UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
FACULDADE DE CIÊNCIAS EXATAS
CURSO DE SISTEMAS DE INFORMAÇÃO

MHV: Uma Metaheurística baseada no comportamento da *Vitis Vinifera*

Hilton Lesllie de Oliveira

Diamantina
2016

UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
FACULDADE DE CIÊNCIAS EXATAS

MHV: Uma Metaheurística baseada no comportamento da Vitis Vinifera

Hilton Lesllie de Oliveira

Orientador:
Prof. Dr. Alessandro Vivas Andrade
Coorientador:
Cristiano Grijó Pitangui

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação, como parte dos requisitos exigidos para a conclusão do curso.

Diamantina
2016

MHV: Uma Metaheurística baseada no comportamento da Vitis Vinifera

Hilton Lesllie de Oliveira

Orientador:
Prof. Dr. Alessandro Vivas Andrade
Coorientador:
Cristiano Grijó Pitangui

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação, como parte dos requisitos exigidos para a conclusão do curso.

APROVADO em 12/08/2016

_____
Profª Drª. Luciana Assis – UFVJM

_____
Prof. Me. Rafael Santin – UFVJM

_____
Prof Dr. Alessandro Vivas Andrade – UFVJM

AGRADECIMENTOS

Agradeço a Deus, a videira verdadeira, por tudo o que me fez, pela vida concedida, por todos aqueles que Ele colocou em minha e por todas as oportunidades de aprendizado e crescimento, agradeço com tudo que tenho e tudo que sou, não sou nada sem Ti;

Aos meus pais, espero um dia ser digno de merecer tudo que fazem por mim, se eu tiver metade do caráter e sabedoria que vocês têm, terei orgulho do homem que me tornei. Palavras não são suficientes para agradecer a vocês por todo amor e por tudo o que me ensinaram e ainda ensinam. Tudo que posso é dizer sinceramente: Amo vocês incondicionalmente;

À minha vó pela constante intercessão, suas orações são um alicerce para minha vida e sempre serei grato a senhora por isso, Te amo vó;

Ao meu irmão Wendel, pelo companheirismo, força, ensinamentos e sobretudo pelo exemplo que é para mim, você é o melhor irmão que alguém poderia pedir. Te amo mano;

À Tainá, por ser o ponto de estabilidade em meio ao caos, sempre manter a calma independente das circunstâncias e me proporcionar sorrisos mesmo em meio aos problemas. Te amo princesa;

Aos amigos, em especial àqueles da República OROBORO, República DjAmas, República Dois Homens e meio, a todos da Igreja Batista Nacional em Turmalina e da Primeira Igreja Batista em Diamantina pelo amor, carinho e interseção constantes;

Aos meus professores, todos aqueles que colaboraram para minha formação intelectual, citando Newton: “Se eu vi mais longe, foi por estar sobre ombros de gigantes." ;

Aos meus orientadores Cristiano e Vivas:

Cristiano, por acreditar no potencial do trabalho, pelas correções durante todo esse percurso, sem a sua visão aprimorada e seletiva, não seria possível chegar aos resultados aqui alcançados, pela parceria e paciência ao longo da produção deste trabalho. Muito obrigado;

Vivas por todo apoio, disposição e paciência para corrigir os erros e estar sempre incentivando a correr atrás, para destacar e levar os frutos de todo o trabalho a frente, para alçar voos que anteriormente não teria almejado. Muito obrigado;

A todos que de alguma forma contribuíram para que eu chegasse até aqui, minha mais sincera gratidão.

De tempos em tempos, os homens tropeçam na verdade,
mas a maioria deles se levanta e segue adiante
como se nada tivesse acontecido.
-WINSTON CHURCHILL-

RESUMO

O presente trabalho apresenta uma nova meta-heurística para problemas de otimização chamada "**Metaheurística *Vitis Vinifera"*** (**MHV**) que se baseia na organogênese da videira. A videira possui uma capacidade natural adaptativa que através dos séculos possibilitou sua sobrevivência e disseminação nos mais diversos climas e condições naturais. A forma como cada fator influencia o seu crescimento faz com que ela se molde as mais diversas situações, encontrando as melhores condições de sobrevivência. Além disso, algumas intervenções humanas facilitam essa adaptação e corrigem problemas encontrados durante o seu desenvolvimento natural. Estes fatores aliados à sua inerente adaptabilidade ampliam de forma significativa sua capacidade de desenvolvimento. Baseado nesses aspectos e nas influências naturais e humanas sobre o desenvolvimento da mesma realizou-se o mapeamento da organogênese da Videira em uma estrutura algorítmica.

O trabalho apresenta a inspiração natural para a criação do algoritmo bem como analisa e discute seus principais passos. O algoritmo proposto foi implementado e funções de otimização foram utilizadas para testar sua robustez, velocidade de convergência, precisão e desempenho. As funções utilizadas para validação da proposta foram a *OneMax*, *Trap5*, *Sphere*, *Rosenbrock* e *Rastrigin*, todas já amplamente empregadas para testes de meta-heurísticas. Após os testes, os resultados foram comparados a outras quatro técnicas clássicas bio-inspiradas, a saber, Algoritmos Genéticos, Enxame de partículas, *Population-Based Incremental Learning* (PBIL) e Acasalamento de Abelhas.

Os resultados obtidos evidenciam que a meta-heurística foi bem sucedida e apontam o seu grande potencial para resolver problemas de otimização, visto que em todas as funções utilizadas para teste os valores obtidos se mantiveram próximos aos melhores, superando expectativas e mantendo sempre um ótimo desempenho, tanto em tempo, quanto em resultados.

Numa análise geral, a **MHV** se mostrou promissora em todas as funções, situando-se, em média, em terceiro lugar dos melhores resultados encontrados, porém, sendo quase imbatível no quesito tempo de execução.

PALAVRAS-CHAVE: Meta-heurísticas, Bioinspirada, Otimização.

LISTA DE FIGURAS

LISTA DE EQUAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

MHV – Metaheurística *Vitis Vinifera*
AG – Algoritmo Genético
PSO – Otimização por Enxame de Partículas
PBIL – Population-Based Incremental Learning
MBO – Metaheurística Acasalamento de Abelhas (Marriage Bees Optimization)
IA – Inteligência Artificial

SUMÁRIO

INTRODUÇÃO

Problemas de otimização combinatória são problemas onde se busca determinar o maior ou o menor valor que uma função pode assumir em um dado intervalo. Esses problemas são comuns em nosso cotidiano, exemplos simples são: qual o melhor caminho a se tomar para se chegar em determinado local, qual a melhor combinação de itens para se levar em um passeio, dividir tarefas em um trabalho de grupo, etc.

Apesar dessa aparente simplicidade, existem problemas dentro desse conjunto que são extremamente complexos, como o Problema do Caixeiro Viajante, onde por exemplo problemas com 20 cidades, para se ter uma ideia da magnitude do tempo envolvido na resolução, por enumeração completa de todas as possíveis soluções, tem-se $6 . 10^{16}$ rotas possíveis. Assim, um computador que avalia uma rota em cerca de 10^{-8} segundos, gastaria cerca de 19 anos para encontrar a melhor rota! Mesmo considerando os rápidos avanços tecnológicos dos computadores, uma enumeração completa de todas essas rotas é inconcebível para um número elevado de cidades. Algumas vezes não é possível garantir que a rota de custo mínimo seja encontrada em tempo polinomial, assim, no pior caso, todas as possíveis soluções devem ser analisadas (JAMILSON et al., 2011).

