplicação de uma heurística. Isto causa de fato uma aproximação durante a busca, mas não tão radical como proibir a avaliação de alguma LP que não satisfaça aos requisitos das heurísticas. A idéia principal com essa consideração é que se torna possível restringir parcialmente regiões do espaço de busca, permitindo à MHO memorizar características das soluções que de outra forma seriam desprezadas, e ao mesmo tempo é possível reduzir o número de avaliações realizadas pelo código de Física de Reatores bem como melhorar a qualidade das soluções a longo prazo.

Em outras palavras, o uso desta nova abordagem de heurísticas guiando o processo de busca satisfaz, entre outros, um importante requisito da OGCIN: está de acordo com a alta complexidade do problem no senso que reduz o número de avaliações necessárias, e portanto o custo computacional.

Para as implementações e para demonstrar a viabilidade do método, usamos apenas um critério para ser verificado durante a otimização. É baseado em uma regra para engenheiros de otimização de PCs na formação de um PC "tabuleiro de xadrez", também citado na literatura como "não carregar um par de elementos frescos vizinhos no interior do núcleo" [3] ou "ter dois ECs frescos adjacentes um ao outro no interior do núcleo é absolutamente não permitido" [13].

Todavia, a abordagem aqui discutida considera não somente um PC com dois ECs frescos adjacentes como indesejáveis na formação de um PC em termos de segurança. Também considera duas classes de PCs, os mais reativos e os menos reativos ECs e sua posição relativa em um PC candidato a solução, com base no ciclo 7 de Angra 1. A tabela 3.1 exibe a posição do PC de referência com os ECs e seus respectivos tipos [36] de acordo com suas características nucleares, como enriquecimento, por exemplo. A coluna *Posição* se refere às posições mostradas na figura 3.3. Essa figura, por sua vez, contém o PC de referência e as classes de cada EC representado (mais reativo – 0, e menos reativo – 1). Os ECs de tipos 2, 5 e 6 foram classificados como 0 e os ECs dos tipos 1, 3 e 4 foram classificados como 1, como indicado na coluna *Classe* da tabela 1, que contém as classificações descritas. Em nosso modelo, vale frisar que um EC correspondente a um quarteto deve apenas ser trocado

com um outro EC de quarteto. ECs correspondentes a octetos, de modo similar, devem apenas ser trocados com ECs octetos.

Tabela 3.1 – Dados do PC de referência para o ciclo 7 de Angra 1 em simetria de 1/8, com as classes de cada EC.

	Posição	Tipo	Classe
Quartetos	(2, 1)	6	0
	(3, 1)	4	1
	(4, 1)	4	1
	(5, 1)	4	1
	(6, 1)	2	0
	(7, 1)	2	0
	(2, 2)	5	0
	(3, 3)	4	1
	(4, 4)	1	1
	(5, 5)	2	0
Octetos	(3, 2)	4	1
	(4, 2)	4	1
	(5, 2)	4	1
	(6, 2)	3	1
	(7, 2)	5	0
	(4, 3)	6	0
	(5, 3)	6	0
	(6, 3)	3	1
	(5, 4)	5	0
	(6, 4)	2	0

A fim de estipular se um PC tem uma vizinhança mais ou menos reativa e se vale a pena ou não avaliá-lo, define-se o parâmetro *grau* como o número de "interfaces" entre os ECs mais reativos dentro de uma certa área da simetria de 1/8 de núcleo, isto é, a adjacência vertical ou horizontal entre dois ECs da classe 0. Como é mostrado na figura 3.3, para a região interior do núcleo (ECs em cinza), quando a subrotina identifica que dois ECs muito reativos são vizinhos verticalmente ou horizontalmente, o parâmetro *grau* é aumentado em uma unidade. O PC mostrado na figura 3.3 tem grau 2. Ele tem dois ECs muito reativos que são vizinhos horizontalmente nas posições (2, 1) e (2, 2), e dois outros ECs vizinhos verticalmente, nas posições (4, 3) e (5, 3). Outro exemplo: supondo que o EC na posição (3, 3) é trocado com o EC na posição (7, 1), e que o EC em (3, 2) é trocado com o EC em (6, 4), o grau deste novo PC seria 5 (figura 3.4), e provavelmente, este seria um PC com altos picos de potência, dado que existem mais ECs da classe 0 que são adjacentes.

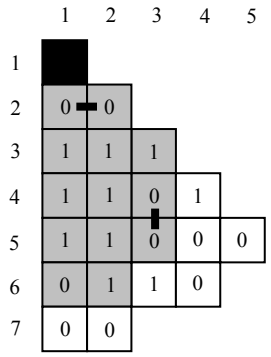


Figura 3.3 – Representação da simetria de 1/8 de núcleo para o ciclo 7 de Angra 1, com as classes de cada EC, de acordo com a tabela 3.1. Os ECs nas posições em cinza terão seu *grau* avaliado. Esse PC tem grau 2 já que apenas dois pares dos ECs mais reativos (classe 0) são adjacentes.

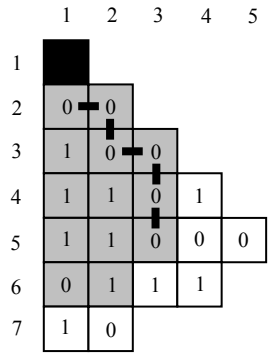


Figura 3.4 – Representação da simetria de 1/8 de núcleo para um PC de grau 5.

A idéia básica é que um PC com grau alto implica que sua avaliação não é vantajosa. Por outro lado, um PC com grau menor, que pode conter caractersticas interessantes para o processo de otimização, não é descartado, mas avaliado. Em outras palavras, o algoritmo usa essa regra heurística para verificar se a avaliação de um PC é vantajosa ou não, mas de acordo com diferentes graus.

Assim, essa heurística pode direcionar a otimização para PCs que são interessantes para o processo, já que o posicionamento de ECs é crítico para a aquisição de PCs possíveis. Além disto, este conhecimento é consistente com julgamento baseado em conhecimento em Engenharia Nuclear.

A principal diferença entre nossa proposta e as de outros trabalhos é que são atribuídos graus e portanto tolerância para PCs que de outra forma não seriam avaliados, de acordo com critérios de exclusão. As vantagens são que esta heurística pode ser usada com qualquer MHO, economiza esforço computacional com base em conhecimento em Engenharia Nuclear, mas com uma regra simples em algumas linhas de código, pode-se permitir que características interessantes de PCs com baixo grau, embora não possíveis não são descartados e portanto úteis para a otimização. Ademais, outras heurísticas geométricas e relacionais podem ser adaptadas para uso na OGCIN usando esta abordagem, que para fins de terminologia foi chamada Heurística de Aceitação de Vizinhança Reativa (HAVR).

No caso do algoritmo RS-HAVR, para cada PC candidato a solução gerado aleatoriamente, pôde-se usar a classificação dada na tabela 3.1 e checar o grau do PC, decidindo se é vantajoso ou não avaliá-lo de acordo com um máximo grau estipulado. Para o PSO-HAVR, de acordo com o mapeamento descrito na subseção 3.2.3, por exemplo, se uma partícula alcança uma posição descrita na figura 3.5a então suas coordenadas serão mapeadas em um conjunto de inteiros sem repetição (figuras 3.5b e 3.5c) que representa um PC (figura 3.5d). A figura 3.5e é a representação deste PC de acordo com a classificação de ECs nas classes 0 e 1, com respeito à tabela 3.1. Os ECs da classe 0 que têm interfaces com outros ECs da mesma classe dentro da região limitada pela linha mais grossa da figura. Assim, este PC tem grau 2 e não seria avaliado em um experimento cujo máximo grau de aceitação fosse 1. Por outro lado, seria avaliado em experimentos com um máximo grau 2 ou 3.

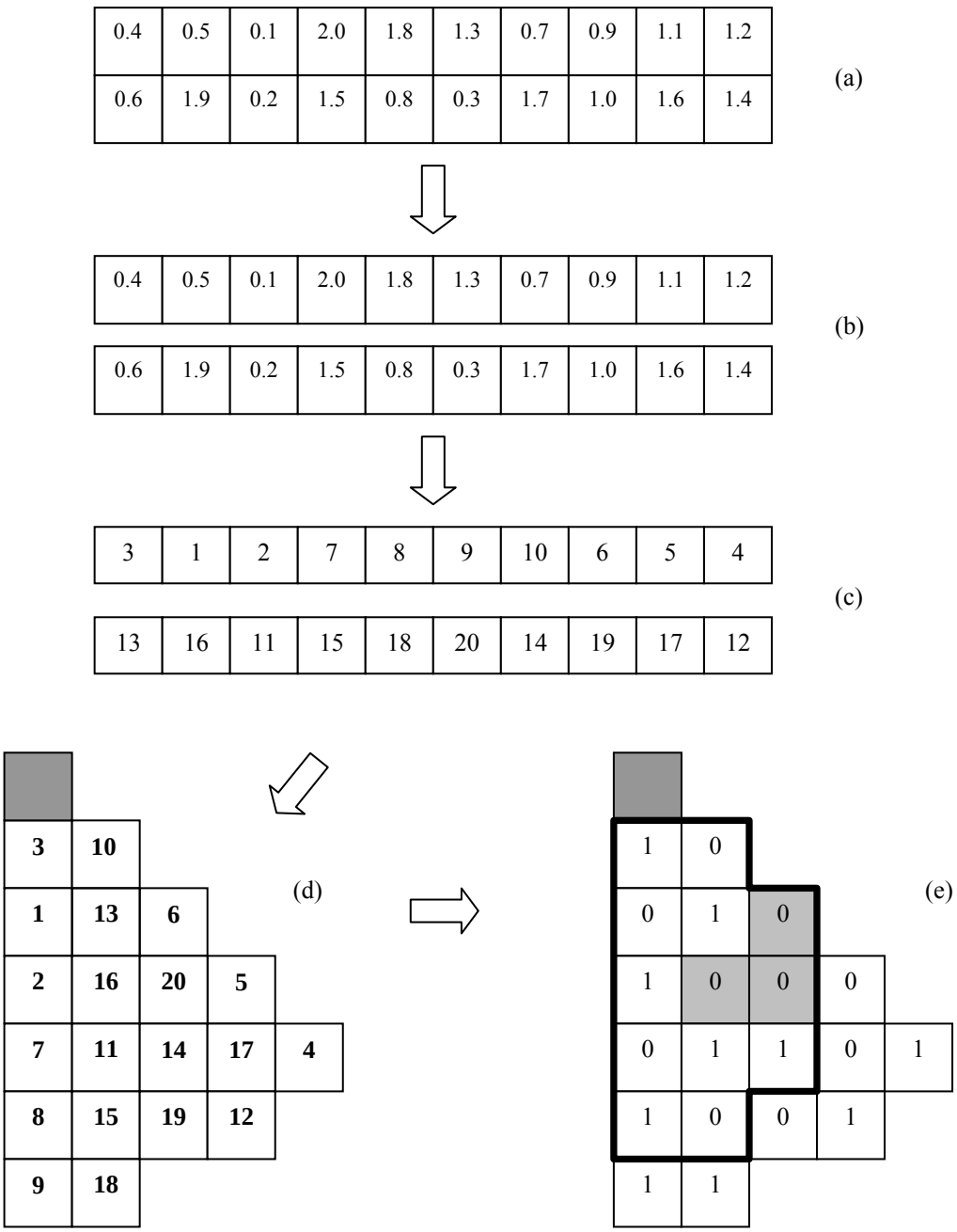


Figura 3.5 – (a) Posição de uma particula no PSO; (b) separação do vetor devido à troca de quartetos com quartetos e octetos com octetos; (c) as 10 primeiras chaves representam uma solução para os quartetos; as outras 10 chaves serão decodificadas para obter uma solução para os octetos; (d) o PC final; (e) a representação deste PC de acordo com as classes da tabela 3.1: o PC gerado tem grau 2. Em cinza os ECs da classe 0 que têm interfaces com outros ECs da mesma classe dentro da região limitada pela linha mais grossa.

3.3.2. Busca Baseada em Classes

3.3.2.1. A BBC e suas motivações: PCs que apresentam características similares e Busca Local

MHO aplicadas à OGCIN como GA [12] e PSO [23] demostraram excelente capacidade de busca global. Não obstante, uma abordagem baseada em busca local poderia ser muito útil neste problema. Esta é a razão do desenvolvimento da BBC, aliada ao sucesso dos resultados da HAVR que serão vistos no capítulo 6. A principal idéia da BBC foi explorar as características nucleares dos PCs a serem carregados no núcleo, especialmente a reatividade, para pequenas mudanças nas configurações, embora não usando uma abordagem determinística pois a mesma se torna inaplicável devido à alta complexidade do problema da OGCIN.

Intuitivamente, uma idéia seria obter, a partir de um PC *corrente* (usado neste sentido para designar a melhor solução candidata de uma determinada iteração ou o padrão inicial selecionado aleatoriamente) por meio de Trocas Binárias (*Binary Exchanges,* BE), ou seja, trocas entre pares de FAs de um mesmo PC. Assim, o melhor PC obtido é usado como base para a próxima iteração do algoritmo. Essa abordagem e seus resultados foram descritos por YAMAMOTO [67] para a OGCIN de um PWR, com uma comparação quantitativa das técnicas de Busca Direta (*Direct Search*, DS), SA, GA e BE, bem como métodos híbridos tais como GA+BE e DS+BE. De acordo com o autor, a BE não tem capacidade de escapar de mínimos locais, embora faça a busca local mais refinada e que o melhor PC final é "ao menos garantido como um ótimo local para a troca binária de ECs". A figura 3.6 exibe o algoritmo *OGCIN-BE* implementado neste trabalho.

```
1. Inicialização: gerar uma solução corrente aleatoriamente.

2. Repetir até que um critério seja satisfeito:
      2.1. Gerar a Vizinhança de soluções candidatas para a solução
      corrente, composta pelos PCs derivados da solução corrente
      pelas BEs (contanto que tais soluções não tenham sido geradas
      e avaliadas anteriormente).
      2.2. Avaliar a fitness de cada PC da Vizinhança de acordo com
      a equação (2.13).
      2.5. Determinar o melhor PC da Vizinhança.
      2.6. solução corrente ← melhor PC da Vizinhança.
```

Figura 3.6 – Algoritmo do *OGCIN-BE*.

Assim, executamos vários testes e foi possível verificar que a troca de dois ECs com características similares em diversos casos não resulta em modificações dramáticas de dois ECs com características similares nos parâmetros nucleares no seguinte sentido. Por exemplo, o PC A, mostrado na figura 3.7 tem os parâmetros nucleares $P_{rm} = 1{,}472$ e $C_B = 1510$. Na prática, com a troca de dois ECs correspondentes às posições (4, 1) e (5, 5), é possível obter um PC B, cuja avaliação resulta no parâmetro nuclear $P_{rm} = 1{,}471$. Neste exemplo, ambos os PCs não são válidos em termos do parâmetro relacionado à segurança já que P_{rm} é maior que $1{,}395$.

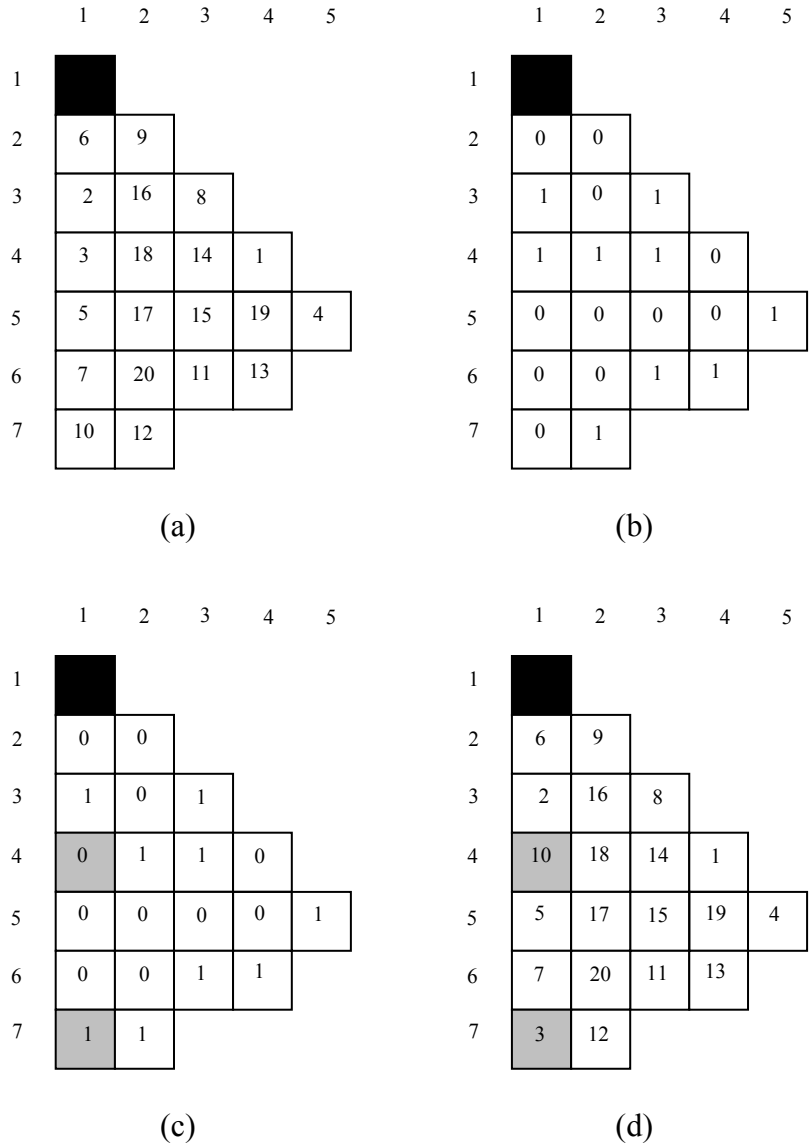


Figura 3.7 – (a) Esquema de representação da simetria de 1/8 de núcleo do PC A; (b) a representação de classes correspondente para a BBC de acordo com a tabela 3.1; (c) o PC baseado em classes resultante para uma troca entre os ECs (4, 1) e (7, 1), que são de diferentes classes; (d) o PC resultante obtido com a abordagem da BBC.

Em vez de usar a BE já que esta possui limitações derivadas do aprisionamento em regiões de mínimos locais com soluções candidatas muito similares como as descritas no exemplo acima, a idéia da BBC é examinar PCs obtidos de um PC corrente por meio de trocas binárias, embora de uma maneira mais inteligente: as trocas são feitas com base nas características nucleares dos ECs, o que causa uma melhor exploração do espaço de busca.

Em outras palavras, se os ECs são classificados em dois grupos (os mais reativos – classe 1 – e os menos reativos – classe 0) seria possível executar uma busca local com mudanças importantes na estrutura do PC, com base nas características aproximadas dos ECs. A tabela 3.1 exibe a classificação usada para os ECs do 7o. ciclo da Usina de Angra 1, e a figura 3.7b mostra a representação do PC A com base nesta classificação, ou seja, com o padrão de classes binárias.

Por exemplo, em uma abordagem como a BE, se o PC A é a solução corrente, então o PC B do exemplo acima mencionado seria um membro da vizinhança do PC A; portanto, já que o algoritmo avalia todas as soluções derivadas da solução corrente, então o PC B seria avaliado. Esse tempo computacional gasto para avaliação de PCs similares tem um alto custo e não necessariamente leva a busca a soluções próximas a ótimas. Com a abordagem da BBC, os ECs do PC A são classificados como mostrado na figura 3.7b. Assim, levando isto em consideração, os PCs que provavelmente são extremamente similares ao PC A não vão fazer parte de sua Vizinhança, de modo que o algoritmo pode realizar uma melhor exploração do espaço de busca. A idéia é que a busca pode ser executada de uma maneira aproximada porém efetiva: a cada iteração, dentro do conjunto de PCs derivados (a Vizinhança), a busca é baseada na seleção e avaliação dos PCs que correspondem a uma troca entre ECs de diferentes classes, consideradas mais importantes e relevantes no contexto da busca metaheurística.

Como mencionado anteriormente, a BBC é baseada na troca de ECs de duas classes distintas (a classe mais reativa e a classe menos reativa) e não entre ECs com características nucleares aproximadas, como a BE permite. Logo, é uma MHO baseada em mudanças relevantes em um PC. Por exemplo, nesse modelo, o PC A da figura 3.7a tem 20 ECs que poderiam ser permutados. De acordo com a tabela 3.1, os 10 ECs quartetos são compostos de 5 ECs de classe 0, bem como 5 ECs da classe 1. O mesmo é

verdade para os ECs octetos, e, portanto, sua representação baseada em classes, como indicado anteriormente é mostrada na figura 3.7b. Como a BBC é baseada em trocas binárias, considerando, por exemplo, uma troca entre os ECs de duas classes distintas, os ECs das posições (4, 1) e (7, 1) na figura 3.7b, a representação baseada em classes é mostrada na figura 3.7c. Desta forma, o PC resultante que pertence à Vizinhança do PC A de acordo com a troca feita com a abordagem BBC é mostrada na figura 3.7d.