Para problemas como o citado acima geralmente são utilizados métodos que nem sempre encontram a solução ótima, pois na prática, em geral, é suficiente encontrar uma "boa" solução para o problema. É nesse âmbito que a Computação biologicamente inspirada aparece, trazendo técnicas de solução para alguns problemas.

A Computação biologicamente inspirada, ou simplesmente bio-inspirada, é o campo de investigação que recorre a metáforas ou modelos teóricos dos sistemas biológicos, a fim de projetar ferramentas computacionais ou sistemas para resolver problemas complexos. Os resultados alcançados são algoritmos ou sistemas que têm uma semelhança com os fenômenos ou modelos biológicos estudados (CASTRO, L. N. DE; ZUBEN, 2005). Em sua totalidade os resultados obtidos atrelam a base biológica observada, aos conceitos computacionais. Essa união têm suas raízes em diversos aspectos, mas principalmente os padrões comportamentais existentes na natureza que são fonte de inspiração e aprendizado para todos os campos científicos, inclusive a computação. Diversos desses padrões fornecem base para os mais variados estudos, porém aqui nos ateremos a um caso especial interessantíssimo, a videira.

A videira possui uma capacidade natural adaptativa que através dos séculos possibilitou a sobrevivência e disseminação nos mais diversos climas, sendo alterada pelos mesmos, gerando variações a partir de um ancestral comum, porém totalmente diferente das espécies iniciais (MCGOVERN, 2003). Baseado nessa capacidade natural adaptativa da videira e nas influências exercidas do ambiente sobre ela, diversos estudos foram realizados, visando entender melhor como se dá o desenvolvimento e como se relaciona com esses fatores ambientais, a partir daí foram criadas práticas agrícolas e métodos que moldem a produtividade e qualidade da videira, e foram elaborados modelos de

desenvolvimento que mapeiam esses fatores e como eles direcionam influenciam o crescimento da planta e consequentemente a sua produção.

Fato é que a capacidade natural, juntamente com os fatores ambientais e intervenções humanas guiam o desenvolvimento da planta, gerando novas espécies e possibilitando a sobrevivência da videira em diversos ambientes, o comportamento da videira e a forma como ela se desenvolve perante as características do ambiente pode ser visto naturalmente, como a distribuição de galhos em locais onde a planta possui um apoio para se fixar, os apêndices se ligam a corpos vizinhos oferecendo suporte para o crescimento, a maneira que a folhagem se distribui em climas com temperaturas mais altas, entre outros aspectos, cada fator está relacionado ao comportamento em resposta ao ambiente.

Seguindo essa linha, a proposta do presente artigo é apresentar uma metaheurística bio-inspirada baseada na organogênese da Vitis Vinifera (**MHV**). Tendo como foco a resolução de problemas de otimização, em um determinado espaço de busca. E assim como o crescimento da planta é alterado pelos fatores iniciais e pelas influências exercidas no seu desenvolvimento, os parâmetros iniciais e alterações ao decorrer do desenvolvimento, moldam a videira pelo espaço de busca e as soluções encontradas, podendo se traçar um padrão de desenvolvimento e melhores circunstâncias para produção e crescimento de acordo com o ambiente e as condições no meio em que ela se encontra.

Esse trabalho de conclusão de curso estrutura-se em seis capítulos, sendo o capítulo corrente, que faz uma introdução geral do trabalho. O segundo capítulo é um referencial teórico que traz ao leitor uma boa base sobre os assuntos abordados, além de conceitos indispensáveis para uma completa compreensão da proposta. O capítulo três apresenta os materiais, métodos e metodologia utilizados durante o desenvolvimento do trabalho e o modo como o mesmo segue padrões que validam e corroboram tanto o método como os resultados alcançados. A seguir o capítulo 4 discute detalhadamente o desenvolvimento natural (base biológica) e o desenvolvimento da metaheurística (base computacional), como os dois se relacionam, expondo a forma como o desenvolvimento natural é visto do ponto de vista computacional e como ele é representado sendo fiel a inspiração natural da metaheurística. Além disso, esse capítulo apresenta a estrutura formal da metaheurística com o algoritmo e fluxogramas. O quinto capítulo traz os resultados obtidos a partir de testes que tinham como principal objetivo a validação do método, bem como a comparação com outros já existentes, incluindo então os testes, resultados e a análise desses resultados. O capítulo final faz uma conclusão sobre o trabalho, destacando possíveis publicações futuras e expondo a contribuição para o conhecimento científico.

REFERÊNCIAL TEÓRICO

1.1 Aspectos computacionais

Alguns conceitos são necessários para a compreensão do que se propõe o presente, conceitos esses que definem e são os fundamentos de todo o plano computacional onde a metaheurística se encontra, esses fundamentos são abordados a seguir, como uma revisão básica para todo o trabalho.

1.1.1 Inteligência Artificial

A inteligência artificial é um campo de estudo da computação e por diversas vezes é descrita como o estudo, planejamento, delineamento e desenvolvimento de sistemas inteligentes, porém existem diversas definições que formalmente a caracterizam de forma diferente, visto a não trivialidade dessa tarefa. Rich e Knight (RICH; KNIGHT, 1990) tratam-na como "O estudo de como fazer os computadores realizarem tarefas que, no momento, as pessoas fazem melhor.", afirmam, porém que essa exposição é efêmera, porque se baseia no estado atual da computação.

Coppin (AUSTIN, 1987) aponta que talvez seja melhor nos perguntarmos o que é inteligência, o que é uma questão complexa com uma resposta não bem definida que confunde biólogos, psicólogos e filósofos por séculos. Então ele apresenta uma definição simplista, "Inteligência artificial é o estudo dos sistemas que agem de uma maneira que, para qualquer observador, parece inteligente." E em seguida dá uma definição que se encaixa "quase" perfeitamente ao método apresentado no presente trabalho, "Inteligência artificial envolve o uso de métodos baseados no comportamento inteligente de humanos e outros animais para resolver problemas complexos." O quase, utilizado na frase, dá se ao fato de que atualmente existem algumas abordagens que criam sistemas baseados no comportamento de plantas e outros hábitos encontrados na natureza, como nesse trabalho que tem como intenção simular o comportamento da videira.

1.1.2 Um breve histórico

A partir do computador moderno a ideia de Inteligência artificial ganhou maior notoriedade e se tornou um campo de estudo e ciência metodológica, mas a sua idealização é bem mais antiga. A história está repleta de pontos onde a ideia já era apresentada desde os filósofos clássicos, com a forma de abordar o processo de pensamento humano, e poemas épicos da mitologia grega, por exemplo, Hefesto - deus dos ferreiros e atividades com metais - construía assistentes de metal e dotava-as de inteligência. Ao longo do tempo, vários outros escritos trouxeram as mais diversas formas de pensar a respeito do assunto, não podendo deixar de citar dentre elas ficções como Frankenstein (Frankenstein: or the Modern Prometheus), Rossum's Universal Robots e as obras de Isaac Asimov.

A IA moderna, bem como o desenvolvimento da computação de forma geral, teve grande influência do trabalho de Alan Turing, um brilhante matemático que usou uma máquina eletromecânica para quebrar códigos alemães durante a segunda guerra mundial. Turing propôs uma série de máquinas e consagrou-se com a projeção de uma máquina que, de acordo com um sistema formal, pudesse fazer operações computacionais. A máquina teórica de Turing pode indicar que sistemas poderosos poderiam ser construídos. Tornou possível o processamento de símbolos, ligando a abstração de sistemas cognitivos e a realidade concreta dos números. Isto é buscado até hoje por pesquisadores de sistemas com Inteligência Artificial (IA). Além disso, desenvolveu um teste para determinar se determinado sistema é dotado de inteligência artificial. Sua máquina pode ser programada de tal modo que pode imitar qualquer sistema formal, ou seja, qualquer sistema de pensamento abstrato bem definido, formulado em um modelo matemático, normalmente esses sistemas caracterizam-se como um par constituído por objetos e regras de derivação. A partir daí a ideia de computabilidade começou a ser delineada, a capacidade de ser computável, existir um método algorítmico de resolver, isto é, uma sequência finita de instruções que possa ser efetuada mecanicamente (MOREIRA, 2003). Devido a esses feitos, Alan Turing é tido como Pai da ciência da computação e da IA (XANTRE, 2015).