Para a simetria de 1/8 de núcleo, considerando somente as trocas binárias de acordo com a abordagem BBC e sem trocas entre ECs quartetos e octetos, é possível mostrar que o número de PCs na vizinhança do PC A é 50, já que as trocas de ECs vão ocorrer ou entre dois ECs quartetos, onde existem 25 possibilidades, ou entre dois octetos, onde existem outras 25 possibilidades, totalizando as 50 soluções candidatas vizinhas baseadas em classe, mencionadas anteriormente, que podem ser avaliadas pelo código de Física de Reatores, contanto que não tenham sido gerados e avaliados anteriormente ao longo do processo de busca, de acordo com nosso modelo.

Dois principais tipos de BBC com relação ao critério de seleção da solução corrente (o melhor PC a cada iteração) foram testados, com algumas variantes. O primeiro foi a estratégia *gananciosa* (*Greedy*) para a BBC (*BBC-Greedy*). Neste algoritmo, no final de cada iteração, a melhor solução candidata de uma Vizinhança (de acordo com a função de fitness da equação 2.13) é copiada para a solução corrente e nenhuma informação concernente a avaliações precedentes de PCs é mantida. A segunda variante retém em uma lista a informação dos PCs vizinhos que não chegaram a ser a solução corrente de alguma iteração, mas que foram selecionadas e avaliadas. Esta segunda abordagem além de lidar com a memorização de padrões intrínsecos das soluções, também lida com a memorização de soluções candidatas, e é chamado *BBC-List*. Os algoritmos são respectivamente descritos nas figuras 3.8 e 3.9.

1. Inicialização: gerar uma solução corrente aleatoriamente.

2. Repetir até que um critério seja satisfeito:
    2.1. Gerar a Vizinhança de soluções candidatas para a solução corrente, composta pelos PCs derivados da solução corrente pelas trocas baseadas em classes (contanto que tais soluções não tenham sido geradas e avaliadas anteriormente).
    2.2. Selecionar um subconjunto dos PCs da Vizinhança para serem avaliados, de acordo com a variante do algoritmo.
    2.3. Avaliar a fitness de cada PCs do subconjunto definido no item 2.2 de acordo com a equação (2.13).
    2.4. Determinar o melhor PC da vizinhança.
    2.5. solução corrente ← melhor PC da vizinhança.

Figura 3.8 – O algoritmo do *OGCIN-BBC_Greedy*.

1. Inicialização: gerar uma solução corrente aleatoriamente.

2. Repetir até que um critério seja satisfeito:
    2.1. Gerar a Vizinhança de soluções candidatas para a solução corrente, composta pelos PCs derivados da solução corrente pelas trocas baseadas em classes (contanto que tais soluções não tenham sido geradas e avaliadas anteriormente).
    2.2. Selecionar um subconjunto dos PCs da Vizinhança para serem avaliados, de acordo com a variante do algoritmo.
    2.3. Avaliar a fitness de cada PCs do subconjunto definido no item 2.2 de acordo com a equação (2.13).
    2.4. Armazenar todas as soluções avaliadas no item 2.3, bem como os resultados de suas avaliações.
    2.5. Determinar o melhor PC armazenado.
    2.6. solução corrente ← melhor PC armazenado.

Figura 3.9 – O algoritmo do *OGCIN-BBC_List*.

Para testar e avaliar a robustez do algoritmo com respeito à seleção e PCs de uma vizinhança, também implementamos diferentes versões dos algorimos *BBC-Greedy* e *BBC-List*. Tais variantes diferem umas das outras com base nos critérios de como selecionar os membros de uma vizinhança que serão avaliados. Desta forma, as variantes são: (i) *BBC-Greedy_50* e *BBC-List_50*, com a avaliação de todos as soluções candidatas que são membras de uma determinada vizinhança; (ii) *BBC-Greedy_ Random10* e *BBC-List_Random10*, com 10 vizinhos selecionados aleatoriamente para serem avaliados em cada iteração; (iii) *BBC-Greedy_Local_Swaps* e *BBC-List_Local_Swaps*, com as avaliações dos subconjuntos de vizinhos que fazem trocas somente de ECs adjacentes; e (iv) com o uso da HAVR (Seção 3.3.1), a *OGCIN-BBC_Greedy_HAVR*. As variantes (i) e (ii) usam métodos triviais de seleção de soluções candidatas. As variantes (iii) e (iv) são descritas nas subseções a seguir. A principal premissa usando tais variantes é que uma apropriada seleção de vizinhos que serão avaliados pode dirigir a busca na direção de soluções próximas a ótimas de uma maneira mais efetiva.

3.3.2.2. A variante Local_Swaps para o OGCIN-BBC

Para investigar maneiras de melhorar a busca, restringimos os tipos de trocas e portanto seleção de vizinhos de uma solução corrente a ser avaliada. A variante *LocalSwaps* (trocas locais) do algoritmo *OGCIN-BBC* foi implementada para investigar como mudanças entre ECs adjacentes poderia funcionar. Os critérios adotados para tais algoritmos são descritos abaixo.

Na figura 3.10 é possível ver a representação de simetria de 1/8 com os ECs quartetos representados por letras. Por exemplo, de acordo com o critério de *LocalSwaps*, o EC F pode ser trocado apenas com o EC E. O EC E, por sua vez, pode ser trocado tanto com o EC F (como mencionado anteriormente) quanto com o EC D. Ainda para os quartetos, seguindo este raciocínio até o EC I, que pode ser trocado com os ECs H ou J, existem no total 9 possibilidades de trocas locais (Local Swaps) para os quartetos. Para os octetos mostrados na figura 3.10b, apenas trocas verticais ou horizontais foram consideradas, assim, por exemplo, o EC K pode somente ser trocado com o EC L; e o EC Q pode ser trocado com os ECs M, P, R, S, e portanto existem 12

possiblidades de trocas locais para os octetos. Logo, o número total de trocas locais seria 21.

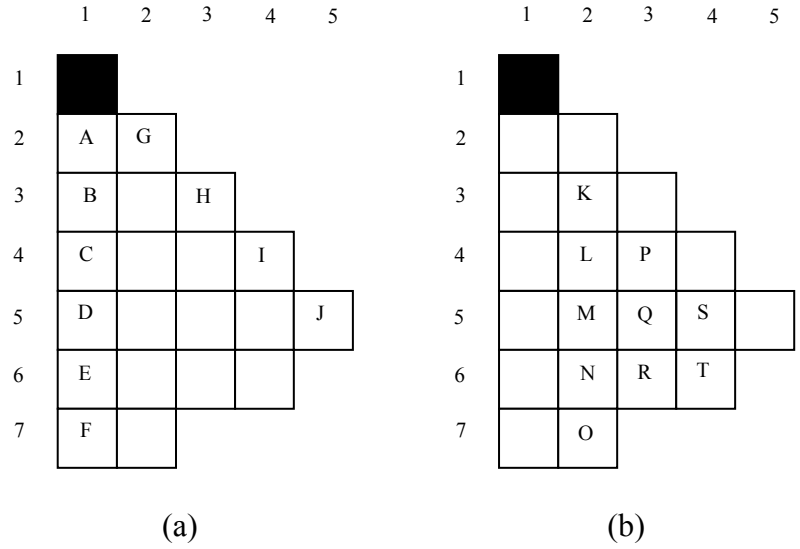


Figura 3.10 – (a) Trocas locais para os ECs quartetos: existem 9 possibilidades de trocas entre ECs que são vizinhos imediatos; (b) Trocas locais para os ECs octetos: existem 12 possibilidades de trocas entre ECs que são vizinhos imediatos.

No entanto, a abordagem BBC define o uso de classes para a representação dos ECs durante a busca. Desta maneira, com as duas classes estipuladas para nossos experimentos, o número de possibilidades reais de trocas entre ECs tende a ser menor que 21. Um exemplo é mostrado na figura 3.11, que exibe a representação das classes de um PC e as possíveis trocas locais. Neste caso, as trocas locais são somente possíveis entre certos ECs. Isto significa que somente os pares de quartetos (2, 1) e (2, 2), (6, 1) e (7, 1) podem ser trocados entre si. Para os ECs octetos: (4, 2) e (4, 3); (5, 2) e (5, 3); (6, 2) e (6, 3). E isto resulta, para este caso, em um total de cinco trocas locais. O algoritmo, então, avalia somente os PCs resultantes de tais trocas locais. Os resultados para essa variante são descritos no capítulo 7, subseção 7.1.4.

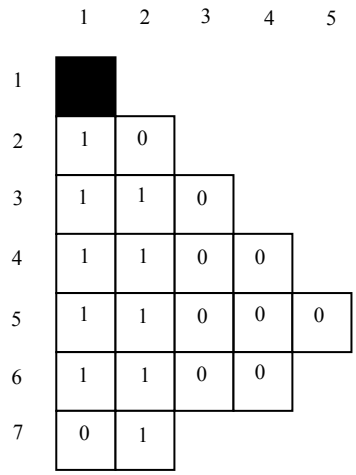


Figura 3.11 – Exemplo de PC baseado em classes e suas possíveis trocas locais. Neste caso, trocas locais são apenas possíveis entre certos ECs: para os quartetos: (2, 1) e (2, 2); (6, 1) e (7, 1); para os octetos: (4, 2) e (4, 3); (5, 2) e (5, 3); (6, 2) e (6, 3).

3.3.2.3. A variante da BBC com Heurística nuclear para a OGCIN

A HAVR (capítulo 3, seção 3.3.1) é baseada na não-avaliação de PCs que provavelmente podem resultar em núcleos muito reativos em virtude do posicionamento relativo de ECs. O parâmentro *grau* é definido como o número de adjacências verticais ou horizontais entre dois ECs da classe 0 em uma determinada área da simetria de 1/8 de núcleo, como detalhado anteriormente. Como um exemplo, as classes dos ECs de um PC com grau 5 é mostrado na figura 3.11, já que na região interior do PC (ECs em cinza), é possível verificar que existem 5 *interfaces* entre PCs da classe 0.

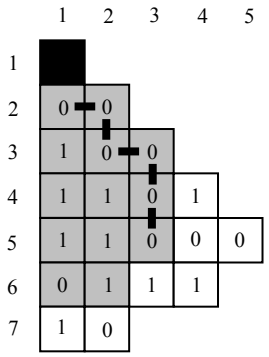


Figura 3.12 – Representação de 1/8 de núcleo para um PC baseado em classes com grau 5.

Entretanto, outra possibilidade para o uso da HAVR neste trabalho é demonstrada, indo além do seu uso para exclusão de PCs e conseqüente redução do número de avaliações. Com a HAVR pode-se selecionar um subconjunto específico de vizinhos que podem ser mais interessantes para avaliação, sendo que neste trabalho desenvolvemos e testamos duas abordagens. Na primeira delas (*OGCIN-BBC_HAVR_Greedy_MinPlus2*), com relação à redução do número de avaliações, durante a execução do algoritmo, os PCs selecionados da Vizinhança de uma solução corrente a serem avaliados foram: (a) aqueles que não foram gerados em iterações anteriores (como nas outras variantes da abordagem BBC) e (b) aqueles que apresentam como grau o menor grau da respectiva vizinhança acrescido de uma unidade (por exemplo, em uma iteração específica, se o grau mínimo de uma determinada vizinhança é 3, então os PCs com os graus 3 ou 4 serão avaliados). Na segunda abordagem (*OGCIN-BBC- HAVR_Greedy_Long_Run*), em relação a execuções mais longas direcionadas a PCs mais reativos, os PCs selecionados foram (a) aqueles que não foram gerados em iterações anteriores (como em outras variantes da abordagem BBC) e (b) os PCs com graus 5, 6 ou 7, que supostamente são PCs altamente reativos. Os resultados para estes algoritmos se encontram no capítulo 7.

Em suma, para investigar procedimentos de trocas locais para a OGCIN, já que a abordagem da BE apresenta limitações relacionadas ao aprisionamento em regiões de mínimos locais, desenvolvemos dois principais tipos de algoritmos *OGCIN-BBC*: *Greedy* e *List*. Suas variantes diferem na maneira pela qual os PCs de uma mesma

vizinhança serão selecionados para serem avaliados: (i) avaliando todos os membros de uma vizinhança; (ii) selecionando aleatoriamente 10 membros de uma vizinhança; (iii) realizando trocas locais entre vizinhos imediatos; e (iv) usando a HAVR.

3.4. Sumário do Capítulo

Este capítulo foi dedicado à apresentação de MHOs: foi apresentada uma visão geral das mesmas, com um destaque para o PSORK, aplicado à OGCIN em parte do trabalho aqui apresentado. Apesar de não ser uma MHO, apresentamos a HAVR, pois em parte do trabalho desenvolvido, tal heurística foi aplicada com RS e PSO. Além disto, o entendimento da HAVR é de fundamental importância para o entendimento da BBC. O próximo capítulo apresenta os resultados do PSORK aplicada aos TSPs Oliver30 e Rykel48 para validação e estudo de constantes para posterior aplicação à OGCIN.

Capítulo 4

Estudo de Sensibilidade da Otimização com Enxame de Partículas no Problema do Caixeiro Viajante

O TSP [47, 68] é um problema combinatório NP-Difícil [69] cujas soluções possíveis são conjuntos discretos correspondendo a uma permutação da seqüência de cidades visitadas. É um problema de referência na Teoria de Complexidade Computacional, considerado de difícil solução embora tendo uma formulação: dado um número n ≥ 3 de cidades (ou nós) e a distância entre eles, o objetivo é determinar a trajetória de menor distância total, visitando cada cidade uma vez e voltando à primeira cidade.

Há dois tipos principais de TSP: os simétricos e os assimétricos. O TSP simétrico é aquele em que, dadas duas cidades diferentes *i* e *j*, a distância d_{ij} para ir de uma para a outra é a mesma em ambos os sentidos, isto é, $d_{ij} = d_{ji} \ \forall i, j = 1, ..., n$. Desta forma, um TSP simétrico tem *[(n-1)!]/2* soluções possíveis. Para um TSP assimétrico, $d_{ij} \neq d_{ji} \ \forall i \neq j$, então há *(n-1)!* possibilidades diferentes.

O PSORK é aplicado ao TSP como apresentado na subseção 3.2.3, como apresentado na figura 3.2, como se fosse um TSP de 5 cidades, ou seja, com o número de coordenadas igual ao número de cidades do problema. Cada solução gerada tem as apresenta as cidades em uma ordem possível pois o RK não permite repetições em seu mapeamento. A avaliação da fitness é a soma das distâncias entre as cidades na ordem

apresentada. Em [31] pode ser encontrado um maior detalhamento a respeito do TSP, aspectos relevantes de Teoria de Complexidade Computacional e de como o PSO é aplicado ao TSP.

4.1. Resultados

Os experimentos para o TSP foram projetados de modo a determinar a relevância do número de partículas e a influência dos parâmetros w, c_1, e c_2 no processo de otimização, buscando aplicar o PSORK ao NRRP.

4.1.1. Particle Swarm Optimization aplicado ao TSP Oliver 30

O problema Oliver30 é um TSP simétrico de 30 cidades [48]. A menor distância conhecida é *423,74*. Os testes foram realizados para enxames de 50 e 500 partículas. Os resultados para o melhor experimento estão na tabela 4.1 (50 partículas, $c_1 = c_2 = 1,68$ and $w = 0,74$) e tabela 4.2 (500 partículas, $c_1 = c_2 = 1,68$ and $w = 0,68$). Todos os enxames usaram a topologia de vizinhança estrela (vizinhança *gbest*) [21]: se uma partícula alcança a melhor posição obtida pelo enxame, a informação é passada para todo o enxame.

Tabela 4.1 – Resultados do melhor experimento para o PSORK aplicado ao TSP Oliver 30 (50 partículas; melhor resultado conhecido 423,74).

Iteração	10	100	200	300	400	500	600	700	800	900	1000
Melhor Fitness	957,05	522,83	445,99	445,79	440,17	440,17	440,17	440,17	438,4	438,4	423,74
Fitness Média Do Enxame	1234,84	706,62	484,87	457,29	449,47	445,55	447,13	446,46	440,31	439,26	430,16

Tabela 4.2 – Resultados do melhor experimento para o PSORK aplicado ao TSP Oliver 30 (500 partículas; melhor resultado conhecido 423,74).

Iteração	10	100
Melhor Fitness	659,47	423,74
Fitness Média do Enxame	955,04	457,49

CHAPOT et al. [12] testaram o GA com população 500, taxa de crossover 80%, taxa de mutação 1% e opção de elitismo (tabela 4.3). Experimentos para a ACO [48] com heurísticas locais alcançaram o melhor resultado conhecido para este problema em 2500 iterações para 10 agentes. Sem o "*local-updating*" proposto pelos autores, o minimo encontrado por esta MHO foi *423,91*. Como escrevemos acima, o PSORK utilizado nestes experimentos não usa nenhuma heurística para o TSP devido à sua posterior aplicação à OGCIN.

Tabela 4.3 – Resultados para o GA aplicado ao TSP Oliver 30 extraídos de [12] (500 indivíduos; melhor resultado conhecido 423,74).

Generation	50	100	150	200	250	300	350	400
GA OX	-	-	-	-	-	-	-	425
GA RK	752,8	606,5	557,2	495,2	473,6	452,2	435,2	423,74
GA LM	579,5	459,9	443,3	438,3	424,4	424,4	423,74	423,74

As figuras 4.1 e 4.2 demonstram o comportamento da técnica PSORK para enxames de 50 e 500 partículas respectivamente, mostrando as melhores avalições globais (fitness) alcançadas pelo melhor em cada grupo de 10 experimentos com cada par de constantes (c, w). Nestes experimentos, usamos as constants $c_1 = c_2 = c$ e em todos os casos $w_{min} = w - 0.40$ com $t_{max} = 1000$ iterações. Para cada par (c, w) onde $c_1 = c_2 = c \in \{1.66, 1.67, ..., 1.75\}$ e $w \in \{0.66, 0.67, ..., 0.75\}$, dez experimentos foram realizados, o que resulta em 1000 experimentos.

Figura 4.1 – PSORK aplicado ao TSP Oliver 30 (50 partículas): melhores fitnesses em 10 experimentos para cada par de parâmetros (*c*, *w*).

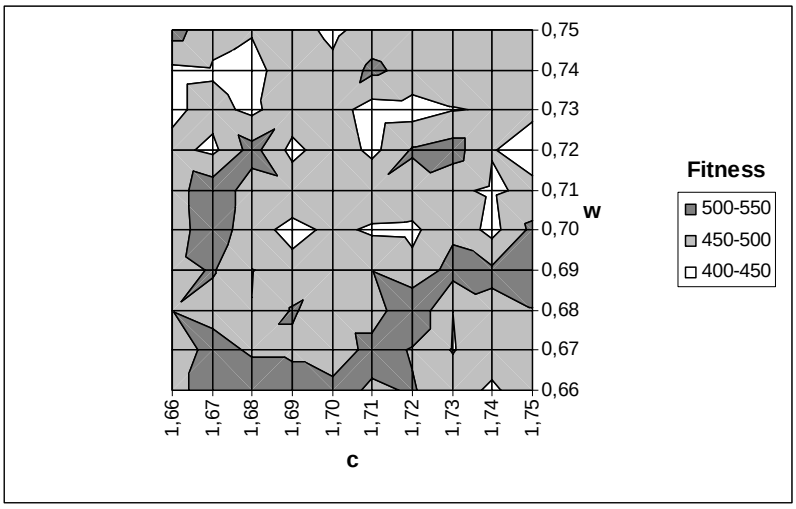
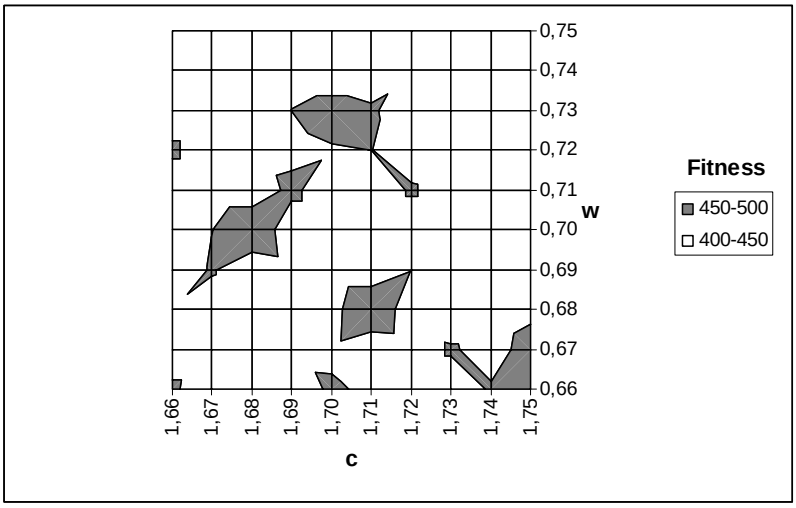


Figura 4.2 – PSORK aplicado ao TSP Oliver 30 (500 partículas): melhores fitnesses em 10 experimentos para cada par de parâmetros (*c*, *w*).