O campo de pesquisa, bem como o termo inteligência artificial foi cunhado formalmente em uma conferência no campus do Dartmouth College no verão de 1956, por John MacCarthy (ZUBEN, 2013). Após isso muitos líderes de pesquisa da área prediziam que uma máquina tão inteligente quando um ser humano iria existir em não mais do que uma geração. Com o passar do tempo, ficou óbvio de que as dificuldades para tal tarefa eram muito maiores do que se esperava. Problemas que mostravam algumas das dificuldades foram apresentados posteriormente, como os problemas combinatórios que veremos em seguida, onde um simples cálculo com 2 ou 3 variáveis se tornavam inviáveis para um número maior de variáveis.

1.1.3 Complexidade

Alguns problemas complexos impediram um maior desenvolvimento da IA, frustrando as predições de muitos, esses problemas serão abordados em seguida, porém antes faz-se necessário uma breve explanação a respeito de com

A complexidade de um problema é medida de acordo a sua dificuldade inerente, relativo aos recursos necessários para a sua resolução. A complexidade de um problema está relacionada ao tempo ou ao espaço, sendo complexidade espacial o espaço de memória necessário para executar até o fim, e complexidade temporal o tempo que demora a executar (tempo de execução).

Na ciência da computação existe um método muito utilizado para se analisar a complexidade de um algoritmo, a chamada Notação O Grande (Conhecida também como Big-O Notation, Notação Landau, Notação Bachmann–Landau ou ainda Notação Assintótica), ela descreve o comportamento limitante de uma função quando o argumento

tende a um determinado valor ou ao infinito, geralmente em termos de funções mais simples (CARRANO, 1995)

Matematicamente, Se f e g são funções, a notação $f(n) = O\big(g(n)\big)$ é usada para expressar o fato de que a taxa de crescimento de f não é maior do que a de o g, ou seja $f(n) = O\big(g(n)\big)$ significa que $\exists c\ \forall n(|f(x)| \leq c.|g(n)|)$. Aqui presume-se que f e g são funções com o mesmo domínio e contradomínio. A definição assume ainda que noções de multiplicação e valor absoluto são definidas no contradomínio (AVIGAD; DONNELLY, 2004).

Em relação a um algoritmo A é O de $f(n)$ – denotado $O(\ f(n))$ – se a constante *k* e n_0 existem tal que A gaste não mais que $k * f(n)$ unidades de tempo para resolver um problema de tamanho $n \geq n_0$.

Se um Algoritmo A gasta um tempo proporcional a $f(n)$ é dito que A é $O(f(n))$. O exemplo a seguir ilustra a definição (CARRANO, 1995):

- Supondo que um algoritmo gaste $n^2 - 3 * n + 10$ segundos para resolver um problema de tamanho *n.* Se a constante *k* e n_0 existem tal que
  $k * n^2 > n^2 - 3 * n + 10$ para todo $n > n_0$
  o algoritmo é ordem n^2. De fato, se *k* é 3 e n_0 é 2,
  $3 * n^2 > n^2 - 3 * n + 10$ para todo $n > 2$
  Assim o algoritmo não gasta mais do que $k * n^2$ unidades de tempo para $n > n_0$, portanto é $O(n^2)$.

A **Figura 1** apresenta alguns limites comumente vistos na aplicação da notação O, uma hierarquia básica de complexidade da notação O, sendo da menor complexidade (*O(1)*) até a maior(*O(n!)*).

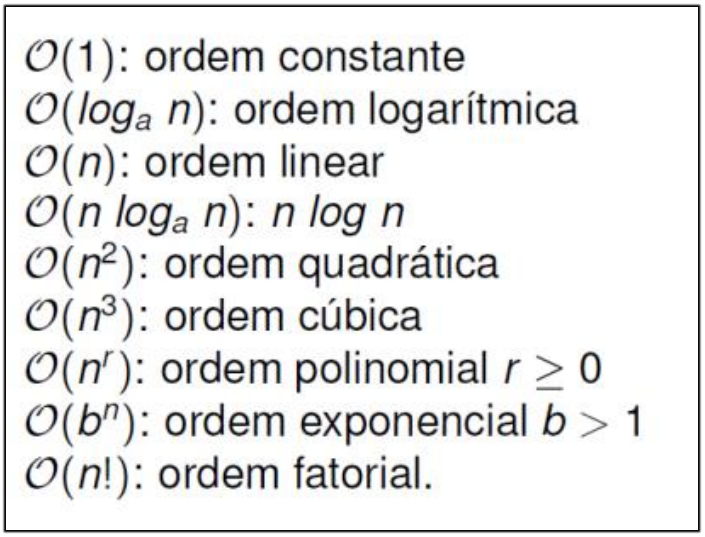

Figura 1 – Hierarquia da notação *O* (COELHO; NETO, 2010)

Falando dos aspectos de complexidade, existem duas grandes classes de problemas, os tratáveis e os intratáveis. Um problema tratável pode ser solucionado por um computador em um tempo aceitável, ou seja, pode ser verificado a partir de um algoritmo de ordem polinomial. Problemas intratáveis podem levar séculos, mesmo para entradas muito pequenas.

Geralmente é utilizado um sistema de classes para separação de problemas quanto a sua complexidade. Essas classes são (ELISA TULER; NIVIO ZIVIANI; CHARLES ORNELAS; LEONARDO ROCHA; LEONARDO MATA., 2006):

- A classe NP, conjunto de todos os problemas que podem ser resolvidos por algoritmos não-deterministas em tempo polinomial.
- A classe P, conjunto de todos os problemas que podem ser resolvidos por algoritmos deterministas em tempo polinomial.

Para mostrar que um determinado problema está em NP, basta apresentar um algoritmo não-determinista que execute em tempo polinomial para resolver o problema. Outra maneira é encontrar um algoritmo determinista polinomial para verificar que uma dada solução é válida.

- Há ainda a classe NP-Completo, problemas que possuem a característica de que se um deles puder ser resolvido em tempo polinomial então todo problema NP-Completo terá uma solução em tempo polinomial. Eles possuem uma roupagem diferente, porém uma raiz igual, por isso essa ligação. Não se conhece nenhum algoritmo que resolva um problema NP-completo em tempo polinomial.

Problemas da classe NP-Completo são considerados os problemas mais difíceis dentro da classe NP. Ainda não se sabe se os problemas em NP podem ser todos resolvidos em tempo polinomial, porém para os problemas dessa classe em específico, ainda não existe algoritmo que o faça.

1.1.4 Otimização Combinatória

Como visto anteriormente, algumas das dificuldades encontradas no meio da história da IA, focalizaram o desenvolvimento e pesquisa em algumas áreas específicas, como o caso citado anteriormente, os algoritmos de otimização combinatória, que são uma área de grande importância para a computação, e possibilitam a análise e otimização de problemas em diversas outros campos, desde gerenciamento de estoque a mapeamento genético.