4.1.2. Particle Swarm Optimization aplicado ao TSP Rykel 48

O problema Rykel 48 é um TSP assimétrico de 48 cidades [70]. A distância relativa à melhor solução conhecida é *14422*. Realizamos três testes de análise de sensibilidade, com w = 0,7; 0,8 e 0,9 e w_{min} = 0,3 em todos os casos. Para cada teste, 10 experimentos para cada par de parâmetros (c_1, c_2), onde $c_1 \in \{1,50; 1,55; ...; 2,45\}$ e $c_2 \in \{1,50; 1,55; ...; 2,45\}$, com diferentes sementes. Os enxames tiveram 500 partículas com $t_{max} = 5000$ iterações. As figuras 4.3, 4.4 e 4.5 mostram o desempenho para cada grupo de experimentos. Os melhores resultados são demonstrados na tabela 4.4 ($w = 0,8$, $c_1 = 2,00$, $c_2 = 2,20$) e tabela 4.5 ($w = 0,8$, $c_1 = 1,60$, $c_2 = 2,40$). As figuras 4.3, 4.4 e 4.5 mostram as melhores fitnesses globais para cada grupo de 10 experimentos.

Tabela 4.4 – Resultados do melhor experimento do PSORK aplicado ao TSP Rykel 48 (500 indivíduos; $w = 0,8$, $c_1 = 2,00$, $c_2 = 2,20$).

Iteração	10	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Melhor Fitness	37975	33153	30763	28206	23057	20687	15731	14763	14422	14422	14422
Fitness Média do Enxame	50492	49758	46728	39668	33859	29464	23633	21455	19023	17548	16989

Tabela 4.5 – Resultados do melhor experimento do PSORK aplicado ao TSP Rykel 48 (500 indivíduos; $w = 0,8$, $c_1 = 1,60$, $c_2 = 2,40$; melhor resultado conhecido 14422).

Iteração	10	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Melhor Fitness	41027	34517	34517	26193	19207	15094	14751	14751	14751	14608	14608
Fitness Média do Enxame	54489	52834	51386	37471	27627	20893	16526	15625	15457	15927	15788

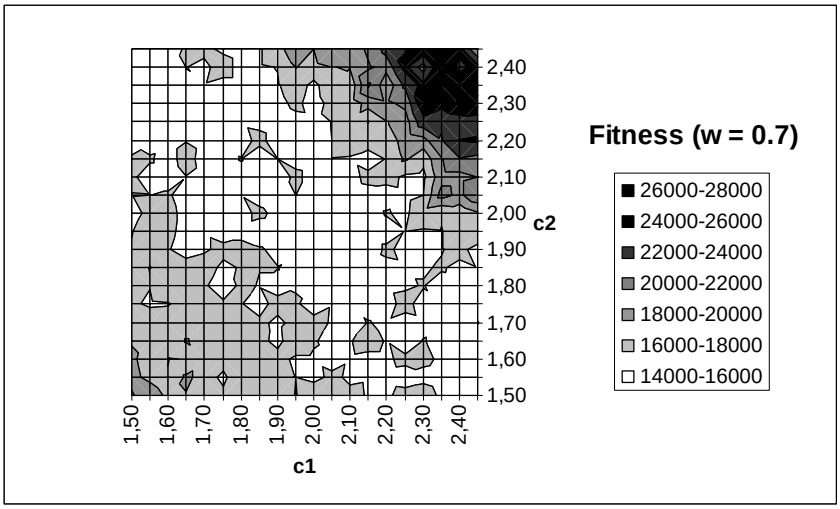


Figura 4.3 – PSORK aplicado ao TSP Rykel (500 partículas, w = 0,7): melhores fitnesses em 10 experimentos para cada par de parâmetros (c_1, c_2).

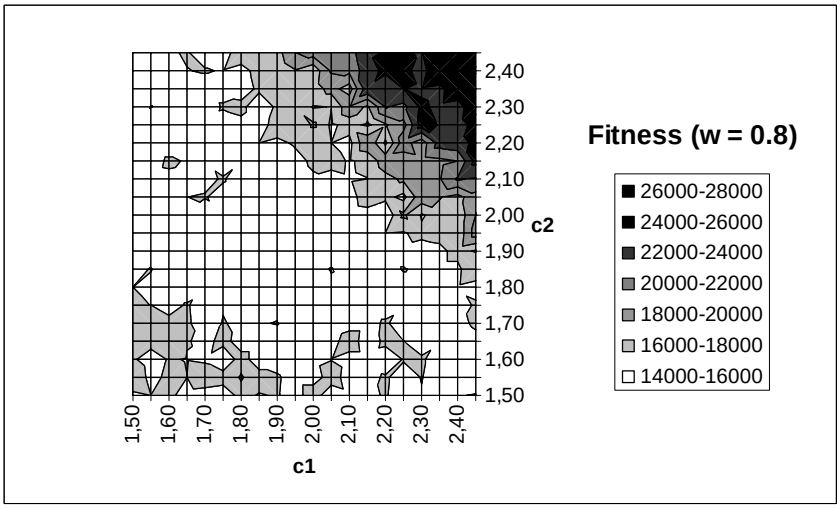


Figura 4.4 – PSORK aplicado ao TSP Rykel (500 partículas, w = 0,8): melhores fitnesses em 10 experimentos para cada par de parâmetros (c_1, c_2).

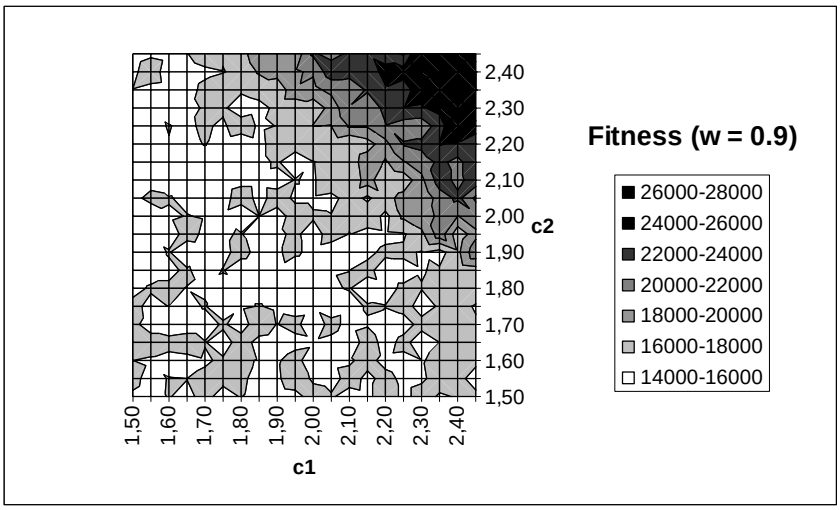


Figura 4.5 – PSORK aplicado ao TSP Rykel (500 partículas, w = 0,9): melhores fitnesses em 10 experimentos para cada par de parâmetros (c_1, c_2).

4.2. Análise dos resultados e comentários

As figuras 4.1 e 4.2 demonstram que um número maior de partículas (500 em vez de 50) levam os enxames a resultados mais homogêneos, com o mesmo número de avaliações nos experimentos com o TSP Oliver 30. Assim, o ganho em aumentar o número de partículas é maior do que buscar ajustamentos finos para os parâmetros w, c_1 e c_2. Neste caso, o PSORK alcançou o resultado 423,74 em 100 iterações (50000 avaliações) enquanto o GA leva 350 iterações (175,000 avaliações).

Grosso modo, para o TSP Rykel 48, as figuras 4.3, 4.4 e 4.5 demonstram que os melhores vôos foram aqueles em que $3{,}50 \leq c_1 + c_2 \leq 4{,}20$ para os valores $w = 0{,}7$ e $0{,}8$. Para $w = 0{,}9$ há resultados consideráveis para $c_1 + c_2 \leq 4{,}00$, embora sua região no

espaço de parâmetros não esteja tão bem definida como nos casos anteriores. Não obstante os resultados indicam convergência, robustez e homogeneidade para a extensão proposta para teste dos parâmetros, consistente com a teoria do PSO [21, 54].

Em suma, a aplicação do PSORK a problemas de referência como os TSPs Oliver30 e Rykel48, além de evidenciar a validação de código implementado, fundamental para a continuidade do trabalho de pesquisa, mostra a potencialidade desta MHO. Tratam-se de instâncias do TSP amplamente citadas na literatura, e como mencionado anteriormente, com alto grau de complexidade. Além disso, o estudo prévio com o estudo de constantes permitiu a seleção de parâmetros que supostamente poderiam dar mais robustez na OGCIN. Também é importante salientar mais uma vez que os resultados se mostraram excepcionais levando-se em conta o fato de que não foi implementada heurística local na solução de tais problemas. No próximo capítulo, serão apresentados resultados e comentários sobre o PSO aplicado à OGCIN.

Capítulo 5

Resultados da Otimização com Enxame de Partículas aplicada à Otimização do Gerenciamento de Combustível Intra-Núcleo

5.1. Resultados

Para aplicar o PSORK aplicado ao problema do mundo real da OGCIN, foram testados 15 enxames com 50 partículas com $c_1 = c_2 = 1,8$, com um w variando de $0,8$ a $0,2$ em 200 iterações. Estes parâmetros são consistentes com a análise do comportamento do PSORK aplicado aos TSPs Oliver 30 e Rykel 48, bem como com os valores encontrados na literatura. A topologia vizinhança de estrela (vizinhança *gbest*) [21] também foi utilizada neste caso. A tabela 5.1 mostra os resultados para a OGCIN.

Tabela 5.1 – PSORK aplicado à OGCIN com 50 partículas: C_B em ppm e P_{rm} em 15 experimentos (no formato C_B / P_{rm}). Resultados notáveis em comparação com GA* e PBIL** em negrito.

Experimento	80 iterações (4000 avaliações)	120 iterações (6000 avaliações)	200 iterações (10000 avaliações)
#1	**1394 / 1,384***	**1394 / 1,384****	1396 / 1,378
#2	**1200 / 1,352***	1211 / 1,388	1244 / 1,392
#3	**1067 / 1,380***	1086 / 1,349	1164 / 1,383
#4	**1039 / 1,382***	**1378 / 1,395****	1378 / 1,395
#5	**1104 / 1,385***	1130 / 1,395	1250 / 1,387
#6	**1032 / 1,385***	1054 / 1,391	1068 / 1,390
#7	**1200 / 1,374***	1204 / 1,380	1205 / 1,381
#8	**1205 / 1,382***	1205 / 1,382	1250 / 1,383
#9	**1127 / 1,379***	1192 / 1,393	1197 / 1,390
#10	**1191 / 1,389***	**1266 / 1,386****	1289 / 1,388
#11	**1235 / 1,382***	**1250 / 1,391****	1264 / 1,392
#12	**1156 / 1,359***	1165 / 1,364	1165 / 1,364
#13	**1289 / 1,392***	**1382 / 1,395****	1382 / 1,395
#14	**1129 / 1,368***	1208 / 1,325	1263 / 1,376
#15	**1159 / 1,386***	1214 / 1,395	1297 / 1,393
Média	1168 / 1,379	1233 / 1,381	1254 / 1,386
Desvio Padrão	95 / 0,011	101 / 0,020	89 / 0,008

* GA – **1026ppm / 1,390**; 4000 avaliações [12]
** PBIL – **1242ppm / 1,361**; 6000 avaliações [15]
Nota: Ant-Q com heurística local de nivelamento – **1297ppm / 1.384** e **1093ppm / 1.351**; 200 e 720 avaliações respectivamente [18]

A figura 5.1 mostra C_B versus P_{rm} para o melhor experimento computacional (#1), para cada partícula do enxame, no começo e no final da otimização. A figura 5.2 representa a evolução da C_B e a média do enxame ao longo de 200 iterações no experimento #1.

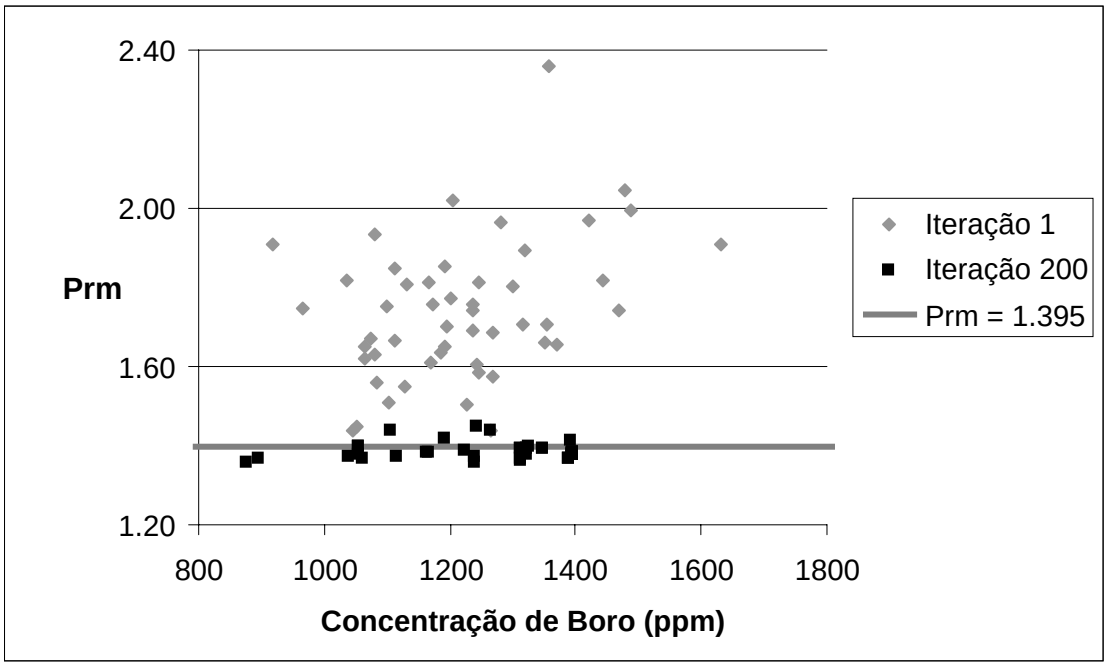


Figura 5.1 – C_B versus P_{rm} para as 50 partículas do melhor experimento para o PSORK aplicado à OGCIN, no início e no final da otimização.

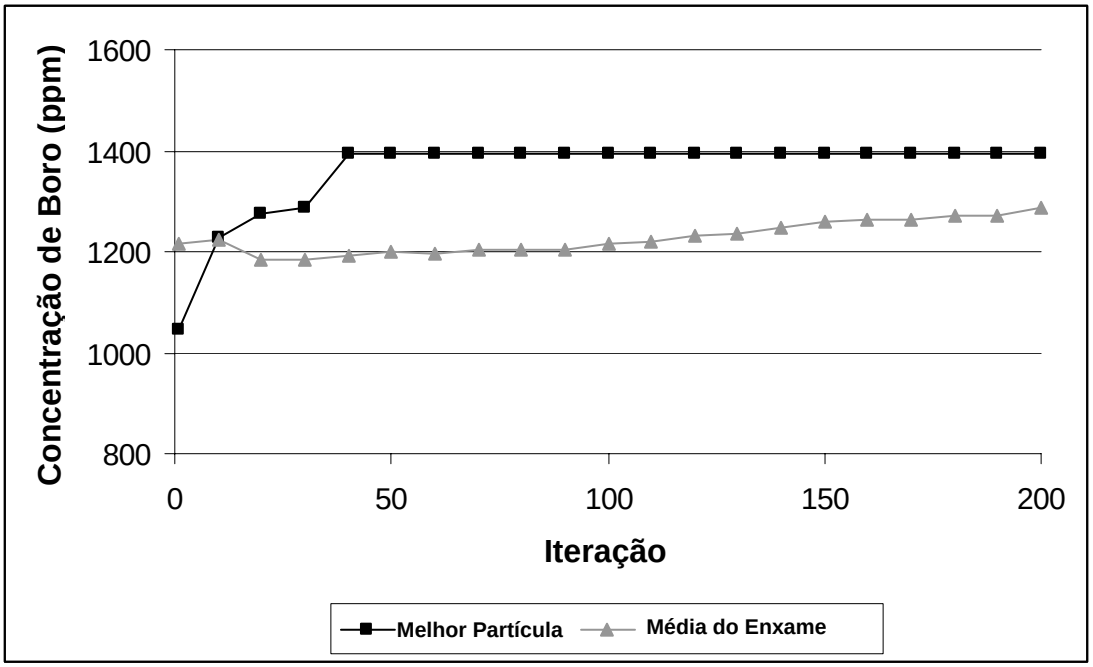


Figura 5.2 – Melhor C_B e C_B média do enxame ao longo do melhor experimento do PSORK aplicado à OGCIN (experimento #1).

5.2. Análise dos resultados e comentários

Os resultados do PSORK para a OGCIN (coluna 4000 avaliações da tabela 5.1) são comparáveis aos resultados obtidos pelo GA (C_B = 1026ppm) em 4000 avaliações [12]. Os resultados do PSORK também são comparáveis aos obtidos pelo PBIL em 6000 avaliações (C_B = 1232ppm) [15].

Pode-se inferir da figura 11 que os PCs dados pela decodificação das partículas no decorrer do processo de otimizão do PSORK tendem a configurações em concordância com os requisitos técnicos de segurança, representados pela restrição imposta pela fitness dada na eq. (2.13). No experimento #1, o enxame obteve 44 PCs na referida condição no final de 200 iterações, com a melhor C_B global de 1396ppm.

A evolução da C_B media mostrada na figura 5.2 demonstra que no começo da otimização as partículas tendem a decrescer C_B enquanto o enxame procura por valores $P_{rm} < 1,395$. Depois, o enxame tende a procurar maiores valores para C_B, aumentando o número de partículas representando o número de LPs válidos, com um também aumento da C_B média ao longo da otimização.

Considerando que os enxames não possuem informações prévias a respeito do problema e que nenhuma heurística foi utilizada, é possível afirmar que os resultados obtidos ratificam que o paradigma da Inteligência de Enxame pode ser usado na OGCIN com considerável desempenho. No capítulo seguinte apresentaremos os resultados da HAVR utilizada em conjunto com os algoritmos de RS e PSORK para aplicação à OGCIN.

Capítulo 6

Resultados da Heurística de Aceitação de Vizinhança Reativa aplicada à Otimização do Gerenciamento de Combustível Intra-Núcleo

Neste capítulo apresentamos os resultados da HAVR aplicada à OGCIN juntamente com a RS e o PSO. A HAVR, como descrito no capítulo 3, é uma heurística que, baseada em conhecimento de engenharia nuclear, permite a seleção de PCs a serem avaliados pelos algoritmos de busca. E algo muito interessante por ela proporcionado, como será visto adiante, é que a avaliação de PCs selecionados é suficiente para que o algoritmo tenha um desempenho considerável em um número de iterações menor, reduzindo o custo computacional da busca.

6.1. HAVR e RS aplicadas à OGCIN

6.1.1. Resultados

Os resultados para a RS com HAVR estão nas tabelas 6.1, 6.2 e 6.3, com máximo grau de tolerância 1, 2, e 3, respectivamente. Um conjunto de 20 experimentos com sementes diferentes foi executado para cada grau máximo de tolerância, resultando em 60 experimentos. Para uma comparação clara com trabalhos anteriores [12, 29], os

dados são dados para o número de soluções geradas 4000, 6000 and 10000. Cada célula das referidas tabelas contêm C_B e P_{rm} e o número real de avaliações que ocorreram até o valor ser determinado.

A figura 6.1 mostra os parâmetro nucleares dos 50 primeiros PCs avaliados, bem como os 50 melhores PCs no final da busca para o melhor experimento do algoritmo RS-HAVR, com máximo grau 3 (tabela 6.3, #18), bem como o limite de segurança ($P_{rm} = 1.395$).

Tabela 6.1 – Resultados de 20 experimentos para RS-HAVR aplicado à OGCIN com máximo grau de tolerância 1 no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experimento	4000 soluções geradas	6000 soluções geradas	10000 soluções geradas
#1	1188 / 1,425 [285]	**899 / 1,370 [424]**	**899 / 1,370 [666]**
#2	891 / 1,463 [267]	1179 / 1,438 [391]	934 / 1,424 [665]
#3	989 / 1,410 [290]	989 / 1,410 [444]	989 / 1,410 [708]
#4	1178 / 1,481 [311]	**926 / 1,375 [449]**	**926 / 1,375 [732]**
#5	**922 / 1,333 [287]**	**922 / 1,333 [433]**	**922 / 1,333 [730]**
#6	1132 / 1,430 [305]	1132 / 1,430 [438]	981 / 1,423 [735]
#7	896 / 1,433 [274]	1074 / 1,432 [401]	1113 / 1,416 [695]
#8	897 / 1,426 [299]	897 / 1,426 [430]	**917 / 1,372 [717]**
#9	927 / 1,404 [274]	927 / 1,404 [425]	927 / 1,404 [695]
#10	**936 / 1,379 [309]**	**936 / 1,379 [456]**	**936 / 1,379 [759]**
#11	**931 / 1,388 [277]**	**931 / 1,388 [429]**	**931 / 1,388 [726]**
#12	**906 / 1,327 [280]**	**906 / 1,327 [406]**	**906 / 1,327 [734]**
#13	**890 / 1,339 [292]**	**890 / 1,339 [438]**	**890 / 1,339 [746]**
#14	1026 / 1,432 [302]	1026 / 1,432 [440]	**957 / 1,362 [711]**
#15	1071 / 1,407 [300]	1057 / 1,396 [438]	1057 / 1,396 [707]
#16	934 / 1,481 [267]	**1058 / 1,394 [419]**	**1058 / 1,394 [732]**
#17	**940 / 1,377 [289]**	**940 / 1,377 [433]**	**940 / 1,377 [707]**
#18	**1174 / 1,394 [271]**	**1174 / 1,394 [415]**	**1174 / 1,394 [713]**
#19	1178 / 1,416 [263]	1106 / 1,409 [419]	1106 / 1,409 [667]
#20	**956 / 1,395 [284]**	**956 / 1,395 [440]**	**956 / 1,395 [740]**
Média	998 / 1,407 [286]	996 / 1,392 [428]	976 / 1,384 [714]
Desvio-Padrão	112 / 0,043 [15]	96 / 0,033 [16]	81 / 0,028 [27]

Tabela 6.2 – Resultados de 20 experimentos para RS-HAVR aplicado à OGCIN com máximo grau de tolerância 2 no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experimento	4000 soluções geradas	6000 soluções geradas	10000 soluções geradas
#1	**900 / 1,383 [816]**	**900 / 1,383 [1219]**	**900 / 1,383 [1941]**
#2	**1103 / 1,322 [784]**	**1103 / 1,322 [1179]**	**1103 / 1,322 [1978]**
#3	989 / 1,410 [804]	**960 / 1,387 [1222]**	**960 / 1,387 [2015]**
#4	1271 / 1,475 [795]	**922 / 1,372 [1183]**	**922 / 1,372 [1989]**
#5	**922 / 1,333 [779]**	**922 / 1,333 [1186]**	**922 / 1,333 [1990]**
#6	943 / 1,425 [808]	**1026 / 1,355 [1180]**	**1026 / 1,355 [1998]**
#7	**854 / 1,376 [793]**	**854 / 1,376 [1198]**	**854 / 1,376 [1998]**
#8	897 / 1,426 [797]	897 / 1,426 [1214]	**917 / 1,372 [2013]**
#9	927 / 1,404 [727]	927 / 1,404 [1137]	927 / 1,404 [1955]
#10	**936 / 1,379 [817]**	**936 / 1,379 [1217]**	**936 / 1,379 [2021]**
#11	**931 / 1,388 [787]**	**931 / 1,388 [1217]**	**931 / 1,388 [2019]**
#12	**906 / 1,327 [822]**	**906 / 1,327 [1193]**	**906 / 1,327 [2027]**
#13	**890 / 1,339 [795]**	**890 / 1,339 [1191]**	**890 / 1,339 [2009]**
#14	**1165 / 1,387 [834]**	**1165 / 1,387 [1209]**	**1165 / 1,387 [1969]**
#15	1071 / 1,407 [826]	1057 / 1,396 [1232]	1057 / 1,396 [2035]
#16	988 / 1,415 [737]	**1058 / 1,394 [1135]**	**1058 / 1,394 [1928]**
#17	**940 / 1,377 [780]**	**940 / 1,377 [1179]**	**940 / 1,377 [1942]**
#18	**1174 / 1,394 [814]**	**1174 / 1,394 [1204]**	**1174 / 1,394 [2021]**
#19	1178 / 1,416 [785]	1106 / 1,409 [1181]	**1033 / 1,372 [1924]**
#20	**956 / 1,395 [798]**	**956 / 1,395 [1187]**	**956 / 1,395 [2028]**
Média	997 / 1,389 [795]	982 / 1,377 [1193]	979 / 1,373 [1990]
Desvio-Padrão	119 / 0,038 [27]	96 / 0,028 [26]	92 / 0,025 [36]

Tabela 6.3 – Resultados de 20 experimentos para RS-HAVR aplicado à OGCIN com máximo grau de tolerância 3 no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experimento	4000 soluções geradas	6000 soluções geradas	10000 soluções geradas
#1	**900 / 1,383 [1567]**	**900 / 1,383 [2328]**	**900 / 1,383 [3826]**
#2	**1103 / 1,322 [1588]**	**1103 / 1,322 [2355]**	**1103 / 1,322 [3931]**
#3	989 / 1,410 [1585]	**960 / 1,387 [2367]**	**960 / 1,387 [3909]**
#4	1099 / 1,419 [1551]	**922 / 1,372 [2306]**	**922 / 1,372 [3883]**
#5	**922 / 1,333 [1506]**	**922 / 1,333 [2269]**	**922 / 1,333 [3883]**
#6	1130 / 1,411 [1593]	**1026 / 1,355 [2336]**	**1026 / 1,355 [3921]**
#7	**854 / 1,376 [1529]**	**854 / 1,376 [2333]**	**854 / 1,376 [3909]**
#8	897 / 1,426 [1587]	897 / 1,426 [2394]	**917 / 1,372 [3945]**
#9	**1099 / 1,325 [1515]**	**1099 / 1,325 [2299]**	**1099 / 1,325 [3869]**
#10	**936 / 1,379 [1552]**	**936 / 1,379 [2319]**	**936 / 1,379 [3906]**
#11	**931 / 1,338 [1566]**	**931 / 1,388 [2355]**	**931 / 1,388 [3918]**
#12	**906 / 1,327 [1613]**	**906 / 1,327 [2371]**	**906 / 1,327 [4008]**
#13	**890 / 1,339 [1520]**	**890 / 1,339 [2293]**	**890 / 1,339 [3862]**
#14	**1165 / 1,387 [1578]**	**1165 / 1,387 [2337]**	**1165 / 1,387 [3868]**
#15	1071 / 1,407 [1595]	1057 / 1,396 [2372]	1057 / 1,396 [3934]
#16	1130 / 1,405 [1533]	**1058 / 1,394 [2316]**	**1058 / 1,394 [3923]**
#17	**940 / 1,377 [1527]**	**940 / 1,377 [2301]**	**940 / 1,377 [3845]**
#18	**1174 / 1,394 [1557]**	**1174 / 1,394 [2333]**	**1174 / 1,394 [3876]**
#19	1178 / 1,416 [1551]	1106 / 1,409 [2342]	**1033 / 1,372 [3841]**
#20	**956 / 1,395 [1558]**	**956 / 1,395 [2331]**	**956 / 1,395 [3922]**
Média	1014 / 1,381 [1559]	990 / 1,373 [2334]	987 / 1,369 [3896]
Desvio-Padrão	112 / 0,034 [30]	99 / 0,030 [28]	95 / 0,026 [45]

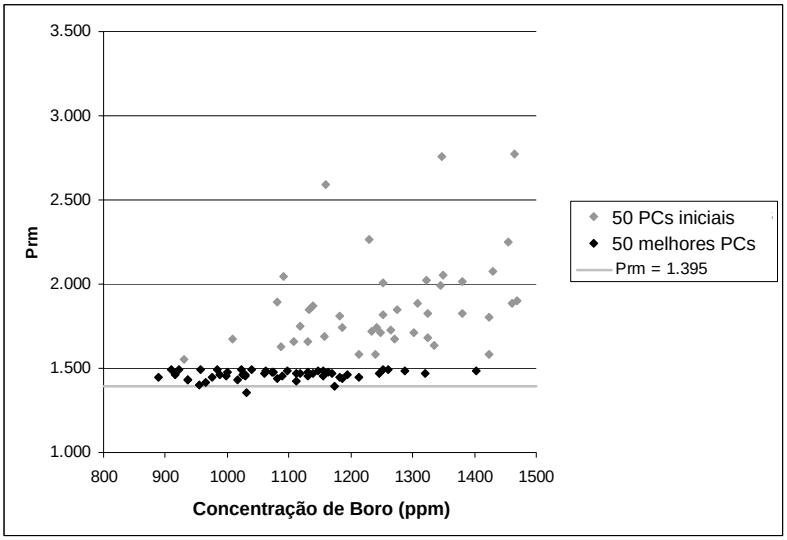


Figura 6.1 – Resultados para os 50 PCs avaliados e os 50 melhores PCs no final do processo de otimização (RS-HAVR, experimento #18, máximo grau 3).

6.1.2. Comentários

Analisando as tabelas 6.1, 6.2 e 6.3, pode-se inferir que a quantidade de soluções válidas com respeito a segurança (P_{rm} = *1.395*) aumenta, apresentando, grosso modo, uma correlação com o máximo grau de aceitação. É possível observar que, para os casos dos experimentos realizados, quanto maior o máximo grau, maior é o número de soluções válidas avaliadas, e portanto ocorre o aumento da qualidade dos resultados a longo prazo, durante a otimização. Isto reforça a idéia de que não é necessário avaliar exaustivamente todas as soluções geradas pela MHO já que tem-se conhecimento em Engenharia Nuclear suficiente para ser usado a fim de eliminar PCs que não valem a pena serem avaliados, mas de uma maneira relativa.

Uma outra importante evidência é que, para a RS-HAVR com grau máximo 3 (tabela 6.3), o melhor experimento com C_B = *1174ppm* em 3876 avaliações e o valor C_B = *987ppm* em 3896 avaliações (médias) são comparáveis aos valores obtidos pelo GA (C_B = *1026 ppm* [12]) e PSORK (C_B = *1168ppm* em média [23]) em 4000 avaliações, sem heurísticas.

Para demonstrar o potencial desta abordagem, é importante ressaltar que, para o máximo grau 1 (tabela 6.1), diversos resultados válidos foram obtidos com somente 286, 428 e 714 avaliações (em média).

Além disto, um fato interessante que demonstra a consistência desta abordagem é que para um máximo grau 1 (tabela 6.1), em 4000, 6000 e 10000 soluções geradas existem em média 286, 428 e 714 PCs avaliados respectivamente, o que corresponde a aproximadamente a 7,15%, 7,13%, e 7,14%. Neste sentido, a proporção de PCs avaliados em média ao longo das iterações é aproximadamente igual. De maneira similar, para um grau máximo 2 (tabela 6.2) com as médias de PCs avaliados 1559, 2334 and 3896, as porcentagens são aproximadamente 19,88%, 19,88% e 19,90%, respectivamente. Para um máximo grau 3 (tabela 6.3), as porcentagens relacionadas com as médias são aproximadamente 38,98%, 38,90% e 38,96%.

Da figura 6.1 pode-se inferir que os PCs ao final da otimização são de fato melhores que os primeiros 50 avaliados. Não obstante, neste experimento (#18) apenas dois PCs foram válidos em termos de segurança, com $P_{rm} \leq 1.395$.

6.2. HAVR e PSORK aplicados à OGCIN

6.2.1. Resultados

Resultados para o PSORK com HAVR nas tabelas 6.4, 6.5 e 6.6, foram obtidos com os graus máximos de tolerância 1, 2, e 3, respectivamente. Como para o RS-HAVR, um conjunto de 20 experimentos com sementes diferentes foi executado em cada caso (máximo grau de tolerância 1, 2 e 3), resultando em 60 execuções. Os dados das tabelas são dados os números de soluções gerados 4000, 6000 e 10000. Os enxames tinham 50 partículas com $c_1 = c_2 = 1,8$, e w variando de $0,8$ a $0,2$ em 200 iterações.

As figuras 6.2, 6.3 e 6.4 mostram os PCs iniciais avaliados pelo código de Física de Reatores e a melhor solução para cada particular ao final do processo de otimização para cada partícula para os melhores experimentos PSO-HAVR em termos de Concentração de Boro: #5 para máximo grau 1, #9 para máximo grau 2, e #17 para máximo grau 3.

Um outro tipo de experimento com PSORK-HAVR foi realizado com o grau máximo variando ao longo das iteraçõs. Neste caso, 20 experimentos foram realizados, mas o número total de iteraçõs foi reduzido para 100. Nestes experimentos, o grau máximo 3 foi utilizado até a 30ª. iteração do algoritmo. Depois, o grau máximo 2 foi utilizado até a 60ª. iteração. Depois, o grau máximo 1 foi utilizado até o fim da otimização. Os resultados podem ser vistos na tabela 6.7.

A figura 6.5 mostra os resultados do melhor experimento (#9), com os parâmetros nucleares avaliados na primeira iteração e os melhores parâmetros nucleares de cada partícula na 100ª. iteração (fim do processo de otimização para este caso).

Tabela 6.4 – Resultados de 20 experimentos para PSORK-HAVR aplicado à OGCIN com máximo grau de tolerância 1 no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experimento	4000 soluções geradas	6000 soluções geradas	10000 soluções geradas
#1	**1028 / 1,379 [1138]**	**1143 / 1,370 [1890]**	**1143 / 1,370 [4161]**
#2	1060 / 1,405 [865]	**1090 / 1,391 [1587]**	**1112 / 1,394 [2891]**
#3	**976 / 1,357 [1198]**	**984 / 1,382 [2302]**	**1026 / 1,371 [4135]**
#4	**1161 / 1,382 [1475]**	**1173 / 1,385 [2483]**	**1173 / 1,385 [5439]**
#5	**939 / 1,286 [470]**	**1124 / 1,370 [861]**	**1221 / 1,390 [2288]**
#6	**1163 / 1,315 [1303]**	**1163 / 1,315 [2478]**	**1208 / 1,388 [5304]**
#7	**1128 / 1,378 [1306]**	**1138 / 1,393 [2466]**	**1138 / 1,393 [5531]**
#8	**1146 / 1,388 [1044]**	**1202 / 1,377 [1963]**	**1202 / 1,377 [4490]**
#9	**984 / 1,389 [1228]**	**1097 / 1,383 [2224]**	**1119 / 1,374 [4490]**
#10	**1207 / 1,390 [477]**	**1207 / 1,390 [906]**	**1207 / 1,390 [2390]**
#11	**1041 / 1,387 [969]**	**1095 / 1,393 [1901]**	**1095 / 1,393 [4817]**
#12	**939 / 1,380 [1172]**	**1165 / 1,354 [2020]**	**1176 / 1,385 [3698]**
#13	**968 / 1,389 [1209]**	**986 / 1,373 [2022]**	**1039 / 1,386 [4134]**
#14	**954 / 1,349 [1226]**	**966 / 1,388 [2270]**	**977 / 1,359 [5376]**
#15	**1033 / 1,393 [907]**	**1145 / 1,390 [1487]**	**1191 / 1,386 [2930]**
#16	**1130 / 1,393 [1014]**	**1182 / 1,346 [1941]**	**1182 / 1,346 [4673]**
#17	**1087 / 1,372 [1281]**	**1118 / 1,360 [2288]**	**1137 / 1,322 [5013]**
#18	**1080 / 1,388 [968]**	**1163 / 1,392 [1830]**	**1163 / 1,392 [4404]**
#19	**977 / 1,380 [1248]**	**990 / 1,395 [2239]**	**990 / 1,395 [4279]**
#20	**1104 / 1,347 [1050]**	**1117 / 1,330 [1742]**	**1117 / 1,330 [3491]**
Média	1055 / 1,372 [1077]	1112 / 1,374 [1945]	1131 / 1,376 [4197]
Desvio-Padrão	84 / 0,029 [257]	75 / 0,022 [460]	73 / 0,021 [984]

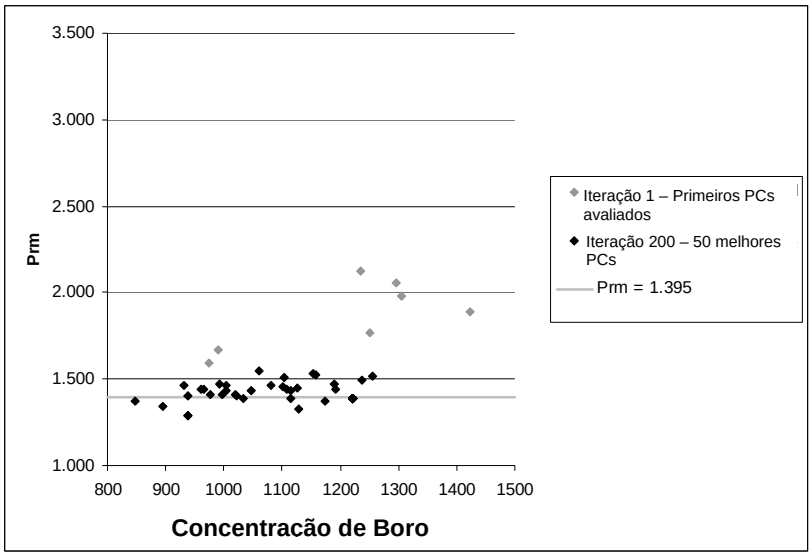


Figura 6.2 – Resultados para os primeiros PCs avaliados e os 50 melhores PCs no final do processo de otimização (PSORK-HAVR, experimento #5, máximo grau 1).

Tabela 6.5 – Resultados de 20 experimentos para PSORK-HAVR aplicado à OGCIN com máximo grau de tolerância 2 no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experiment	4000 soluções geradas	6000 soluções geradas	10000 soluções geradas
#1	**1181 / 1,386 [1404]**	**1181 / 1,386 [2259]**	**1232 / 1,389 [4123]**
#2	**1116 / 1,360 [1583]**	**1119 / 1,381 [2411]**	**1141 / 1,376 [4901]**
#3	**1247 / 1,381 [1628]**	**1250 / 1,391 [2764]**	**1250 / 1,391 [5424]**
#4	**1034 / 1,372 [1682]**	**1048 / 1,389 [2481]**	**1048 / 1,389 [4257]**
#5	**919 / 1,373 [1365]**	**1125 / 1,381 [2421]**	**1167 / 1,314 [4567]**
#6	**1220 / 1,386 [2117]**	**1220 / 1,386 [3137]**	**1220 / 1,386 [5639]**
#7	**1236 / 1,394 [2240]**	**1237 / 1,395 [3744]**	**1237 / 1,395 [7076]**
#8	**1185 / 1,375 [1606]**	**1185 / 1,375 [2379]**	**1185 / 1,375 [4626]**
#9	**1089 / 1,375 [1903]**	**1231 / 1,370 [2748]**	**1325 / 1,379 [4806]**
#10	**1169 / 1,370 [2455]**	**1169 / 1,370 [4099]**	**1172 / 1,378 [7593]**
#11	**1165 / 1,391 [2241]**	**1251 / 1,390 [3610]**	**1251 / 1,390 [6264]**
#12	**1152 / 1,395 [2648]**	**1179 / 1,383 [3946]**	**1280 / 1,395 [6991]**
#13	**1123 / 1,371 [1679]**	**1184 / 1,386 [2725]**	**1213 / 1,360 [5351]**
#14	**1109 / 1,365 [2072]**	**1117 / 1,367 [3320]**	**1119 / 1,386 [6379]**
#15	**1115 / 1,378 [2327]**	**1116 / 1,379 [3713]**	**1123 / 1,367 [6742]**
#16	**1122 / 1,380 [1240]**	**1122 / 1,380 [2170]**	**1122 / 1,380 [4411]**
#17	**1185 / 1,369 [1932]**	**1186 / 1,374 [3336]**	**1186 / 1,374 [6660]**
#18	**1194 / 1,368 [1979]**	**1199 / 1,358 [3323]**	**1220 / 1,387 [6669]**
#19	**899 / 1,337 [2061]**	**1161 / 1,383 [3454]**	**1182 / 1,393 [6330]**
#20	**1052 / 1,394 [2803]**	**1052 / 1,394 [4259]**	**1068 / 1,380 [7301]**
Média	1126 / 1,376 [1948]	1167 / 1,381 [3115]	1183 / 1,379 [5087]
Desvio-Padrão	93 / 0,014 [428]	59 / 0,010 [656]	66 / 0,018 [1123]

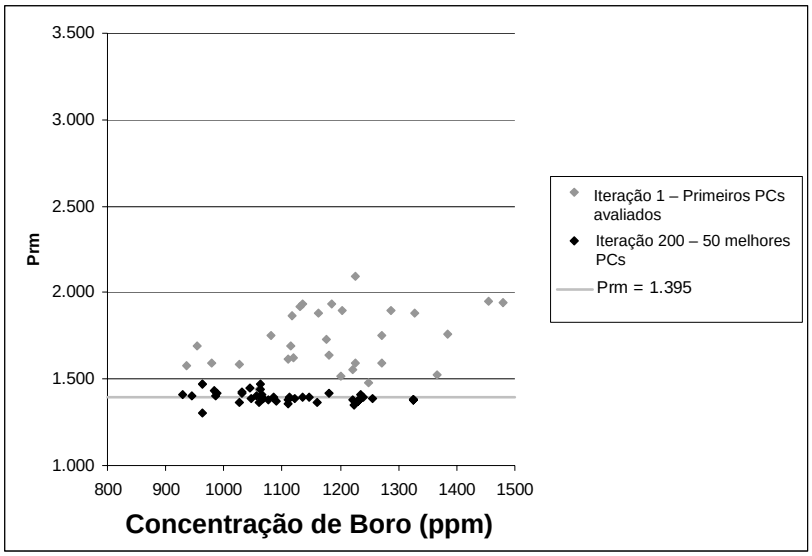


Figura 6.3 – Resultados para os primeiros PCs avaliados e os 50 melhores PCs no final do processo de otimização (PSORK-HAVR, experimento #9, máximo grau 2).