Um problema de otimização combinatória pode ser definido por um conjunto $E = \{1, \dots, n\}$, um conjunto de soluções viáveis $F \subseteq 2^E$ e uma função objetivo $f : 2^E \rightarrow \mathbb{R}$, todos definidos para cada problema específico (SILVÉRIO; RODRIGUES; STEINER, 2013).

Exemplos de problemas de Otimização Combinatória (OC) são: o Problema do Caixeiro Viajante, o Problema da Cobertura Mínima por Conjuntos, o Problema da Mochila, o Problema da Árvore de Steiner, dentre outros. Esses problemas possuem aplicações em diversas áreas, tais como projeto de redes de telecomunicações, análise de dados, projeto de circuitos impressos, empacotamento de contêineres, roteamento de veículos, análise de cadeias de DNA, etc. (CARNEIRO et al., 2014).

Encontrar comprovadas soluções ótimas em problemas de otimização combinatória é uma tarefa árdua, mas muito progresso vem sendo alcançado com o emprego de técnicas como *branch and bound* (BALAKRISHNAN; BOYD; BALEMI, 1991; BOYD; MATTINGLEY, 2003), plano de corte e programação dinâmica, assim como soluções aproximadas via algoritmos aproximados. Porém muitos problemas de otimização combinatória se beneficiam de métodos heurísticos, esses geralmente são embasados em alguma metaheurística, e que rapidamente produzem soluções de boa qualidade sem necessariamente garantir a otimalidade. Neste sentido, muitas heurísticas modernas para otimização combinatória seguem os padrões descritos no *Guidelines* (HERTZ; WIDMER, 2003), tais como: algoritmos genéticos (ALGORITHMS, 2005; MITCHELL, 1995), *simulated annealing* (BERTSIMAS; TSITSIKLIS, 1993), busca tabu (GLOVER; LAGUNA; MARTI, 2007) (GLOVER, 1989), *variable neighborhood search* (MLADENOVIĆ; HANSEN, 1997) (AVANTHAY; HERTZ; ZUFFEREY, 2003), *scatter search* (GLOVER, 2003; MARTÍ; LAGUNA; GLOVER, 2006), *path relinking* (YAGIURA; IBARAKI; GLOVER, 2006), *iterated local search* (LOURENÇO; MARTIN; STUTZLE, 2002), *ant colony optimization* (MARCO DORIGO, MAURO BIRATTARI, 2006), *swarm optimization* (EBERHART; KENNEDY, 1995; KENNEDY; EBERHART, 1997; MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007) e procedimento de busca guloso, aleatório e adaptativo (*GRASP*) (SILVÉRIO; RODRIGUES; STEINER, 2013).

1.1.4.1 Metaheurística

O termo heurístico provê do grego *heuriskein* = descobrir, do mesmo radical que deu origem a palavra *heureca*, imortalizada pelo matemático e filósofo grego Arquimedes. Uma heurística é um procedimento algorítmico desenvolvido através de um modelo cognitivo, usualmente através de regras baseadas na experiência dos desenvolvedores (CORDENONSI; MÜLLER; DE, 2007).

Como dito anteriormente, para alguns problemas seriam necessário anos para encontrar a melhor solução em meio a todas as possíveis. Por isso, enquanto métodos exatos buscam todas as possíveis soluções para um problema, ou realizam combinações para encontrar a melhor solução possível, as heurísticas utilizam estratégias, procedimentos e métodos aproximativos, visando encontrar soluções boas, não necessariamente a melhor, dentro de um tempo computacional aceitável.

Dorigo e Birattari (MARCO DORIGO, MAURO BIRATTARI, 2006) dizem que metaheurística é um conjunto de conceitos algorítmicos que podem ser usadas para definir métodos heurísticos aplicáveis a um vasto conjunto de problemas diferentes. Outra definição é dada em (CHRISTIANE; ZAPELINI, 2009) que diz que metaheurística é um conjunto de conceitos que pode ser utilizado para definir métodos não exatos (heurísticos) aplicáveis a uma ampla gama de problemas diversos. Em outras palavras, uma metaheurística pode ser vista como uma estrutura algorítmica geral que pode ser empregada na resolução de diferentes problemas de otimização, com um número relativamente reduzido de modificações que a adaptem para o tratamento de cada problema específico.

A busca pela solução ótima se torna mais complexa à medida que aumenta o tamanho do espaço de busca do problema. Espaço de busca ou espaço de estados é definido como um conjunto S de estados e por um conjunto A de ações que mapeiam um estado em outro.

O uso de metaheurísticas tem aumentado significativamente a capacidade de encontrar soluções de alta qualidade para os problemas de otimização combinatória, que possuem um espaço de busca muito grande, em um tempo razoável.

Em geral, tais algoritmos não garantem encontrar a solução ótima, porém, avaliações estatísticas mostram que eles têm alcançado com regularidade a finalidade de retornar uma solução de boa qualidade em tempo computacional aceitável (SILVÉRIO; RODRIGUES; STEINER, 2013).

Blum (GENDREAU; POTVIN, 2005) descreve as propriedades fundamentais que caracterizam uma metaheurística:

1. Metaheurísticas são estratégias que “guiam” o processo de busca.
2. O objetivo é explorar o espaço de busca de forma eficiente a fim de encontrar a solução ótima ou uma próxima.
3. Técnicas que constituem metaheurísticas vão de simples procedimentos de buscas locais a complexos processos de aprendizado.
4. Metaheurísticas são aproximadas e normalmente não são determinísticas.
5. Algumas vezes incorporam mecanismos para evitar ficar estagnado em áreas confinadas do espaço de busca.
6. Metaheurísticas não são específicas para um determinado problema.
7. Metaheurísticas podem usar o conhecimento específico de domínio sob a forma de heurísticas que são controlados por estratégias de nível superior.

Hoje as mais avançadas metaheurísticas usam a experiência de pesquisa (incorporada como uma forma de memória) para guiar a busca.
Em suma, podemos dizer que as metaheurísticas são estratégias de alto nível para explorar espaços de busca usando métodos diferentes.

Ao longo dos últimos anos, grande parte do esforço de pesquisa concentrou-se no desenvolvimento de metaheurísticas, usando principalmente dois princípios: busca local e busca populacional.

A busca local é um método onde a exploração intensa do espaço de soluções é executada, movendo a cada passo de uma solução corrente para outra solução promissora na vizinhança. *Simulated Annealing*, Busca Tabu e o Variable Neighbourhood Search (VNS) são os mais famosos métodos de busca local (HERTZ; WIDMER, 2003).

A busca populacional consiste em manter uma gama de boas soluções e combiná-las, a fim de produzir soluções ainda melhores (É o que se espera). Exemplos clássicos são algoritmos genéticos (HERTZ; WIDMER, 2003).

1.1.4.2 Metaheurísticas Populacionais e Bio-inspiradas

As técnicas computacionais que são hoje denominadas Computação Evolutiva e Metaheurísticas se desenvolveram, de maneira relativamente independente, durante os últimos 40 anos do século XX, entre duas comunidades científicas que mantiveram relativamente pouco contato ao longo desse período. Durante esse tempo, ambos os conjuntos de técnicas se consolidaram, sendo hoje reconhecidos como parte integrante do repertório fundamental de ferramentas da Computação e da Engenharia que possibilitam a síntese de muitos dos sistemas tecnológicos hoje existentes. Apenas no decorrer da última década do século XX se formou, nas respectivas comunidades científicas, uma consciência das conexões existentes entre esses dois corpos de conhecimento, que partilham muitos dos seus princípios e fundamentos (TAKAHASHI, 2013).

Métodos de buscas populacionais são técnicas de soluções que tem um grupo de indivíduos inicialmente e faz com que eles sejam moldados de acordo com algumas regras previamente estabelecidas. Normalmente a cada iteração, períodos de auto adaptação são intercalados com períodos de cooperação (HERTZ; WIDMER, 2003).

Uma metaheurística populacional inicia o processo de otimização com um conjunto de soluções, denominado população inicial, sendo que cada indivíduo representa uma solução viável para o problema. Iterativamente ela gera novos indivíduos e troca a população corrente por uma nova população de soluções (TALBI; POPULACIONAIS, 2014). As regras previamente estabelecidas são a base da evolução desses indivíduos, são elas que determinam as alterações que os mesmos sofrerão ao longo da existência.

Existem duas formas de evolução dos indivíduos que fazem com que os mesmos sejam alterados, são os períodos de auto adaptação e os períodos de cooperação. A auto adaptação significa que os indivíduos evoluem de forma independente, enquanto a cooperação implica uma troca de informações entre os indivíduos. Muitos algoritmos diferentes podem ser descritos por essa abordagem, por exemplo, os operadores de seleção e de cruzamento dos algoritmos genéticos podem ser visto como procedimentos do período

de cooperação, enquanto o operador de mutação pode ser visto como procedimento do período de auto adaptação (HERTZ; WIDMER, 2003).

Existem diversas metaheurísticas bio-inspiradas, algumas, como o Algoritmo Genético (ALGORITHMS, 2005) e a Colônia de formigas (MARCO DORIGO, MAURO BIRATTARI, 2006), são mais conhecidas devido a eficiência em problemas complexos como os de otimização combinatória.

Com base nesses delineamentos, o presente trabalho propõe uma nova metaheurística bio-inspirada populacional baseada no comportamento da videira. A inspiração natural para a criação do algoritmo, a analise e discussão dos principais passos seguem logo após uma breve introdução biológica.

1.2 Aspectos biológicos

Assim como no nível computacional, para uma completa compreensão do algoritmo, incluindo sua estrutura funcional e seus aspectos específicos, faz-se necessário uma abordagem da estrutura biológica envolvida no desenvolvimento, como o conhecimento do contexto, conceitos básicos e os fundamentos biológicos para a criação da metaheurística.

1.2.1 Contextualização

A videira é uma trepadeira pertencente a família das vitáceas, cujo fruto é a uva, o seu desenvolvimento é determinado principalmente pelos fatores ambientais onde ela se encontra.

Os registros arqueológicos sugerem que o cultivo domestico da videira, *Vitis vinifera*, começou 6.000-8.000 anos atrás, a partir de progenitores selvagens. Os milhares de cultivares de uva em uso hoje tem sido, desde então, gerados por propagação vegetativa, principalmente por enxertia, e cruzamentos (MYLES et al., 2011).

1.2.2 Características específicas

A videira possui uma capacidade natural adaptativa que fez com que suportasse os mais diversos climas e fosse alterada pelos mesmos, gerando variações a partir de um ancestral comum, porém totalmente diferentes, como observado no caso de espécies como a *Eurasian* cujos frutos descendentes, em contrastes, podem ser grandes ou pequenos; esféricos ou alongados; de quase qualquer cor no espectro visível; e com quantidades variadas e infinitas combinações de açúcares, ácidos e uma série de outros compostos químicos (MCGOVERN, 2003). Muitas das diversas espécies recentes são o resultado dos estudos que possibilitam a escolha de características que são desejáveis na propagação desses.

Baseando-se nessa capacidade e nas influências exercidas do ambiente sobre ela, diversas pesquisas foram realizadas visando entender melhor como se dá o desenvolvimento e seu relacionamento com esses fatores ambientais. Foram criadas práticas agrícolas e métodos que moldem a produtividade e qualidade da videira, e foram elaborados modelos de desenvolvimento que mapeiam esses fatores e como eles direcionam e influenciam o crescimento da planta e consequentemente a sua produção.

1.2.3 Conceitos básicos

Alguns conceitos são importantes para a compreensão da composição e com relação à organogênese natural da videira, alguns desses são listados e definidos a seguir (WESTERKAMP, 2012):

- Gavinhas são apêndices filiformes, por meio dos quais as plantas se ligam entre si, ou a corpos vizinhos. Elas oferecem suporte para o crescimento da planta.
- Clusters ou cachos são conjuntos de frutos, unidos em torno de uma mesma haste. São os resultados ou produtos gerados pela planta.
- Folhas são órgãos das plantas, especializados na captação de luz e trocas gasosas com a atmosfera, para realização da fotossíntese, transpiração e respiração. As folhas fornecem alimento e é um dos principais fatores no desenvolvimento da planta.
- Nó é o ponto de inserção de uma folha, às vezes engrossado.
- Entrenó é a região caulinar entre dois nós consecutivos, também conhecido como inter-nó.
- Gema é o ramo apical(da ponta) ou lateral, com os órgãos ainda pequenos, às vezes envolvido por órgãos protetores.
- Fitômeros são unidades morfogênicas constituídas de um nó, com sua folha, o entrenó abaixo e a gema. São os responsáveis pelo que chamamos de crescimento modular, ou seja, a planta cresce em várias direções e indefinidamente.

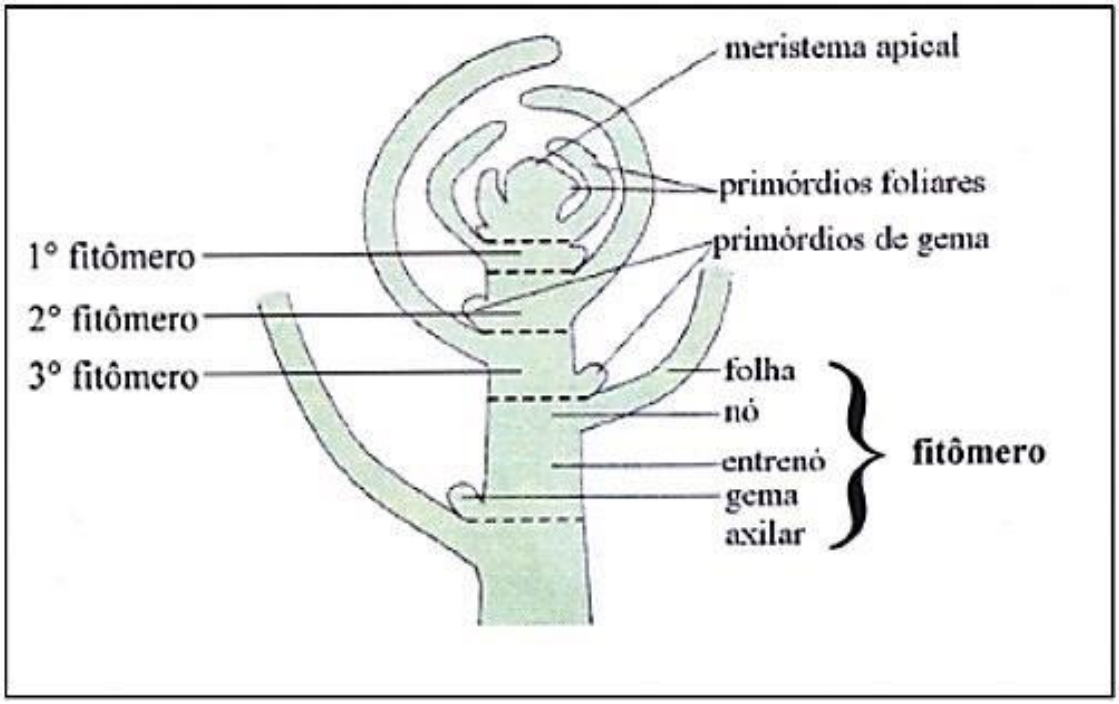

Figura 2 - Estrutura básica de um fitômero(DULCE; SUCENA, [s.d.])**.**

A **Figura 2** apresenta a estrutura de um fitômero, ele é composto por um entrenó, uma gema, um nó e uma folha, podem ainda possuir uma gavinha ou um cacho.