Tabela 6.6 – Resultados de 20 experimentos para PSORK-HAVR aplicado à OGCIN com máximo grau de tolerância 3 no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experimento	4000 soluções geradas	6000 soluções geradas	10000 soluções geradas
#1	**1175 / 1,392 [2404]**	**1314 / 1,395 [3669]**	**1317 / 1,388 [6462]**
#2	**1165 / 1,389 [2996]**	**1184 / 1,371 [4633]**	**1192 / 1,363 [8173]**
#3	**1227 / 1,388 [3158]**	**1227 / 1,388 [4912]**	**1227 / 1,388 [8523]**
#4	**1163 / 1,379 [3032]**	**1163 / 1,379 [4852]**	**1163 / 1,379 [8693]**
#5	**1200 / 1,385 [2789]**	**1200 / 1,385 [4394]**	**1217 / 1,394 [7776]**
#6	**1189 / 1,349 [3037]**	**1218 / 1,390 [4719]**	**1226 / 1,394 [8396]**
#7	**941 / 1,389 [3119]**	**995 / 1,393 [4817]**	**1177 / 1,392 [8385]**
#8	**1138 / 1,386 [1962]**	**1139 / 1,386 [2939]**	**1139 / 1,386 [5434]**
#9	**1203 / 1,393 [2937]**	**1211 / 1,389 [4559]**	**1213 / 1,383 [8014]**
#10	**914 / 1,369 [2785]**	**1051 / 1,344 [4445]**	**1089 / 1,377 [7797]**
#11	**1160 / 1,386 [2396]**	**1250 / 1,388 [3800]**	**1250 / 1,388 [6536]**
#12	**1174 / 1,392 [3427]**	**1174 / 1,392 [5230]**	**1214 / 1,386 [9078]**
#13	**1120 / 1,368 [2307]**	**1332 / 1,390 [3720]**	**1340 / 1,395 [6911]**
#14	**1171 / 1,345 [2182]**	**1267 / 1,374 [3583]**	**1267 / 1,374 [6665]**
#15	**1142 / 1,349 [3075]**	**1167 / 1,350 [4738]**	**1193 / 1,384 [8337]**
#16	**1122 / 1,380 [2115]**	**1122 / 1,380 [3473]**	**1122 / 1,380 [6511]**
#17	**1335 / 1,390 [2812]**	**1342 / 1,372 [4188]**	**1342 / 1,372 [7160]**
#18	**1243 / 1,368 [2179]**	**1243 / 1,368 [3527]**	**1243 / 1,368 [6655]**
#19	**1158 / 1,390 [2828]**	**1158 / 1,390 [4389]**	**1205 / 1,391 [7740]**
#20	**1184 / 1,317 [3120]**	**1184 / 1,317 [4662]**	**1185 / 1,315 [7513]**
Média	1157 / 1,375 [2733]	1197 / 1,377 [4262]	1216 / 1,380 [7583]
Desvio-Padrão	92 / 0,021[422]	86 / 0,020 [613]	66 / 0,018 [954]

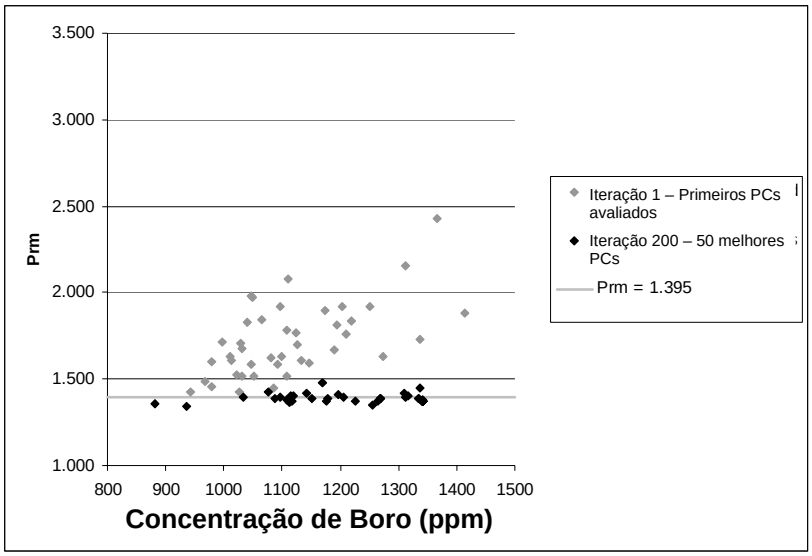


Figura 6.4 – Resultados para os primeiros PCs avaliados e os 50 melhores PCs no final do processo de otimização (PSORK-HAVR, experimento #17, máximo grau 3).

Tabela 6.7 – Resultados de 20 experimentos para PSORK-HAVR aplicado à OGCIN com máximo grau de tolerância variável (3 até a iteração 30, depois 2 até a iteração 60, e então 1 até a iteração 100) no formato C_B / P_{rm} [*número de avaliações*]. Resultados dentro do limite de segurança em negrito.

Experimento	2000 soluções geradas	3000 soluções geradas	5000 soluções geradas
#1	**1199 / 1,381 [976]**	**1199 / 1,381 [1370]**	**1199 / 1,381 [1646]**
#2	**1171 / 1,375 [1089]**	**1171 / 1,375 [1414]**	**1171 / 1,375 [1634]**
#3	**1200 / 1,368 [965]**	**1200 / 1,368 [1455]**	**1200 / 1,368 [2146]**
#4	941 / 1,407 [1135]	**1057 / 1,380 [1599]**	**1198 / 1,383 [2088]**
#5	**1061 / 1,377 [1594]**	**1133 / 1,373 [2199]**	**1178 / 1,387 [2978]**
#6	**1103 / 1,381 [1382]**	**1148 / 1,387 [1921]**	**1148 / 1,387 [2310]**
#7	**1087 / 1,390 [927]**	**1101 / 1,335 [1329]**	**1101 / 1,335 [1719]**
#8	913 / 1,424 [1423]	**924 / 1,393 [1964]**	**924 / 1,393 [2564]**
#9	**936 / 1,330 [1222]**	**1000 / 1,394 [1711]**	**1244 / 1,360 [2145]**
#10	**1119 / 1,384 [1290]**	**1119 / 1,384 [1878]**	**1213 / 1,394 [2765]**
#11	**1165 / 1,328 [1433]**	**1169 / 1,385 [2106]**	**1170 / 1,384 [3227]**
#12	**1099 / 1,385 [1416]**	**1111 / 1,382 [1995]**	**1111 / 1,382 [2368]**
#13	**1189 / 1,389 [1617]**	**1210 / 1,372 [2284]**	**1210 / 1,372 [3113]**
#14	**1051 / 1,392 [1251]**	**1144 / 1,375 [1767]**	**1176 / 1,388 [2375]**
#15	926 / 1,401 [1521]	**1066 / 1,310 [2209]**	**1126 / 1,376 [3392]**
#16	**964 / 1,392 [1186]**	**964 / 1,392 [1796]**	**1002 / 1,379 [2570]**
#17	**944 / 1,397 [1551]**	**915 / 1,389 [2350]**	**1024 / 1,356 [3545]**
#18	**1092 / 1,332 [1095]**	**1092 / 1,332 [1351]**	**1092 / 1,332 [1479]**
#19	**1019 / 1,391 [1111]**	**1081 / 1,392 [1653]**	**1081 / 1,392 [1971]**
#20	**1129 / 1,384 [1534]**	**1129 / 1,384 [2147]**	**1196 / 1,387 [2822]**
Média	1065 / 1,380 [1286]	1097 / 1,374 [1825]	1138 / 1,376 [2443]
Desvio-Padrão	99 / 0,025 [221]	87/ 0,023 [332]	82 / 0,018 [609]

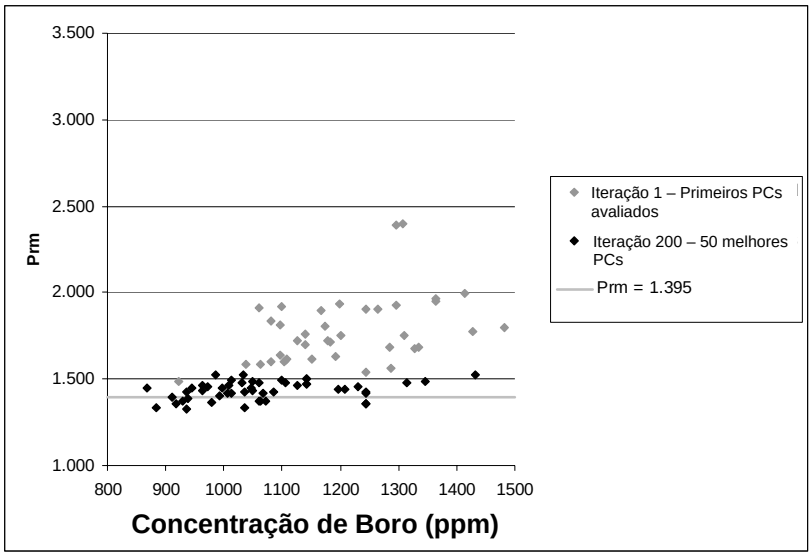


Figura 6.5 – Resultados para os primeiros PCs avaliados e os 50 melhores PCs no final do processo de otimização (PSORK-HAVR, experimento #9, máximo grau variável de 3 a 1 em 100 iterações).

A tabela 6.8 mostra as médias das experiências realizadas com a HAVR, para efeitos comparativos.

Tabela 6.8 – Comparação entre os resultados da RS e do PSO com HAVR (C_B/P_{rm} [*número de avaliações*] {*total de experimentos com soluções válidas*}).

Experimento	**4000 soluções geradas**	**6000 soluções geradas**	**10000 soluções geradas**
RS-HAVR (Máx. Grau 1)	998 / 1,407 [286]{08}	996 / 1,392 [428]{11}	976 / 1,384 [714] {13}
RS-HAVR (Máx. Grau 2)	997 / 1,389 [795]{12}	982 / 1,377 [1193] {16}	979 / 1,373 [1990] {18}
RS-HAVR (Máx. Grau 3)	1014 / 1,381 [1559] {13}	990 / 1,373 [2334] {17}	987 / 1,369 [3896] {19}
PSO-HAVR (Máx. Grau 1)	1055 / 1,372 [1007] {19}	1112 / 1,374 [1945] {20}	1131 / 1,376 [4197] {20}
PSO-HAVR (Máx. Grau 2)	1126 / 1,376 [1948] {20}	1167 / 1,381 [3115] {20}	1183 / 1,379 [5087] {20}
PSO-HAVR (Máx. Grau 3)	1157 / 1,375 [2733]{20}	1197 / 1,377 [4262] {20}	1216 / 1,380 [7583] {20}

6.2.2. Comentários

O número de PCs avaliados aumenta consideravelmente quando comparamos RS-HAVR e PSO-HAVR em longo prazo. Por exemplo, com máximo grau 1, o número de avaliações em media é 714 para o RS-HAVR e 4197 para o PSO-HAVR em 10000 soluções geradas. De uma maneira similar, para o máximo grau 2, em média, em 10000 soluções geradas, RS-HAVR apresenta 1990 avaliações e PSO-HAVR apresenta 5087. Para o máximo grau 3, em média, RS-HAVR apresenta 3896 avaliações e PSO-RNAH apresenta 7583 avaliações. Isto é consistente com o fato de que o PSO direciona as buscas para PCs que, de acordo com os critérios da HAVR, valem ser avaliados.

Para o máximo grau 1 (tabela 6.4) é notável o fato de que 19 dos 20 experimentos forneceram resultados válidos em termos de segurança para 4000 soluções geradas com uma média de $C_B = 1055ppm$ e um número médio de avaliações 1077. Comparando ao GA (*1026ppm* em 4000 avaliações [12]), 12 dos 20 experimentos apresentam resultados melhores. Assim, é possível achar diversos PCs comparáveis e/ou melhores que o melhor resultado do GA em aproximadamente um quarto das avaliações.

Com um máximo grau 2 (tabela 6.5) foi possível alcançar a média $C_B = 1167ppm$ em 3115 avaliações (para 6000 soluções geradas). PSORK sem heurística [23] alcança em média $C_B = 1168ppm$ em 4000 avaliações. O número de avaliações com a HAVR foi reduzido em quase três quartos. As figuras 6.2, 6.3 e 6.4 mostram comparações entre os parâmetros nucleares dos PCs avaliados na primeira iteração e o melhor de cada partícula no final do processo de otimização para cada melhor experimento de cada um dos máximos graus 1, 2 e 3. É possível inferir que o número de avaliações na iteração inicial (pontos cinzas) aumenta de acordo com o grau máximo em cada um dos casos apresentados. Além disto, também é possível ver que os parâmetros nucleares para as melhores partículas no final da otimização (pontos pretos) possuem uma melhor qualidade (ambos menor P_{rm} e maior C_B) quanto maior for o máximo grau.

Embora, em comparação com o melhor experimento para o PSORK, existissem 44 PCs válidos em termos de segurança ($P_{rm} \leq 1{,}395$) (como mostrado na figura 5.1), para estes melhores experimentos o número de PCs válidos no final da otimização foram 25, 33 e 39, para os graus máximos 1, 2, e 3 respectivamente. De maneira óbvia, se o número de avaliações é reduzido, várias partículas não serão capazes de melhorar a qualidade de suas soluções ao longo do processo de otimização. No entanto, como demostram os resultados, a HAVR não prejudica a qualidade da melhor solução global, que é alcançado em um número de avaliações consideravelmente menor.

Os resultados dos experimentos da tabela 6.7 foram obtidos, como mencionado anteriormente, de uma maneira tal que o máximo grau é reduzido de uma maneira diferente: o algoritmo começa com um grau máximo 3, reduzindo posteriormente para 2 e finalmente para 1. Isso leva a um interessante resultado, com uma média de $C_B = 1138ppm$ em um número médio de avaliações 2443, que é menos de três quartos das 4000 avaliações realizadas pelo GA ($C_B = 1026ppm$ [12]) e PSORK ($C_B = 1168ppm$ em média – tabela 5.1), com a geração de somente 5000 soluções a longo prazo, metade das soluções geradas quando apenas um grau máximo específico é usado ao longo de todas as iterações, como nos experimentos cujos resultados estão nas tabelas 6.1, 6.2 and 6.3. A figura 6.5 mostra os parâmetros de saída para o melhor experimento nestas condições (#9). Neste caso, na iteração 100 existem 14 LPs possíveis em termos de segurança, já que o número de soluções geradas também é reduzido, embora isto não prejudique a qualidade da melhor solução como nos casos anteriores.

O próximo capítulo apresentará e discutirá os resultados da BBC aplicada à OGCIN.

Capítulo 7

Resultados da Busca Baseada em Classes aplicada à Otimização do Gerenciamento de Combustível Intra-Núcleo

7.1. Resultados

7.1.1. OGCIN-BE

Tomando-se como motivação as similaridades entre PCs foi realizada uma busca com base na BE para a OGCIN do 7º. ciclo da usina Angra 1, para confirmar ou refutar nossas expectativas. No nosso modelo de simetria de 1/8 de núcleo (figura 2.2), bem como nos trabalho descrito em [12], exceto pelo EC central que não é permutado, os ECs pertencentes aos eixos de simetria (quartetos) são apenas permutados com outros ECs que pertençam aos eixos de simetrias. Os ECs octetos (ou os ECs que não pertencem aos eixos de simetria) são somente permutados com outros ECs octetos. Já que existem 10 ECs quartetos FAs e 10 ECs octetos, tem-se então aproximadamente $1{,}3\times10^{13}$ permutações possíveis. Seguindo as regras para a BE descritas no capítulo 3 (item 3.3.2.1), dado um PC específico, sua vizinhança determinísticamente obtida possui 90 PCs derivados. Na busca BE realizada neste trabalho, todos os PCs de uma determinada vizinhança foram avaliados, exceto aqueles gerados e avaliados em alguma iteração anterior, para evitar que o algoritmo estagnasse a execução, repetindo as

mesmas soluções correntes. O critério de parada foi o número de 5000 avaliações. O algoritmo de BE implementado segue a descrição mostrada na figura 7.

Dez experimentos (execuções do *OGCIN-BE*) foram realizados para analisar seu comportamento da BE aplicada à OGCIN de Angra 1. Os resultados gerais são exibidos na tabela 7.1. A figura 7.1 mostra a evolução da fitness do experimento número 6 (o experimento que obteve o melhor resultado) ao longo das iterações.

Tabela 7.1 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BE*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o primeiro resultado com a melhor fitness	Número total de soluções correntes válidas
#1	1411 / 1.427	444	0
#2	1510 / 1.471	620	0
#3	**1057 / 1.381**	2104	55
#4	1082 / 1.498	354	0
#5	1274 / 1.458	1059	0
#6	**1252 / 1.375**	795	55
#7	**1011 / 1.392**	444	56
#8	**1065 / 1.394**	356	57
#9	1360 / 1.491	531	0
#10	974 / 1.438	444	0
Média	1200 / 1.433	715	22
Desv. Pad.	187 / 0.046	535	29

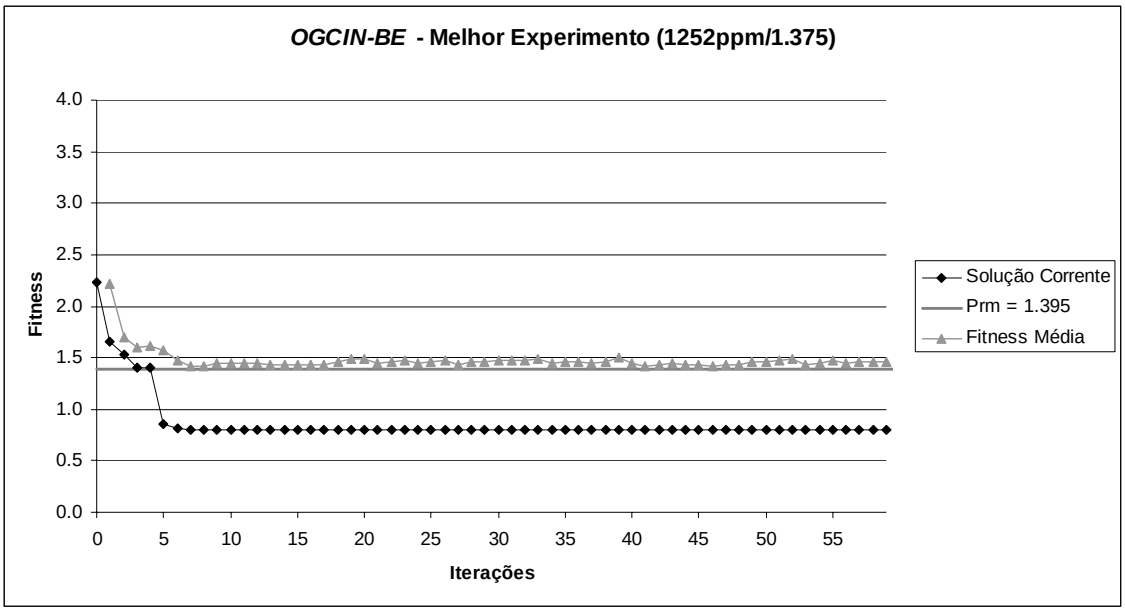


Figura 7.1 – Comportamento da fitness para o melhor experimento do algoritmo *OGCIN-BE* em 5000 evaluations (experimento #6).

7.1.2. OGCIN-BBC_Greedy_50 e OGCIN-BBC_List_50

Nestes experimentos, avaliaram-se todos os PCs de uma determinada Vizinhança de uma solução corrente, exceto aqueles gerados e avaliados em iterações precedentes. As tabelas 7.2 e 7.3 exibem os resultados para os algoritmos *OGCIN-BBC_Greedy_50* e *OGCIN-BBC_List_50*, respectivamente com 10 experimentos para cada um. As figuras 7.2 e 7.3 mostram a evolução das fitnesses dos algoritmos nos melhores experimentos (#5 para o *OGCIN-BBC_Greedy_50* e #6 para o *OGCIN-BBC_List50*) ao longo de 5000 avaliações.

Tabela 7.2 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC _Greedy_50*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	1167 / 1.407	3931	0
#2	1344 / 1.466	5044	0
#3	**1237 / 1.395**	4809	33
#4	**1198 / 1.395**	728	83
#5	**1311 / 1.392**	4429	71
#6	**1303 / 1.395**	2179	88
#7	**1272 / 1.386**	1499	83
#8	**1247 / 1.381**	3658	83
#9	1184 / 1.418	5007	0
#10	**1269 / 1.392**	2992	22
Média	1253 / 1.403	3428	47
Desv. Pad.	58 / 0.025	1530	37

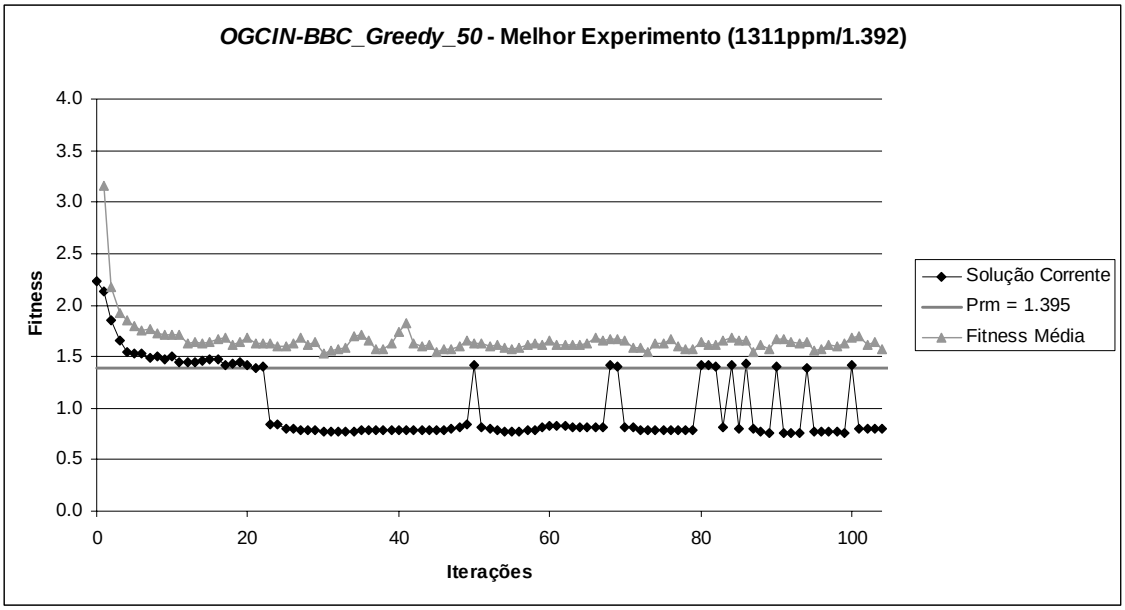


Figura 7.2 – Comportamento da fitness para o melhor experimento do algoritmo *OGCIN-BBC _Greedy_50* em 5000 evaluations (experimento #5).

Tabela 7.3 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC _List_50*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	1470 / 1.440	1436	0
#2	1398 / 1.454	4996	0
#3	1034 / 1.414	1779	0
#4	**1206 / 1.391**	2822	93
#5	**1194 / 1.382**	2676	43
#6	**1364 / 1.391**	2905	102
#7	**1215 / 1.374**	3896	96
#8	**1231 / 1.389**	4166	98
#9	**1265 / 1.382**	3770	29
#10	**1074 / 1.394**	579	87
Média	1245 / 1.401	2903	55
Desv. Pad.	136 / 0.027	1358	45

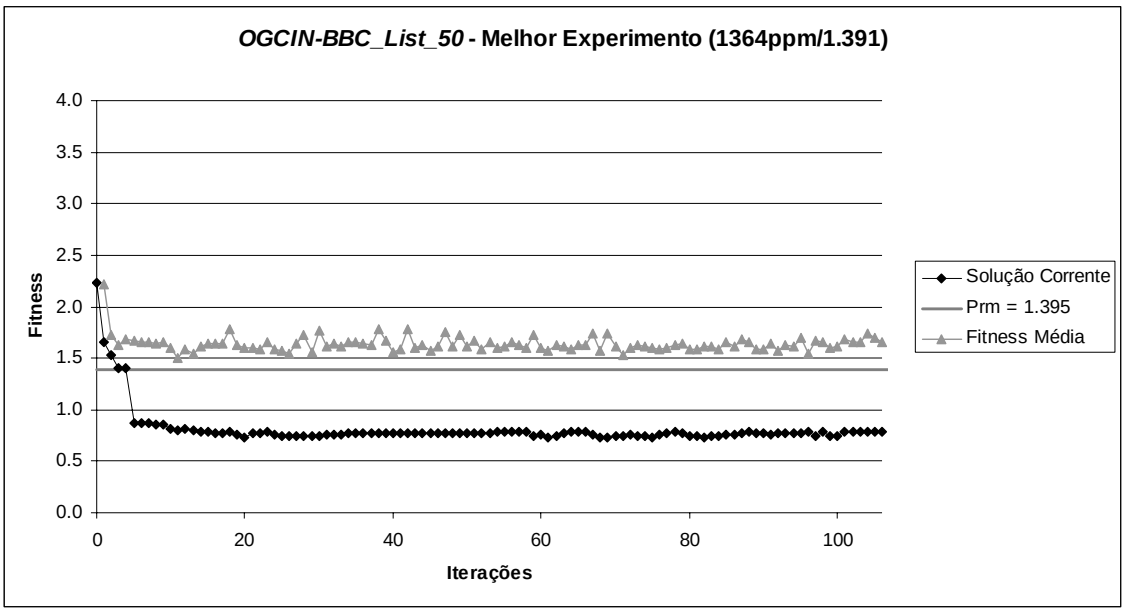


Figura 7.3 – Comportamento da fitness para o melhor experimento do algoritmo *OGCIN-BBC _List_50* em 5000 evaluations (experimento #6).

7.1.3. OGCIN-BBC_Greedy_Random_10 e OGCIN-BBC_List_Random_10

Nos experimentos dos algoritmos *OGCIN-BBC_Greedy_Random_10* e *OGCIN-BBC_List_Random_10*, somente dez entre os 50 PCs da vizinhança da solução corrente são avaliados, selecionados aleatoriamente. Se um dos dez tiver sido avaliado anteriormente então um outro é selecionado, substituindo-o. As tabelas 7.4 e 7.5 mostram dez experimentos para cada um dos algoritmos. As figuras 7.4 e 7.5 mostram a evolução dos algoritmos para os melhores experimentos em 5000 avaliações (#3 para o algoritmo *OGCIN-BBC_Greedy_Random_10* e #10 para o algoritmo *OGCIN-BBC_List_Random_10*).

Tabela 7.4 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC_Greedy_Random_10*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	**1262 / 1.378**	4950	42
#2	**1178 / 1.392**	4910	116
#3	**1335 / 1.361**	1410	113
#4	**1194 / 1.389**	680	88
#5	**1213 / 1.378**	3950	82
#6	**1230 / 1.379**	3930	97
#7	**1251 / 1.389**	4370	48
#8	**1259 / 1.363**	3730	49
#9	**1191 / 1.384**	3570	78
#10	**1211 / 1.393**	2940	59
Média	1232 / 1.381	3444	77
Desv. Pad.	46 / 0.011	1410	27

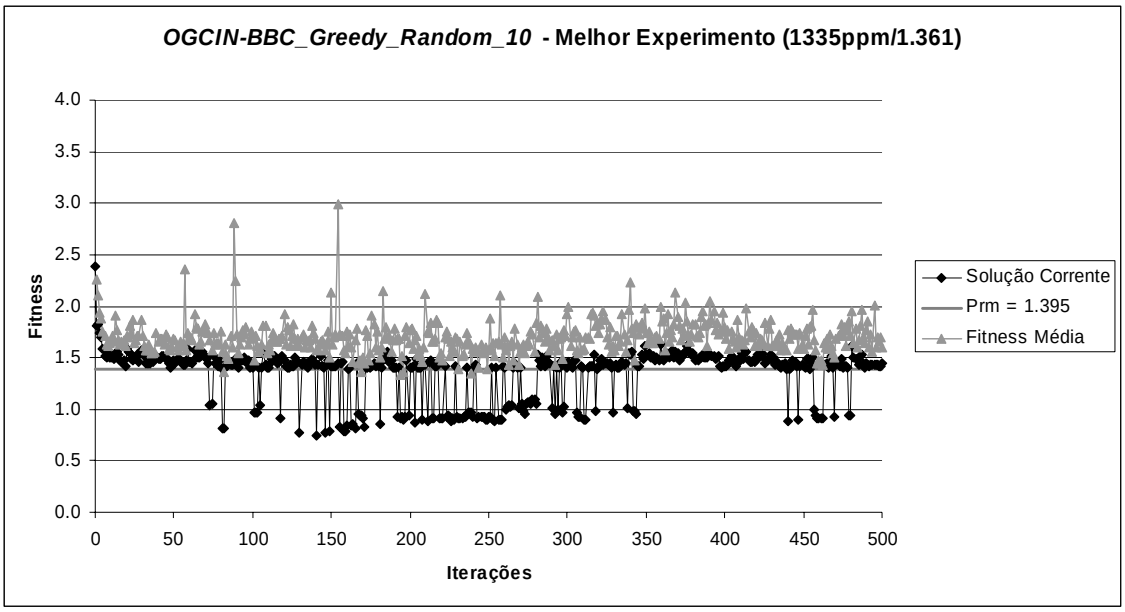


Figura 7.4 – Comportamento da Fitness e Fitness Média para o melhor experimento do algoritmo *OGCIN-BBC_Greedy_Random_10* em 5000 avaliações (experimento #3).

Tabela 7.5 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC _List_Random_10*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	**1226 / 1.390**	1050	318
#2	**1261 / 1.379**	840	138
#3	**1218 / 1.371**	4320	383
#4	**1336 / 1.391**	3930	222
#5	**1415 / 1.387**	4610	67
#6	**1407 / 1.383**	1830	268
#7	**1089 / 1.375**	3950	421
#8	**1222 / 1.369**	1860	332
#9	**1227 / 1.395**	1620	471
#10	**1431 / 1.393**	380	250
Média	1283 / 1.383	2439	287
Desv. Pad.	111 / 0.009	1594	125

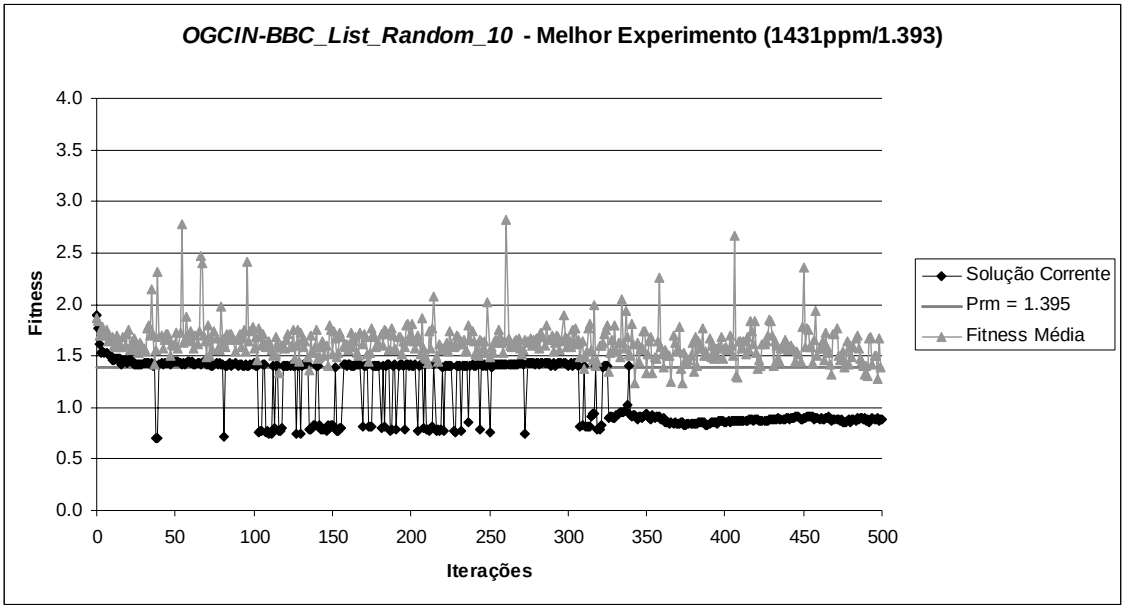


Figura 7.5 – Comportamento da Fitness e Fitness Média para o melhor experimento do algoritmo *OGCIN-BBC_List_Random_10* em 5000 avaliações (experimento #10).

7.1.4. OGCIN-BBC_Greedy_LocalSwaps e OGCIN-BBC_List_LocalSwaps

Nos experimentos dos algoritmos *OGCIN-BBC_ Greedy_LocalSwaps* e *OGCIN -BBC_List_ LocalSwaps*, um número variável de PCs é avaliado em cada iteração. As trocas são realizadas de acordo com as regras mostradas no capítulo 3 (item 3.3.2.2). A avaliação de PCs gerados e avaliados anteriormente é suprimida. As tabelas 7.6 and 7.7 exibem os resultados para 10 experimentos de cada algoritmo respectivamente. As figuras 7.6 e 7.7 mostram a evolução da fitness para os melhores experimentos de cada um dos algoritmos em 5000 avaliações (#8 para *OGCIN-BBC_ Greedy_LocalSwaps* e #8 para *OGCIN-BBC_List_ LocalSwaps*).

Tabela 7.6 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC_Greedy_LocalSwaps*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	**1291 / 1.384**	2471	182
#2	**1184 / 1.388**	476	8
#3	**1180 / 1.382**	1895	9
#4	**1077 / 1.372**	639	32
#5	**1147 / 1.392**	1806	29
#6	**1044 / 1.367**	2556	29
#7	**1118 / 1.365**	4647	18
#8	**1328 / 1.374**	3940	36
#9	**1122 / 1.391**	3850	29
#10	**1065 / 1.363**	784	16
Média	1156 / 1.378	2306	40
Desv. Pad.	94 / 0.011	1471	51

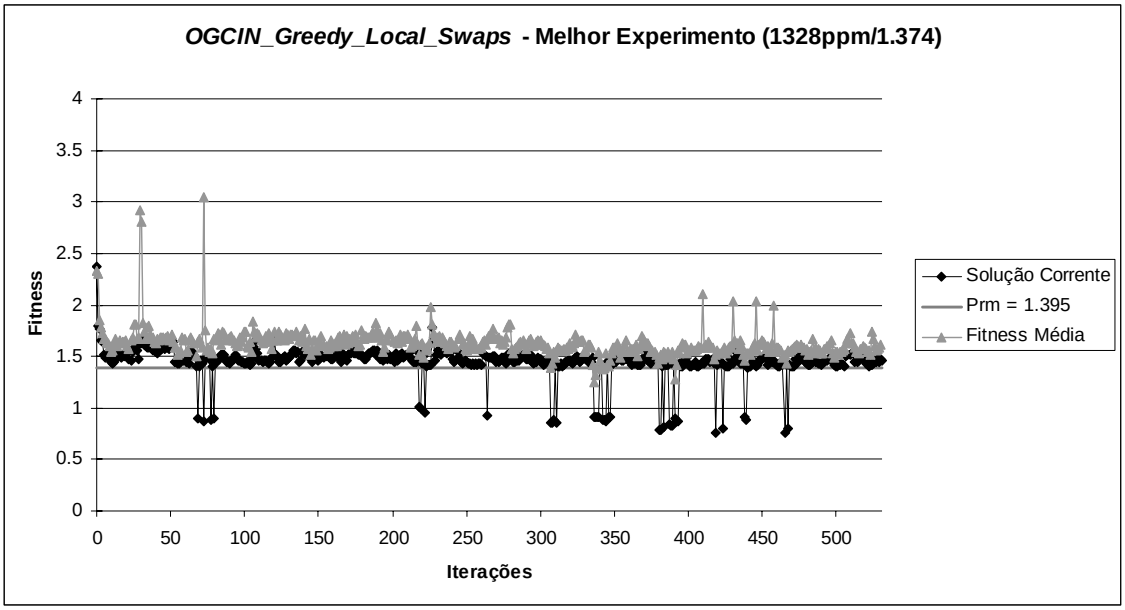


Figura 7.6 – Comportamento da Fitness e Fitness Média para o melhor experimento do algoritmo *OGCIN-BBC_Greedy_LocalSwaps* em 5000 avaliações (experimento #8).

Tabela 7.7 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC _List_LocalSwaps*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	**1237 / 1.387**	4665	199
#2	**1191 / 1.383**	4196	139
#3	**1165 / 1.377**	3845	36
#4	**1101 / 1.393**	3960	22
#5	**1239 / 1.377**	1192	211
#6	**1161 / 1.394**	4783	82
#7	**1312 / 1.372**	4617	22
#8	**1329 / 1.373**	846	12
#9	**1184 / 1.370**	1561	90
#10	**1151 / 1.384**	531	15
Média	1207 / 1.381	3020	83
Desv. Pad.	72 / 0.009	1754	76

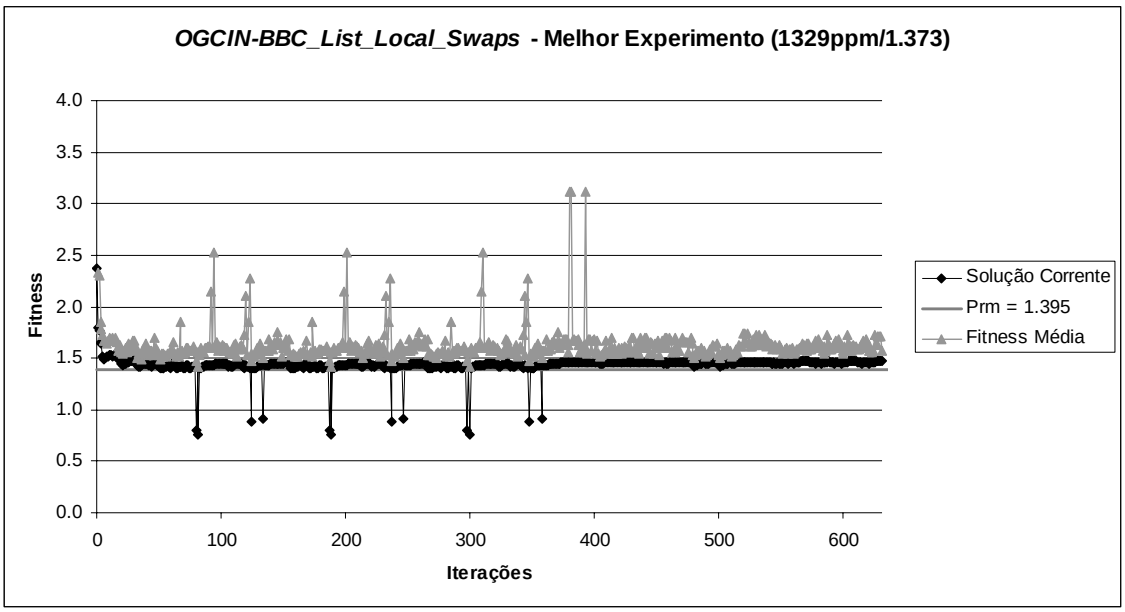


Figura 7.7 – Comportamento da Fitness e Fitness Média para o melhor experimento do algoritmo *OGCIN-BBC_List_LocalSwaps* em 5000 avaliações (experimento #8).

7.1.5. OGCIN-BBC com Heurística Nuclear (HAVR)

Os resultados para os algoritmos *OGCIN-BBC_HAVR_Greedy_MinPlus2*, *OGCIN-BBC_HAVR_Greedy_LongRun* encontram-se nas tabelas 7.8 e 7.9, respectivamente. As figuras 7.8, 7.9 exibem a evolução dos melhores experimentos das três variantes (#5 e #3, respectivamente).

Tabela 7.8 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC _Greedy_MinPlus2*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	**1296 / 1.387**	574	153
#2	**1198 / 1.358**	4377	130
#3	**1223 / 1.382**	4834	126
#4	**1158 / 1.353**	1156	99
#5	**1311 / 1.384**	4939	167
#6	**1158 / 1.368**	3424	137
#7	**1166 / 1.393**	4920	167
#8	**1173 / 1.393**	4121	111
#9	**1276 / 1.379**	456	123
#10	**1206 / 1.390**	2369	128
Média	1217 / 1.379	3117	134
Desv. Pad.	58 / 0.014	1831	22

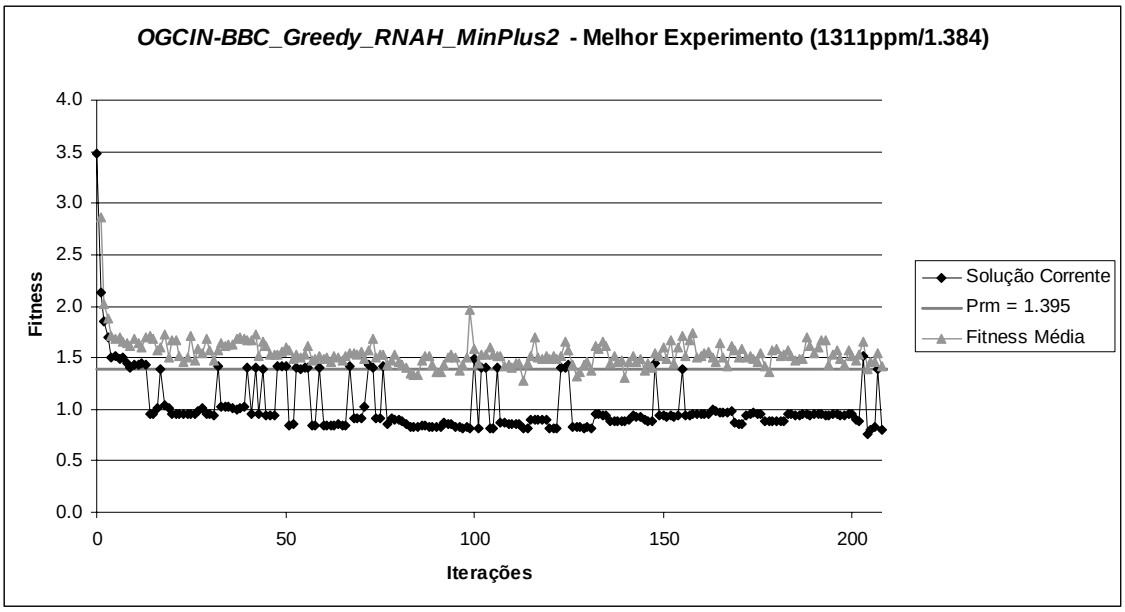


Figura 7.8 – Comportamento da Fitness e Fitness Média para o melhor experimento do algoritmo *OGCIN-BBC_HAVR_Greedy_MinPlus2* em 5000 avaliações (experimento #5).