Pallas (PALLAS et al., 2010) propõe que o sistema modular de ramificação possui um eixo principal e muitos eixos secundários organizados numa estrutura de três tipos de fitômeros sucessivos (P0, P1 e P2). P0 fitômeros sem qualquer gavinha ou cacho. Fitômeros com um cacho ou gavinha são classificados como P1 ou P2, dependendo das suas posições com respeito ao fitômero P0 anterior.

A videira é composta por essas unidades morfogênicas de forma que elas se repetem por todo o vegetal, esse esquema permite um crescimento em todas as direções e facilita a sua adaptação ao ambiente, além de proporcionar um sistema de independência parcial, onde uma falha genética ou problema em uma unidade não significa o mesmo problema nas outras.

DISCUSSÃO

1.3 Desenvolvimento Natural

A videira por possuir um sistema modular de ramificação tem um crescimento com independência parcial, isso proporciona o desenvolvimento de cada fitômero de modo único de acordo com o ambiente e a situação em que o mesmo se encontra. Os fatores ambientais, como a temperatura do local onde o fitômero se encontra, qual a melhor localização para crescimento, etc., influenciam no desenvolvimento da videira como um todo, já que os ramos se espalham peço espaço de acordo com esse desenvolvimento e é ele que determina a estrutura da videira como um todo.

Segundo Pallas (PALLAS et al., 2010) o desenvolvimento da Videira é determinado principalmente pelos fatores ambientais, como temperatura e outros abióticos, cujos efeitos são modulados pela sua estrutura topológica complexa. Esses mais diversos fatores são responsáveis por delinear o crescimento da videira, alguns possuem uma maior e outros uma menor influência. Os fatores podem ser naturais (situações ou condições impostas pelo ambiente) ou artificiais (proporcionados pela intervenção humana), a seguir são abordados alguns desses elementos e como funcionam a sua ação sobre a organogênese da videira.

1.3.1 Temperatura

A temperatura determina o potencial de organogênese dos eixos, a competição trófica modula seu desenvolvimento (PALLAS et al., 2008, 2010). A temperatura do ar interfere na atividade fotossintética das plantas. As reações da fotossíntese são menos intensas em temperaturas inferiores a 20ºC, crescem com aumento desse parâmetro climático, atingindo o máximo entre 25 e 30ºC, voltando a cair quando aproxima-se de 45ºC. Os limites de resistência situam-se entre 38 e 50º C. A faixa de temperatura média considerada ideal para a produção de uvas de mesa situa-se entre 20 e 30ºC (TEIXEIRA; ANTONIO HERIBERTO DE C., 2004).

A influência da temperatura ocorre em todos os estádios fenológicos da videira, ou seja, desde o repouso vegetativo, brotação, floração, frutificação, crescimento das bagas, maturação, até a queda das folhas (MANDELLI, 2009).

Cada estádio fenológico necessita de uma quantidade adequada de luz, água e calor para que a videira possa se desenvolver e produzir uvas de qualidade, pois a uva é uma cultura que sofre muita influência dos fatores climáticos, porém pode se adaptar a diferentes tipos de clima (NILSON, 2010). A variação da temperatura ideal, de cada estágio, para a temperatura ambiente proporciona diversas variações na planta, e a intensidade dessas variações ocorrem proporcionalmente a essa diferença.

Os ciclos fenológicos, bem como as temperaturas médias apresentadas, usados como base são uma média dos apresentados por Sellmer em (NILSON, 2010), e são apresentados na **Tabela 1** como foram utilizados:

Ciclo Fenológico	Temperatura
Repouso Vegetativo	10
Brotação	16,5
Floração - Frutificação	19
Maturação e Colheita	27,5

Tabela 1 – Ciclos Fenológicos e temperaturas médias usadas

1.3.2 Enxerto

A enxertia é a união dos tecidos de duas plantas de diferentes espécies, passando a formar uma planta com duas partes: o enxerto e o porta-enxerto (DA SILVA et al., 2015). Uma das maiores vantagens do enxerto é a capacidade de reunir características variadas numa só planta, mas ele ainda possibilita a resistência a certas enfermidades e pragas, pode modificar o porte das plantas, restaurá-las e ainda assegurar a criação de novas variedades. É válido lembrar que nem sempre um enxerto é bem sucedido e produz o efeito desejado. O enxerto consiste basicamente em realizar um corte em uma parte de uma planta que será o enxerto, fazer outro corte de iguais dimensões no porta-enxerto, encaixar a parte no galho receptor, com a gema virada para cima para possibilitar o desenvolvimento. Em seguida, se faz a amarração para evitar a entrada de ar e água. Deve-se retirar todos os brotos do porta-enxerto para ele não se sobrepor à enxertia.

A combinação de duas espécies é de grande utilidade, porém para a construção de novas espécies são necessários vários testes, pois é possível que as características herdadas não sejam exatamente as esperadas e em cada enxerto o resultado é diferente, logo para o alcance dos melhores resultados deverão ser feitas análises para seleção dos mais promissores e que de alguma forma melhoraram em relação a espécie inicial.

1.3.3 Competição Trófica (Torneio)

A competição trófica é a disputa dos fitômeros pelos nutrientes oferecidos e ela ocorre durante o crescimento da planta, modulando o desenvolvimento da mesma.
Um aumento na competição trófica tende a ter o mesmo efeito no desenvolvimento da planta que um déficit hídrico do solo, pois o déficit reduz a atividade de fotossíntese da planta (PALLAS et al., 2010). Sendo assim a competição trófica e a variação da quantidade de água possuem o mesmo efeito no desenvolvimento que é a modulação do desenvolvimento de cada fitômero.

A atividade fotossintética gera o alimento para a planta, a distribuição do mesmo para os fitômeros ocorre de forma que os que ocupam locais mais promissores recebam maior quantidade, porém não deixando que os menos promissores morram por inanição de qualquer forma. Apesar de muitos não receberem esse alimento por estarem em locais menos adequados para o crescimento, outros não são completamente descartados. Isso é importante, pois algumas vezes, com o tempo, esse apoio leva a videira a fixar suas gavinhas em locais que proporcionem mais luz, umidade, melhor temperatura, etc... A distribuição na videira leva em consideração diversos fatores, como a temperatura do local, posicionamento em relação a apoio das gavinhas, recepção de luz e outros. Os fitômeros vizinhos recebem uma quantidade de seiva parecida, dado as condições próximas em que se encontram.

Entre os diversos elementos - naturais e humanos - que alteram a organogênese da videira, alguns são implementados nessa metaheurística, para essa escolha, foram consideradas as influências exercidas sobre a mesma em um plano geral. Os fatores considerados para a idealização e o desenvolvimento foram:

1. O principal elemento natural, a Temperatura.
2. O principal elemento artificial ou fator humano, que é o Enxerto.
3. O método de seleção que simula a competição trófica, o Torneio.