Tabela 7.9 – Resultados gerais de 10 experimentos para o algoritmo *OGCIN-BBC_HAVR_Greedy_LongRun*. Melhores resultados no formato C_B *(ppm)*/ P_{rm} em 5000 avaliações. Resultados válidos com relação aos requisitos de segurança em negrito.

Experimento	Melhor Resultado	Avaliações necessárias para achar o melhor resultado	Número total de soluções correntes válidas
#1	**1432 / 1.395**	106014	1179
#2	**1421 / 1.389**	37076	1242
#3	**1494 / 1.394**	13892	1167
#4	**1258 / 1.393**	79947	319
#5	**1419 / 1.388**	116539	745
#6	**1421 / 1.389**	71495	1133
#7	**1432 / 1.395**	14143	920
#8	**1432 / 1.395**	5976	815
#9	**1446 / 1.394**	72033	1180
#10	**1422 / 1.389**	44096	1162
Média	1418 / 1.392	56121	986
Desv. Pad.	60 / 0.003	39107	291

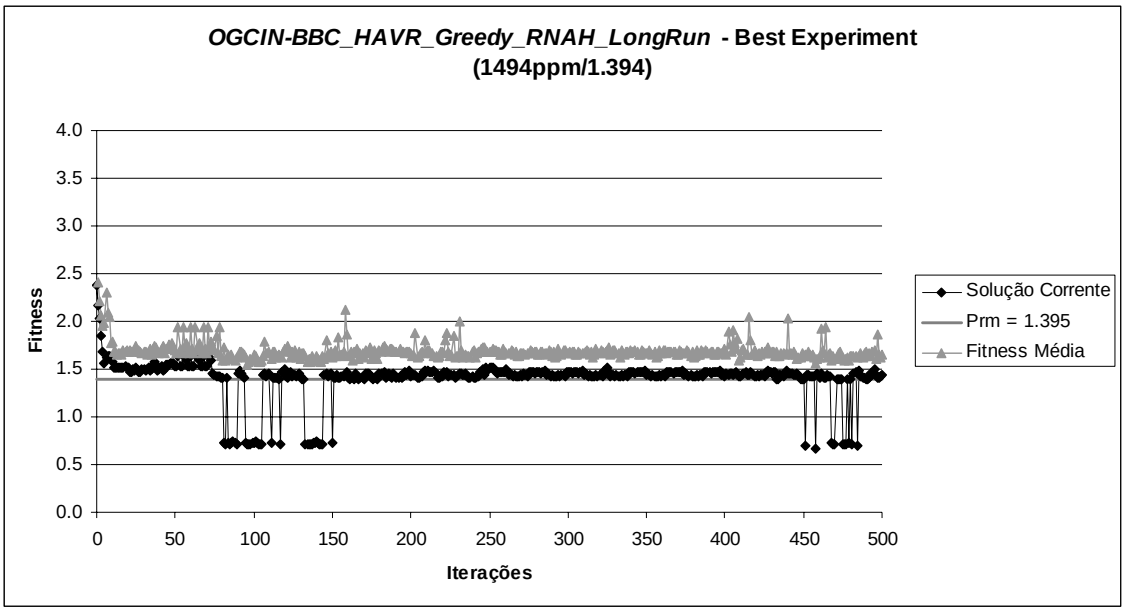


Figura 7.9 – Comportamento da Fitness e Fitness Média para o melhor experimento do algoritmo *OGCIN-BBC_HAVR_Greedy_LongRun* em 5000 avaliações (experimento #3).

A tabela 7.10 mostra as médias para os experimentos envolvendo cada uma das variantes do OGCIN, para fins de comparação.

Tabela 7.10 – Comparação entre os resultados da RS e do PSO com HAVR (C_B/P_{rm} [*número de avaliações*] {*exps. com sols. válidas*}).

Experimento	**Resultado**	**Número de experimentos com soluções válidas**
OGCIN-BE	1200 / 1.433*	04/10
OGCIN-BBC_Greedy_50	1253 / 1.403*	07/10
OGCIN-BBC_List_50	1245 / 1.401*	07/10
OGCIN-BBC_Greedy_Random_10	1232 / 1.381	10/10
OGCIN-BBC_List_Random_10	1283 / 1.383	10/10
OGCIN-BBC_Greedy_LocalSwaps	1156 / 1.378	10/10
OGCIN-BBC_List_LocalSwaps	1207 / 1.381	10/10
OGCIN-BBC_Greedy_HAVR_MinPlus2	1217/1.379	10/10
OGCIN-BBC_Greedy_HAVR_LongRun	1418/1.392	10/10

7.2. Comentários

A BBC é uma MHO que realiza uma busca aproximada baseada em trocas relevantes entre ECs considerando informações sobre suas características nucleares. Resultados comparáveis a outras técnicas, com alguns deles ainda melhores no sentido da maximização do ciclo de operação, demonstram que foi possível para este algoritmo a retenção de informação importante sobre as estruturas dos PCs ao longo das iterações, usando tais padrões intrínsecos para construir soluções candidatas com qualidade considerável com relação aos parâmetros nucleares.

O algoritmo pode ser implementado para reter em memória as soluções que foram geradas e avaliadas anteriormente, embora a BBC não tenha a proibição de mudanças como base para uma decisão; em vez disso, usa simples mas efetivas regras, como não avaliar um PC que já tenha sido gerado e avaliado. Também difere da BE, cuja abordagem determinística faz o algoritmo verificar tediosamente soluções que são similares, o que causa aprisionamento entre uma série de PCs similares. As evidências do aprisionamento da busca baseada na BE indicam o fato de que a busca poderia ter melhores resultados se os ECs de diferentes classes fossem trocados, ou seja, se apenas as mudanças mais relevantes nas estruturas dos PCs fossem verificadas durante a execução do algoritmo, o que permite à BBC uma melhor movimentação e conseqüente exploração do espaço de busca.

Ademais, quando a troca entre ECs de diferentes classes é verificada, mesmo se não há um melhor resultado na iteração seguinte durante a execução do algoritmo, é possível, continuar a busca mesmo a partir de uma solução pior, evitando aprisionamento local e esta é uma outra vantagem da BBC. Além disto, o número relativamente baixo de regras e restrições, bem como a ausência de parâmetros numéricos a serem ajustados, fazem-no um algoritmo de fácil implementação, o que o torna muito atrativo para uso na OGCIN, já que novas regras poderiam ser adaptadas, bem como implementações híbridas com outros algoritmos.

É importante salientar que, com a BBC para a OGCIN, uma exploração de um espaço de busca altamente complexo está sendo realizada com base nas trocas entre ECs de diferentes classes. Como uma conseqüência da "paisagem" de fitness (*fitness*

landscape) não-linear, as soluções correntes obtidas a partir destas trocas não são regularmente válidas em cada iteração. Por exemplo, para o PSORK aplicado à OGCIN (capítulo 5; como exemplo, a figura 5.2), com o enxame realizando a busca em espaços contínuos e mapeando as posições em soluções combinatórias, a média das fitnesses referentes às melhores posições das partículas do enxame ao longo das iterações diminui conforme a fitness referente à melhor posição global também diminui. Portanto no caso da BBC, não é possível nem afirmar que a fitness das soluções correntes vá ter sempre um comportamento decrescente (para o problema de miminização), embora tal fato ocorra nas iterações iniciais, nem que as médias das Vizinhanças vão também possuir um comportamento sempre decrescente, e de fato, os experimentos indicam que ambas as avaliações das soluções correntes e as médisa das vizinhanças em geral oscilam ao longo das iterações. Não obstante, em diversos experimentos, em especial para os algoritmos *OGCIN-BBC_List*, depois de uma certa iteração, quando a solução corrente alcançou uma solução válida, diversas soluções subseqüentes, também foram válidas (um exemplo pode ser visto na figura 7.5 com o algoritmo *OGCIN-BBC_List_Random_10* após a iteração 350, approximadamente). Tal fato é notável, dado que ocorreu em diversos experimentos, reforçando o aspecto de retenção de padrões intrínsecos durante a BBC.

A discussão sobre os resultados específicos de cada variante é apresentada nos itens abaixo.

7.2.1. OGCIN-BE

Há fortes evidências em todos os experimentos de que o algoritmos não é capaz de escapar de mínimos locais, apresentando uma convergência prematura, quando um PC válido ou mesmo um não-válido em termos de segurança é alcançado, corroborando a descrição da BE feita em [67]. Esse fato é fortemente relacionado à nossa consideração inicial de que algumas soluções possuem avaliações de fitness muito similares. Um exemplo se encontra no experimento #6 (quando, em vez de 5000 são feitas 10000 avaliações), onde a avaliação do melhor PC resulta nos parâmetros nucleares $C_B = 1256ppm$ e $P_{rm} = 1.393$, valores que por sua vez, resultam no valor de 0.796178 para a função de fitness. Um outro PC com $C_B = 1255ppm$ e $P_{rm} = 1.393$ (fitness = 0.796813) é bem similar ao melhor PC e foi achado na iteração 5159. Outro exemplo, o PC com $C_B = 1252ppm$ e $P_{rm} = 1.376$ (fitness = 0.798722) é também muito similar.

Outra evidência do aprisionamento local é a indicação da coluna "Número total de soluções correntes válidas" na tabela 2. Em 6 dos 10 experimentos, ocorre o valor 0 (zero) nesta coluna, o que significa que o algoritmo realiza sua busca e não consegue achar nenhuma solução válida nestes casos. Para outros experimentos (#7 e #8), feasible PCs que são achados no começo dos experimentos não são melhorados. Em termos práticos, os resultados demonstram que o algoritmo mantém sua procura entre PCs muito similares, levando indubitavelmente ao aprisionamento da busca baseada na BE e mesmo em alguns casos, em regiões que nem mesmo podem ser consideradas ótimos locais da OGCIN.

7.2.2. OGCIN-BBC_Greedy50 e OGCIN-BBC_List50

Os resultados das variants *OGCIN-BBC_Greedy_50* e *OGCIN-BBC_List_50* (tabelas 7.3 e 7.4, respectivamente) indicam que 7 dos 10 experimentos alcançaram resultados válidos, de acordo com o parâmetro P_{rm}, relacionado à segurança. Sendo assim, estas duas variantes não apresentam os melhores resultados para a OGCIN, mas ambos foram uma primeira evidência do potencial de nosso algoritmo de busca para lidar com o aprisionamento, em relação à abordabem BE.

Um comportamento interessante é mostrado na figura 7.3, para o algoritmo *OGCIN-BBC_List_50*. Neste caso, uma vez que o algoritmo acha uma certa solução corrente válida, todas as soluções correntes subseqüentes também são válidas. De fato, isto é mais provável que aconteça com os algoritmos que implementam a estrutura *List*, pois há uma tendência maior de retenção de padrões intrínsecos. Um outro exemplo pode ser visto na figura 7.5, para o algoritmo *OGCIN-BBC_List_ Random_10*.

7.2.3. OGCIN-BBC_Greedy_Random_10 e OGCIN-BBC_List_Random_10

As figuras 7.4 e 7.5 mostram uma oscilação maior para a fitness média das Vizinhanças a cada iteração em relação aos experimentos *OGCIN-BE* e *OGCIN-BBC_Greedy_50* and *List_50*, provavelmente devido à seleção aleatória dos vizinhos de uma solução corrente que será avaliada. Apesar disto, essa seleção aleatória produziu melhores resultados do que aquelas obtidas com a avaliação determinística de todos os membros de uma Vizinhança.

Além disto, o melhor resultado para a variante *OGCIN-BBC_List_Random_10* (experimento #10: $C_B = 1431ppm$ e $P_{rm} = 1.393$, em 380 avaliações) foi melhor que os melhores resultados para o PSORK no mesmo problema de otimização, que foram $C_B = 1394ppm$ e $P_{rm} = 1.384$ (capítulo 5, tabela 5.1, experimento #1), para um número de avaliações menor ou igual a 6000. A média dos resultados para *OGCIN-BBC_List_Random_10* ($C_B = 1283ppm$ e $P_{rm} = 1.383$) também foi melhor que a média

para o PSORK (C_B = *1233ppm* e P_{rm} = *1.381*) para 6000 e mesmo para 10000 avaliações (C_B = *1254ppm and* P_{rm} = *1.386*) – ambas as médias exibidas na tabela 5.1.

7.2.4. OGCIN-BBC_Greedy_Local_Swaps e OGCIN-BBC_List_Local_Swaps

Esta primeira investigação sobre como direcionar a seleção de vizinhos a serem avaliados resultou em resultados médios C_B = *1156ppm* e P_{rm} = *1.378* (*OGCIN-BBC_Greedy_Local_Swaps*), bem como C_B = *1207ppm* e P_{rm} = *1.382* (*OGCIN-BBC_List_Local_Swaps*). Todos os resultados dos 10 experimentos também foram válidos com relação ao parâmetro P_{rm}.

7.2.5. OGCIN-BBC com Heurística de Engenharia Nuclear

Os resultados do algoritmo *OGCIN-BBC_Greedy_HAVR_MinPlus2* (em média C_B = *1217ppm* e P_{rm} = *1.375*) foram melhores em média do que os resultados das variantes *OGCIN-BBC_Greedy_Local_Swaps* e *OGCIN-BBC_List_Local_Swaps*, com um menor desvio padrão relativo a C_B. Em todos os experimentos, o número de avaliações necessárias para achar o melhor resultado foi menor que 3000 e em média, 1255. Isso indica que o uso da HAVR foi mais efetivo que as *Local_Swaps* para essa busca, corroborando os resultados mostrados no capítulo 6, que demonstraram que a HAVR pode ser utilizada para direcionar a busca realizada por MHOs.

O algoritmo *OGCIN-BBC_Greedy_RNAH_Long_Run* obteve os melhores resultados achados para essa variante da OGCIN (Código de Física de Reatores RECNOD, ECs quartetos não trocados com ECs octetos) em processamento serial (para um número de avaliações menor que 150000 avaliações), comparando-se aos trabalhos [12, 15, 18, 23, 36], que usaram critérios similares.

Na prática, evidências apontam para o fato que, em um sentido geral, soluções válidas com uma fitness alta possuem um grau mais alto (grau de acordo com a definição apresentada no capítulo 3, seção 3.3). Embora o contrário nem sempre seja verdadeiro, usamos uma abordagem "agressiva" para exploração do espaço de busca, em contraste com a abordagem conservative da busca baseada no grau mínimo da Vizinhança (*OGCIN-BBC_Greedy_HAVR_MinPlus2*), no sentido do risco de não serem encontradas soluções válidas com um grau mais alto. Para um número de avaliações menor que 6000 e 10000, o melhor resultado obtido foi $C_B = 1432ppm$ e $P_{rm} = 1.395$ (5976 avaliações; experimento #8); para um número de avaliações menor que 15000, o melhor resultado obtido foi $C_B = 1494$ e $P_{rm} = 1.394$ (13892 avaliações; experimento #3). Assim, a estratégia de usar um número de graus mais alto (5, 6 e 7) para direcionamento da busca foi uma estratégia que surtiu efeito para seleção dos vizinhos de uma solução corrente, com relação à maximização do ciclo de operação. De fato, um número mais alto de graus em alguns casos reduziu o número de soluções correntes válidas ao longo da execução do algoritmo, embora na prática tenha levado a melhores resultados em um número razoável de iterações.

Capítulo 8

Conclusões e Propostas de Trabalhos Futuros

A OGCIN é um importante problema de Engenharia Nuclear, de alta complexidade, riscos de aproximação e regiões desconexas de soluções possíveis, estudado por mais de 40 anos. Além da otimização manual, realizada por engenheiros de projeto de padrão de carregamento e métodos baseados em conhecimento, MHOs vêm sido utilizadas ao longo dos anos, com resultados notáveis para este problema. As principais características de MHOs, entre outras, são um baixo acoplamento com o problema e memorização das principais características das soluções candidatas. Heurísticas, por outro lado, são altamente acopladas aos problemas, já que representam métodos específicos baseados em conhecimento ou critérios para decidir qual alternativa seguir para atingir determinado objetivo.

Nesta tese, relatamos a investigação MHO para a OGCIN, em particular o PSO e a BBC. Também mostramos o desenvolvimento uma heurística chamada HAVR. Os resultados gerais de nosso trabalho podem ser comparados a outros através da tabela 8.1.

Tabela 9.1 – Comparação entre os melhores resultados de diversas técnicas aplicadas à otimização do ciclo 7 da Usina Nuclear de Angra 1.

Referência	C_B	P_{rm}[1]	Técnica	Heurística	Aval.
Chapot et al., 1999 [12]	955	1.345	Manual	-	-
Chapot et al., 1999 [12]	1026	1.390	GA	No	4000
Machado and Schirru, 2002 [18]	1297	1.384	Ant-Q	Sim	200
Machado, 2005 [15]	1242	1.361	PBIL	Não	6000
Machado, 2005 [15]	1305	1.349	PBIL-MO[2]	Sim	10000
De Lima, 2005 [19]	1424	1.386	RCFA[3]	Sim	329000
Caldas and Schirru, 2008 [17]	1554	1.381	FPBIL[4]	Não	430364
Meneses et al., 2009a [23]	1394	1.384	PSORK	Não	4000
Meneses et al., 2009b [29]	1174	1.394	RS-RNAH	Sim	271
Meneses et al., 2009b [29]	1207	1.390	PSORK-RNAH	Sim	477
Esta Tese	1431	1.393	BBC-LR10[5]	Não	2439[6]
Esta Tese	1494	1.394	BBC-HAVR[7]	Sim	56121[8]

[1] F_{XY} para otimização manual
[2] PBIL Multi-Objetivo
[3] Redes Conectivas de Formigas Artificiais
[4] *Parameter-Free* PBIL (PBIL sem parâmetros)
[5] *OGCIN-BBC_List_Random_10*
[6] Média das avaliações necessárias para achar o melhor resultado (Tabela 7.5)
[7] *OGCIN-BBC_HAVR_Greedy_LongRun*
[8] Média das avaliações necessárias para achar o melhor resultado (Tabela 7.9)

Além das conclusões específicas de cada técnica e método já evidenciados nos capítulos anteriores, como conclusões gerais desta tese, podemos dizer que:

(i) conseguimos desenvolver com sucesso um modelo de PSO, o PSORK, para o problema combinatório da OGCIN, com resultados superiores a outras metaheurísticas de otimização;

(ii) desenvolvemos uma Heurística para a OGCIN, a HAVR, que demostrou proporcionar uma redução geral do número de avaliações dos algoritmos RS e PSORK, o que é muito importante devido ao custo computacional dos códigos de Física de Reatores utilizados para avaliação de soluções candidatas na OGCIN; e

(iii) também desenvolvemos uma MHO eficiente a ser aplicada à OGCIN, a BBC, com resultados competitivos em relação a outras MHOs consagradas.

Além disso, é importante frisar que as técnicas aqui propostas podem ser ainda investigadas em maior profundidade e bem como podem ser utilizadas em composição com outras técnicas a fim de formarem métodos híbridos (como a utilização com métodos de nichos), o que pode constituir trabalhos futuros a serem desenvolvidos.

Referências Bibliográficas

[1] LEVINE, S., "In-Core Fuel Management of Four Reactor Types". In: Williams, M. M. R. (ed.), *Handbook of Nuclear Reactor Calculation* Vol. II., EUA, CRC Press, 1987.

[2] MALDONADO, G. I., "Optimizing LWR Cost of Margin One Fuel Pin at a Time", ***IEEE Transactions on Nuclear Science*** 52, pp. 996-1003, 2005.

[3] STEVENS, J. G., SMITH, K. S., REMPE, K. R., DOWNAR, T. J., "Optimization of Pressurized Water Reactor Shuffling by Simulated Annealing with Heuristics", ***Nuclear Science and Engineering*** 121, pp. 68-77, 1995.

[4] POON, P. W., PARKS, G. T., "Optimising PWR Reload Core Designs". In: Manner, R., Manderick, B. (eds.), *Parallel Problems Soving from Nature II*, EUA, Elsevier Science, 1992.

[5] WALL, I., FENECH, H., "Application of Dynamic Programming to Fuel Management Optimization", ***Nuclear Science and Engineering*** 22, pp. 285-297, 1965.

[6] TABAK, D., "Optimization of Nuclear Reactor Fuel Recycle via Linear and Quadratic Programming", ***IEEE Transactions on Nuclear Science*** 15 (1), pp. 60-64, 1968.

[7] TAILLARD, E. D., GAMBARDELLA, L. M., GENDREAU, M., POTVIN, J.-Y., "Adaptive memory programming: A unified view of metaheuristics", ***European Journal of Operational Research*** 135, pp. 1-6, 2001.

[8] PARKS, G. T., "An intelligent stochastic optimization routine for nuclear fuel cycle design", ***Nuclear Technology*** 89, pp. 233-246, 1990.