A decisão de quais os fatores influenciadores que deveriam ser implementados inicialmente se deu com base na análise do efeito proporcionado no processo depois de adaptado para formato algorítmico e na maior influência real na organogênese da videira. Para a seleção destes fatores, foram pesquisadas as influências ambientais para o desenvolvimento da Videira. Por exemplo, a temperatura determina o potencial de organogênese dos eixos, a Competição Trófica modula desenvolvimento da Videira, o Déficit Hídrico do solo reduz a atividade de fotossíntese da planta e, desta forma, tende a ter o mesmo efeito no desenvolvimento da planta como um aumento na Competição Trófica. Assim, a Competição Trófica e a Variação da Quantidade de Água possuem, basicamente, o mesmo efeito no desenvolvimento, que é a modulação do desenvolvimento de cada fitômero. Portanto, de forma geral, a Temperatura e a Competição Trófica resumem o comportamento da planta como um todo. Então o efeito simulado é o comportamento de videiras que estejam bem regadas. A competição trófica em cultivares com essas condições são suficientes para alterar o crescimento, como já analisado em (PALLAS et al., 2008). Além deles, um fator que produz um efeito muito interessante no comportamento da videira é o enxerto, por ser capaz de modular o desenvolvimento da planta através de intervenções humanas, obtendo assim um produto final mais próximo do desejado.

1.4 Desenvolvimento da Metaheurística

O **MHV** tem como foco a resolução de problemas de otimização em um determinado espaço de busca, e assim como o crescimento da planta é alterado pelos fatores iniciais e pelas influências exercidas no seu desenvolvimento, os parâmetros iniciais e alterações ao

decorrer do desenvolvimento, moldam a videira pelo espaço de busca e as soluções encontradas, podendo se traçar um padrão de desenvolvimento e melhores circunstâncias para produção e crescimento de acordo com o ambiente e as condições no meio em que ela se encontra.

A metaheurística tem como fundamento o mapeamento da organogênese da videira. O modo como ela caminha pelo espaço de busca será baseado nesse comportamento, sendo que os fatores ambientais alterarão o crescimento e o direcionamento da mesma, ela crescerá em várias direções, porém moldada pelo ambiente. As gavinhas são o que mantém a busca em um espaço promissor. Os fitômeros são as soluções no espaço de busca e são os locais onde as funções que alteram o crescimento são aplicadas. Além de todos os fatores naturais que alteram o desenvolvimento e consequentemente a produção dos frutos, existem fatores artificiais ou humanos que alteram o desenvolvimento da planta como a poda, adubação, irrigação, etc.

O principal elemento natural que altera a forma como a videira se desloca no espaço de busca, que é a Temperatura. O principal elemento artificial ou fator humano que alteram a forma como a videira se desloca no espaço de busca, que é o Enxerto. O método de seleção aplicado sobre os fitômeros, o Torneio, que simula a Competição Trófica.

1.4.1 Temperatura

A temperatura ambiente é um parâmetro inicial que simula a variação de temperatura do local onde a videira está inserida, essa temperatura varia de acordo com uma faixa de valores determinado inicialmente. A influência da temperatura sobre a videira ocorre de forma única em cada fitômero, pois o crescimento modular possibilita isso, as alterações ocorrem de acordo com uma taxa de variação da temperatura.

$$\Delta T = \frac{|t - t_i|}{t}$$

Equação 1 - Taxa de Variação da Temperatura

A taxa de variação (ΔT) que faz com que os fitômeros sofram alterações é obtida pelo módulo da diferença entre a Temperatura atual (t) e a Temperatura ideal (t_i), em seguida esse valor é dividido pela Temperatura atual (t) (Como é apresentado na **Equação 1**).

A Temperatura ideal (t_i) obtida a partir do quadro, de acordo com o ciclo atual em que a videira se encontra, considera-se como se cada ciclo fenológico tivesse a duração igual, sendo assim cada ciclo dura $1/4$ do número máximo de interações determinado como parâmetro do algoritmo. Ou seja, da iteração inicial até $1/4$ do número máximo de iterações o ciclo é o Repouso Vegetativo, de $1/4$ a $2/4$ é a Brotação, de $2/4$ a $3/4$ Floração – Frutificação e de $3/4$ até a última iteração Maturação e Colheita.

A probabilidade da posição do fitômero sofrer alteração é essa taxa que varia entre 0 e 1, no caso de diferenças muito grandes a probabilidade de que ocorra uma alteração no desenvolvimento é maior. O método pode em alguns casos se assemelhar, em partes, ao comportamento do *Simulated Annealing* (BERTSIMAS; TSITSIKLIS, 1993), quando a diferença é maior no início e menor no fim, ocorre uma maior alteração das soluções no início, possibilitando uma busca mais global e no ultimo ciclo fenológico uma convergência para buscas locais.

1.4.2 Enxerto

O enxerto na metaheurística, assim como na realidade, possibilita aos fitômeros a obtenção de características que são favoráveis a um melhor desenvolvimento e no algoritmo consequentemente soluções diferentes o que pode levar a sair de uma estagnação momentânea.

O método consiste basicamente em realizar um corte em uma parte do fitômero, que será o porta-enxerto ou receptor, e outro corte de iguais dimensões em outro fitômero, que será o enxerto. Assim como na prática, o enxerto muitas vezes pode não funcionar como se espera. Como é possível que as características herdadas não sejam exatamente as esperadas, dado que cada enxerto gera um resultado diferente, para o alcance dos melhores resultados utilizam-se os fitômeros mais promissores em que as características desejadas são mais presentes, no caso os fitômeros que possuam um valor *q* melhor do que o receptor.

É comum se escolher aleatoriamente um dos *n* melhores fitômeros para a realização do enxerto. Outra estratégia que pode ser adotada é a aplicação de uma busca local em um fitômero, e a partir desse realizar o enxerto.

A probabilidade que define o tamanho do enxerto normalmente é: $1 - q(Receptor)/q(Enxerto)$. Essa probabilidade faz com que quanto maior a diferença entre o q do enxerto e o q do receptor, maior é a alteração sofrida, de forma que quanto melhor o receptor, menos correções serão feitas na sua estrutura e quanto pior maiores as alterações.
Existe ainda uma probabilidade de o enxerto ser realizado, visto que na realidade o enxerto não é realizado a todo o momento de forma arbitrária. A probabilidade utilizada normalmente gira em torno de 0 a 0.1, ou seja a possibilidade de ocorrer um enxerto em cada iteração é essa probabilidade definida inicialmente.

1.4.3 Competição Trófica (Torneio)

O método de seleção tem como intenção escolher os melhores fitômeros para o desenvolvimento da videira. A estratégia de seleção leva às melhores soluções. Entretanto, fitômeros piores não devem ser descartados e eles devem ter uma chance de ser selecionados. Os fitômeros selecionados são aqueles que possuem o melhor *q*, ou seja,

esses receberão maior quantidade de alimento. Os que não se encontram em um ambiente promissor possuem a possibilidade de serem escolhidos, porém muitos são descartados.

A seleção por torneio consiste em selecionar aleatoriamente *n* fitômeros. O parâmetro *n* é o tamanho do grupo do torneio. É aplicado o torneio aos *n* membros do grupo para selecionar o melhor. Para selecionar *x* fitômeros, o procedimento deve ser repetido *x* vezes.

1.5 Algoritmo

O algoritmo **MHV** é iniciado com um número *n* de fitômeros, o desenvolvimento da videira é direcionado pelos parâmetros que alteram a direção ao longo das iterações de índice *i* sobre a superfície da função objetivo em estudo. O conjunto de fitômeros é a videira e cada fitômero representa uma possível solução para a função matemática que se deseja encontrar o ponto ótimo, sendo *x** a melhor posição/solução na iteração *i*. O espaço de busca é o ambiente onde a videira está inserida e cada local do espaço de busca possui uma sequência de parâmetros, esses são alterados de acordo com o passar das iterações, simulando a movimentação da videira pelo espaço (a permanência da videira em determinado local do espaço de busca são as gavinhas de alguns fitômeros).
Inicialmente a videira começa no primeiro ciclo fenológico que é o Repouso Vegetativo e o número de iterações *i/4* é a duração do ciclo atual. A alternância desses ciclos ocorre naturalmente a cada ¼ do número total de iterações.