[9] KROPACZEK, D. J., TURINSKY, P. J., "In-Core Nuclear Fuel Management Optimization for Pressurized Reactors Utilizing Simulated Annealing", ***Nuclear Technology*** 95, pp. 9-31, 1991.

[10] PARKS, G.T., "Multi-objective pressurized water reactor reload core design by non-dominated genetic algorithm search", ***Nuclear Science and Engineering*** 124, 178–187, 1996.

[11] DeCHAINE, M.D., FELTUS, M.A., "Fuel management optimization using genetic algorithms and expert knowledge", ***Nuclear Science and Engineering*** 124, 188–196, 1996.

[12] CHAPOT, J. L. C., DA SILVA, F. C., SCHIRRU, R., "A new approach to the use of genetic algorithms to solve the pressurized water reactor's fuel management optimization problem", ***Annals of Nuclear Energy*** 26, pp. 641-655, 1999.

[13] LIN, C., YANG, J.-I., LIN, K.-J., WANG, Z.-D., "Pressurized Water Reactor Loading Pattern Design Using the Simple Tabu Search", ***Nuclear Science and Engineering*** 129, pp. 61-71, 1998.

[14] MACHADO, M. D., *Um Novo Algoritmo Evolucionário com Aprendizagem LVQ para Otimização de Problemas Combinatórios como a Recarga de Reatores Nucleares*. Dissertação de Mestrado, COPPE/UFRJ, Brasil, 1999.

[15] MACHADO, M. D., *Algoritmo evolucionário PBIL multi-objetivo aplicado ao problema da recarga de reatores nucleares*. Tese de Doutorado, COPPE/UFRJ, Brasil, 2005.

[16] SCHIRRU, R., DE LIMA, A.M.M., MACHADO, M.D., "Parallel evolutionary methods applied to a PWR core reload pattern optimization". In: *Proceedings of the Seventh International FLINS Conference on Applied Artificial Intelligence*, Itália, 2006.

[17] CALDAS, G. H. F., SCHIRRU, R., "Parameterless evolutionary algorithm applied to the nuclear reload problem", ***Annals of Nuclear Energy*** 35, pp. 583-590, 2008.

[18] MACHADO, L., SCHIRRU, R., "The Ant-Q algorithm applied to the nuclear reload problem", ***Annals of Nuclear Energy*** 29, pp. 1455-1470, 2002.

[19] DE LIMA, A. M. M., *Recarga de reatores nucleares utilizando redes conectivas de colônias artificiais*. Tese de Doutorado, COPPE/UFRJ, Brasil, 2005.

[20] DE LIMA, A. M. M., SCHIRRU, R., DA SILVA, F. C., MEDEIROS, J. A. C. C., "A nuclear reactor core fuel reload optimization using ant colony connective networks", ***Annals of Nuclear Energy*** 35, pp. 1606-1612, 2008.

[21] KENNEDY, J., EBERHART, R. C., **Swarm Intelligence**. EUA, Morgan Kaufmann Publishers, 2001.

[22] EBERHART, R. C., KENNEDY, J., *A new optimizer using particles swarm theory*. In: *Proceedings of Sixth International Symposium on Micro Machine and Human Science*, EUA, IEEE Service Center, pp. 39–43, 1995.

[23] MENESES, A. A. M., MACHADO, M. D., SCHIRRU, R., "Particle Swarm Optimization applied to the nuclear reload problem of a Pressurized Water Reactor", ***Progress in Nuclear Energy*** 51, pp. 319-326, 2009.

[24] DOMINGOS, R. P., SCHIRRU, R., PEREIRA, C. M. N. A., "Particle swarm optimization in reactor core design", ***Nuclear Science and Engineering*** 152, pp. 197–203, 2006.

[25] MEDEIROS, J. A. C. C., SCHIRRU, R., "Identification of nuclear power plant transients using the Particle Swarm Optimization algorithm", ***Annals of Nuclear Energy*** 35, pp. 576-582, 2008.

[26] OLIVEIRA, M. V., SCHIRRU, R., 2009. "Applying particle swarm optimization for tuning a neuro-fuzzy inference system for sensor monitoring", ***Progress in Nuclear Energy*** 51, pp. 177-183.

[27] WAINTRAUB, M., SCHIRRU, R., PEREIRA, C. M. N. A., "Multiprocessor Modeling of Parallel Particle Swarm Optimization applied to Nuclear Engineering Problems" ***Progress in Nuclear Energy*** 51, pp. 680-688, 2009.

[28] BEAN, J. C., "Genetic algorithms and random keys for sequencing and optimization", ***ORSA Journal of Computing*** 6 (2), 1994.

[29] MENESES, A. A. M., GAMBARDELLA, L. M., SCHIRRU, R., "A New Approach for Heuristic-Guided Search in the In-Core Fuel Management Optimization". In Press. ***Progress in Nuclear Energy*** doi:10.1016/j.pnucene.2009.07.007, 2009.

[30] MENESES, A. A. M., SCHIRRU, R., "Particle Swarm Optimization Aplicado ao Problema Combinatório com Vistas à Solução do Problema de Recarga em um Reator Nuclear". In: *Proceedings of the International Nuclear Atlantic Conference INAC 2005*, Brasil, 2005.

[31] MENESES, A. A. M., *Otimização por Enxame de Partículas Aplicado ao Problema Combinatório da Recarga de um Reator Nuclear*, Dissertação de Mestrado, COPPE/UFRJ, Brasil, 2005.

[32] MENESES, A. A. M., SCHIRRU, R., "Particle Swarm Optimization applied to the combinatorial problem in order to solve the Nuclear Reactor Fuel Reloading Problem". In: *Proceedings of the Seventh International FLINS Conference on Applied Artificial Intelligence*, Itália, 2006.

[33] MENESES, A. A. M., MACHADO, M. D., MEDEIROS, J. A. C. C., SCHIRRU, R., "Particle swarm optimization with random keys applied to the nuclear reactor reload problem". In: *Proceedings of the International Nuclear Atlantic Conference INAC 2007*, Brasil, 2007.

[34] MENESES, A. A. M., SCHIRRU, R., "Guaranteed Convergence Particle Swarm Optimization Model Adapted to Combinatorial Problems for Application to the Nuclear Reactor Reload Problem". In: *Anais do XI Encontro de Modelagem Computacional*, Brasil, 2008.

[35] MENESES, A. A. M., SCHIRRU, R., "Influence of the Confinement in the Particle Swarm Optimization of Combinatorial Problems for Application to the Nuclear Reactor Reloading". In: *Proceedings of the 8th International FLINS Conference*. Inglaterra, World Scientific, pp. 1075-1080, 2008.

[36] CHAPOT, J. L. C., *Otimização Automática de Recargas de Reatores a Água Pressurizada Utilizando Algoritmos Genéticos*. Tese de Doutorado. COPPE/UFRJ, Brasil, 2000.

[37] BERGH, F. van den, *An Analysis of Particle Swarm Optimizers*. Tese de Doutorado, Universidade de Pretória, África do Sul, 2001.

[38] BRITS, R., *Niching Strategies for Particle Swarm Optimization*, Dissertação de Mestrado, Universidade de Pretória, África do Sul, 2002.

[39] ENGELBRECHT, A. P., ***Computational Intelligence***, England, John Wiley and Sons, 2007.

[40] NAFT, B. N., SESONSKE, A., "Pressurized Water Optimal Fuel Management", ***Nuclear Technology*** 14, pp. 123-132, 1972.

[41] HAIBACH, B. V., FELTUS, M. A., "A study on the optimization of integral fuel burnable absorbers using the genetic algorithm based CIGARO fuel management system", ***Annals of Nuclear Energy*** 24, pp. 439-448, 1997.

[42] GALPERIN, A., KIMHY, Y., "Application of Knowledge-Based Methods to In-Core Fuel Management", ***Nuclear Science and Engineering*** 109, pp. 103-110, 1991.

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

[44] PARKS, G. T., LEWINS, J. D.,. "In-core fuel management and optimization - the state of the art", ***Nuclear Europe Worldscan XII***, 41. 1992

[45] ERDOĞAN, A., GEÇKINLI, M., "A PWR reload optimization code (*XCore*) using artificial neural networks and genetic algorithms". ***Annals of Nuclear Energy*** 30, pp. 35-53, 2003.

[46] ZIVER, A. K., PAIN, C. C., Carter, J. N., de Oliveira, C. R. E., Goddard, A. J. H., Overton, R. S., "Genetic algorithms and artificial neural networks for loading pattern optimization of advanced gas-cooled reactors", ***Annals of Nuclear Energy*** 31, pp. 431-457, 2004.

[47] VANDERBEI, R. J., ***Linear Programming: Foundations and Extensions***. EUA, Kluwer Academic Publishers, 1992.

[48] DORIGO, M., GAMBARDELLA, L. M., "Ant colony system: a cooperative learning approach to the traveling salesman problem". ***IEEE Transactions on Evolutionary Computation*** 1 (1), pp. 53–66, 1997.

[49] LAWLER, E. L., "The Quadratic Assignment Problem". ***Management Science*** 9, pp. 586-599, 1963.

[50] KIRKPATRICK, S., GELATT, C. D., VECCHI, M. P., "Optimization by Simulated Annealing", ***Science*** 220 (4598), pp. 671-680, 1983.

[51] HOLLAND, J. H., ***Adaptation in Natural and Artificial Systems***, EUA, University of Michigan Press, 1975.

[52] GOLDBERG, D. E., ***Genetic Algorithms in Search, Optimization and Machine Learning***, EUA, Addison Wesley, 1989.

[53] JIANG, S., ZIVER, A. K., CARTER, J. N., PAIN, C. C., GODDARD, A. J., FRANKLIN, S., PHILLIPS, H. J., "Estimation of distribution algorithms for nuclear fuel management optimization", ***Annals of Nuclear Energy*** 33, pp. 1039-1057, 2006.

[54] PARSOPOULOS, K. E., VRAHATIS, M. N., "Recent aproaches to global optimization problems through Particle Swarm Optimization", ***Natural Computing*** 1, pp. 235-306, 2002.

[55] KENNEDY, J., EBERHART, R. C., "A discrete binary version of the particle swarm algorithm". In: *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, pp. 4104–4109, 1997.

[56] WANG, K.-P., HUANG, L., ZHOU, C.-G., PANG, W., "Particle Swarm Optimization for Traveling Salesman Problem". In: *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, pp. 1583-1585, China, 2003.

[57] CLERC, M., "Discrete Particle Swarm Optimization, Illustrated by the Traveling Salesman Problem". ***France Télécom Recherche & Développement***, 2004.

[58] YIN, P.-Y., "A discrete particle swarm algorithm for optimal polygonal approximation of digital curves". ***Journal of Visual Communication and Image Representation*** 15, pp. 241-260, 2004.

[59] SALMAN, A., AHMAD, I. e Al-MADANI, S., "Particle Swarm Optimization for Task Assignment Problem". ***Microprocessors and Microsystems*** 26 (8), pp. 363-371, 2002.

[60] SHI, X.H., LIANG, Y.C., LEE, H.P., LU, C., WANG, Q.X. Particle swarm optimization-based algorithms for TSP and generalized TSP. ***Information Processing Letters*** 103, pp. 169-176, 2007.

[61] PANG,W.,WANG, K.-P., ZHOU, C.-G., DONG, L.-J., LIU, M., ZHANG, H.-Y.,WANG, J.-Y. Modified particle swarm optimization based on space transformation for solving traveling salesman problem. In: *Proceedings of the Third International Conference on Machine Learning and Cybernetics*, 2004.

[62] LANGENBUCH, S., MAURER, W., WERNER, W., "Coarse mesh flux expansion method for analysis of space-time effects in large water reactor cores". ***Nuclear Science and Engineering*** 63, pp. 437-456, 1977.

[63] LIU, Y. S. et al., 1985. *ANC: A Westinghouse Advanced Nodal Computer Code.* Tech Rep WCAP-10965, Westinghouse.

[64] MONTAGNINI, B., SORAPERRA, P., TRENTAVIZI, C., SUMINI, M., ZARDINI, D. M., "A well-balanced coarse mesh flux expansion method". ***Annals of Nuclear Energy*** 21 (11), pp. 45-53, 1994.

[65] PEARL, J., ***Heuristics: intelligent search strategies for computer problem solving***. EUA, Addison-Wesley, 1985.

[66] SUH, J. S., LEVINE, S. H., "Optimized Automatic Reload Program for Pressurized Water Reactors Using Simple Direct Optimization Techniques". ***Nuclear Science and Engineering*** 105, pp. 371-382, 1990.

[67] YAMAMOTO, A., 1997. "A Quantitative Comparison of Loading Pattern Optimization Methods for In-Core Fuel Management of PWR". ***Journal of Nuclear Science and Technology*** 34, pp. 339-347.

[68] LAWLER, E. L., LENSTRA, J. K., KAN, A. H. G. R., SHMOYS, D. B., 1985. ***The Traveling Salesman Problem: a guided tour of combinatorial optimization***, EUA, John Wiley & Sons.

[69] PAPADIMITRIOU, C.H., STEIGLITZ, K., ***Combinatorial Optimization***. EUA, Prentice-Hall, 1982.

[70] TSPLIB. http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95.

Anexos

Anexo I – Resumo da Publicação [29]

Particle Swarm Optimization aplicado ao Problema Combinatório com Vistas à Solução do Problema da Recarga de um Reator Nuclear

INAC 2005 – International Nuclear Atlantic Conference

Autores: Anderson Alvarenga de Moura Meneses e Roberto Schirru

This work focuses on the usage the Artificial Intelligence technique Particle Swarm Optimization (PSO) to optimize the fuel recharge at a nuclear reactor. This is a combinatorial problem, in which the search of the best feasible solution is done by minimizing a specific objective function. However, in this first moment it is possible to compare the fuel recharge problem with the Traveling Salesman Problem (TSP), since both of them are combinatorial, with one advantage: the evaluation of the TSP objective function is much more simple. Thus, the proposed methods have been applied to two TSPs: Oliver 30 and Rykel 48. In 1995, KENNEDY and EBERHART presented the PSO technique to optimize non-linear continued functions. Recently some PSO models for discrete search spaces have been developed for combinatorial optimization. Although all of them having different formulation from the ones presented here. In this paper, we use the PSO theory associated with to the Random Keys (RK) model, used in some optimizations with Genetic Algorithms. The Particle Swarm Optimization with Random Keys (PSORK) results from this association, which combines PSO and RK. The adaptations and changes in the PSO aim to allow the usage of the PSO at the nuclear fuel reload. This work shows the PSORK being applied to the proposed combinatorial problem and the obtained results.

Anexo II – Resumo da Publicação [30]

Particle Swarm Optimization aplicado ao Problema Combinatório com Vistas à Solução do Problema da Recarga em um Reator Nuclear

Dissertação de Mestrado
Autor: Anderson Alvarenga de Moura Meneses
Orientador: Prof. Roberto Schirru

O foco deste trabalho é a utilização da técnica de Otimização com Enxame de Partículas (Particle Swarm Optimization, PSO) com vistas à otimização da recarga de combustível em um reator nuclear. Para viabilizar esta aplicação do PSO, foi desenvolvido um modelo utilizando chaves aleatórias (Random Keys, RK). Tal adaptação tem como finalidade possibilitar a utilização do PSO em problemas combinatórios com grau de complexidade similar à da otimização da recarga de combustível em reatores nucleares, já que as modelagens existentes não se apresentam satisfatórias para este tipo de problema. Mostramos os resultados obtidos e a análise da otimização de dois problemas de caixeiro viajante (Traveling Salesman Problem, TSP): Oliver 30 e Rykel 48. Estabelecemos também comparações dos resultados deste trabalho com os obtidos por outros métodos de otimização já consolidados.

Anexo III – Resumo da Publicação [31]

Particle Swarm Optimization applied to the combinatorial problem in order to solve the Nuclear Reactor Fuel Reloading Problem

FLINS 2006 – 7th International Conference on Applied Artificial Intelligence

Autores: Anderson Alvarenga de Moura Meneses e Roberto Schirru

This work focuses on the use of the Artificial Intelligence metaheuristic technique Particle Swarm Optimization (PSO) to optimize a nuclear reactor fuel reloading. This is a combinatorial problem, in which the goal is to find the best feasible solution, minimizing a specific objective function. However, in the first moment it is possible to compare the fuel reloading problem with the Traveling Salesman Problem (TSP), since both of them are combinatorial and similar in terms of complexity, with one advantage: the evaluation of the TSP objective function is simpler. Thus, the proposed method has been applied to two TSPs: Oliver 30 and Rykel 48. In 1995, KENNEDY and EBERHART presented the PSO technique to optimize non-linear continuous functions. Recently some PSO models for discrete search spaces have been developed for combinatorial optimization, although all of them have different formulation from the one presented in this work. Here we use the PSO theory associated with to the Random Keys (RK) model, used in some optimizations with Genetic Algorithms, as a way to transform the combinatorial problem into a continuous space search. The Particle Swarm Optimization with Random Keys (PSORK) results from this association, which combines PSO and RK. The adaptations and changes in the PSO aim to allow the appliance of the PSO at the nuclear fuel reloading problem. This work shows the PSORK applied to the TSP and the obtained results as well.

Anexo IV – Resumo da Publicação [32]

Particle Swarm Optimization with Random Keys applied to the Nuclear Reactor Reload Problem

INAC 2007 –International Nuclear Atlantic Conference

Autores: Anderson Alvarenga de Moura Meneses, Marcelo Dornellas Machado, Jose Antonio Carlos Canedo Medeiros e Roberto Schirru

In 1995, KENNEDY and EBERHART presented the Particle Swarm Optimization (PSO), an Artificial Intelligence metaheuristic technique to optimize non-linear continuous functions. The concept of Swarm Intelligence is based on the socials aspects of intelligence, it means, the ability of individuals to learn with their own experience in a group as well as to take advantage of the performance of other individuals. Some PSO models for discrete search spaces have been developed for combinatorial optimization, although none of them presented satisfactory results to optimize a combinatorial problem as the nuclear reactor fuel reloading problem (NRFRP). In this sense, we developed the Particle Swarm Optimization with Random Keys (PSORK) in previous research to solve Combinatorial Problems. Experiences demonstrated that PSORK performed comparable to or better than other techniques. Thus, PSORK metaheuristic is being applied in optimization studies of the NRFRP for Angra 1 Nuclear Power Plant. Results will be compared with Genetic Algorithms and the manual method provided by a specialist. In this experience, the problem is being modeled for an eight-core symmetry and three-dimensional geometry, aiming at the minimization of the Nuclear Enthalpy Power Peaking Factor as well as the maximization of the cycle length.

Anexo V – Resumo da Publicação [33]

Particle Swarm Optimization with Random Keys applied to the Nuclear Reactor Reload Problem

INAC 2007 – International Nuclear Atlantic Conference

Autores: Anderson Alvarenga de Moura Meneses, Marcelo Dornellas Machado, Jose Antonio Carlos Canedo Medeiros e Roberto Schirru

In 1995, KENNEDY and EBERHART presented the Particle Swarm Optimization (PSO), an Artificial Intelligence metaheuristic technique to optimize non-linear continuous functions. The concept of Swarm Intelligence is based on the socials aspects of intelligence, it means, the ability of individuals to learn with their own experience in a group as well as to take advantage of the performance of other individuals. Some PSO models for discrete search spaces have been developed for combinatorial optimization, although none of them presented satisfactory results to optimize a combinatorial problem as the nuclear reactor fuel reloading problem (NRFRP). In this sense, we developed the Particle Swarm Optimization with Random Keys (PSORK) in previous research to solve Combinatorial Problems. Experiences demonstrated that PSORK performed comparable to or better than other techniques. Thus, PSORK metaheuristic is being applied in optimization studies of the NRFRP for Angra 1 Nuclear Power Plant. Results will be compared with Genetic Algorithms and the manual method provided by a specialist. In this experience, the problem is being modeled for an eight-core symmetry and three-dimensional geometry, aiming at the minimization of the Nuclear Enthalpy Power Peaking Factor as well as the maximization of the cycle length.

Anexo VI – Resumo da Publicação [34]

Guaranteed Convergence Particle Swarm Optimization Model Adapted To Combinatorial Problems for Application to the Nuclear Reactor Reload Problem

EMC 2008 – XI Encontro de Modelogem Computacional

Autores: Anderson Alvarenga de Moura Meneses e Roberto Schirru

The concept of Swarm Intelligence is based on socials aspects of intelligence, that is, the ability of individuals to learn with their own experience in a group as well as to take advantage of the performance of other individuals. The Particle Swarm Optimization (PSO) is a Computational Intelligence optimization metaheuristic with several models for continuous search spaces, although PSO models for discrete search spaces for combinatorial optimization with satisfactory results for application to combinatorial problems such as the Traveling Salesman Problem (TSP) or analogous problems such as the Nuclear Reactor Reload Problem (NRRP) still need to be developed. In this sense, this paper presents the application of the Guaranteed Convergence Particle Swarm Optimization with Random Keys (GC-PSORK) to optimize benchmark problems in order to optimize the NRRP. The GC-PSO has demonstrated both theoretical development and successful application to optimization of continuous problems. For the optimization of combinatorial problems, its discrete version GC-PSORK has accomplished considerable results under the conditions discussed in this paper.