A temperatura ambiente é também um parâmetro inicial, que simula a média de temperaturas no local onde a videira está sendo plantada. Tal atributo varia a cada iteração i dentro de um limite que também é um parâmetro *l*. É calculado a taxa de variação como apresentado na **Equação 1** entre a temperatura ideal do ciclo que a videira está e a temperatura atual e essa taxa é a probabilidade de cada parte do fitômero ser alterado ou não.

O enxerto é um meio de se obter a combinação de dois fitômeros, um dos melhores fitômeros da videira (enxerto) e um aleatório (porta-enxerto), a intenção do enxerto é o mesmo que o cruzamento aplicado no Algoritmo Genético, que é cruzar os fitômeros para que se obtenham genes interessantes de outros fitômeros e para que assim os receptores possam se tornar mais promissores e redirecionar o campo para um espaço de busca diferente do inicial. O enxerto mescla as características de diferentes fitômeros aleatoriamente de forma proporcional a quão melhor o enxerto é do que o receptor como visto anteriormente, ou seja, soluções diferentes serão geradas a partir desse cruzamento.

O método de seleção determina quais fitômeros permanecerão, de acordo com o espaço de busca mais promissor, esses métodos alinhados com a manipulação da busca no espaço objetivam alcançar o máximo ou mínimo, como proposto pelo problema. O método utilizado é o torneio e pela comparação dos valores de qualidade (*q*) de cada fitômero são decididos quais permanecerão e quais serão descartados do grupo de soluções existentes.

```
Função objetivo f(x), x = (x1x2, ..., xd)^E
Inicializa os ramos da videira x_i (i = 1, 2,..., n)
Define o valor q_i de cada x_i de acordo com o problema
Classifica os fitômeros e localiza o melhor
    Enquanto (N < N° máximo de iterações)
    l = Valor aleatório entre a faixa de alteração da temperatura
    estabelecida
    T_i= Temperatura ideal do Ciclo Fenológico atual
    T = Temperatura ambiente inicial + limite l
        Enquanto (fitômero i!= ultimo)
        Gera novas soluções pela Equação 1
            VideiraFutura ← Equação 1(VideiraAtual)
            Aplica o torneio e decide quais dos valores
            permanece.
            VideiraAtual ← Torneio(VAtual, VFutura)
            Calcula a melhor solução até o momento
        Fim-Enquanto
        Se (valor sorteado < probabilidade de enxerto)
            Pega um dos n melhores fitômeros
            Se (q do fitômero x_n> q do fitômero x_i aleatório))
                Aplica o Enxerto de x_n no receptor x_i
                x_i := Enxerto (x_n , x_i)
                Aplica o Torneio e decide se o fitômero x_i
                gerado substitui o fitômero x_i antigo.
            Fim-Se
        Fim-Se
    Verifica as soluções obtidas e retorna a melhor x*
    Fim-Enquanto
Fim-Algoritmo
```

Algoritmo 1 – Algoritmo **MHV**

Inicialmente desenvolve-se a videira com condições básicas passadas como parâmetros, essa videira gera seus primeiros fitômeros, soluções iniciais, aleatoriamente para o problema em análise. Com as soluções iniciais conhecidas, é calculado a qualidade(q) de cada uma, de acordo com a função objetivo e parâmetros definidos pelo problema em questão. A **Figura 3** apresenta o fluxograma básico do algoritmo.

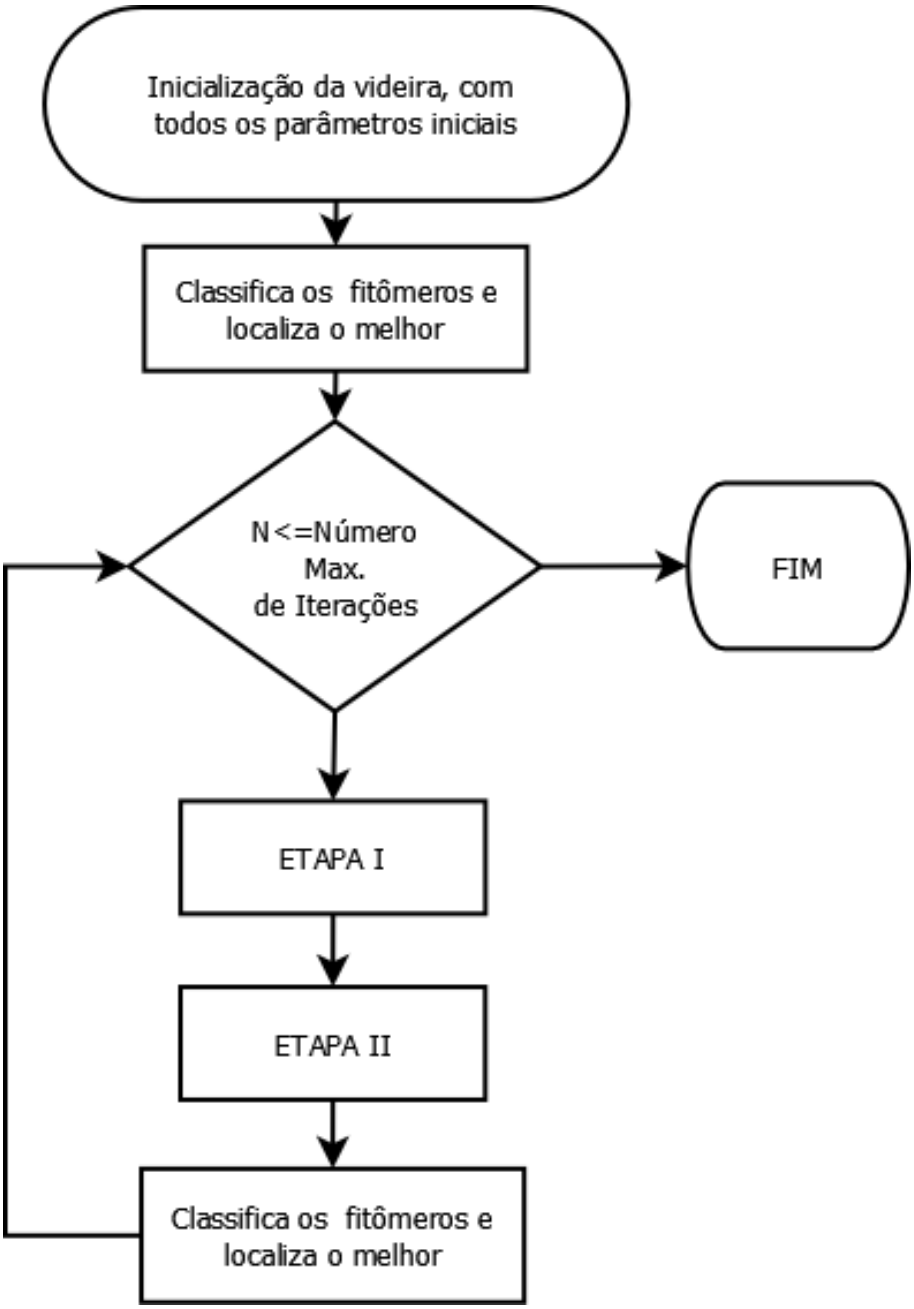

Figura 3 – Fluxograma básico do MHV

A seguir, entra-se no processo de busca, sendo a primeira etapa apresentada no fluxograma abaixo, **Figura 4**. A Etapa I consiste na geração de novos fitômeros pela equação de temperatura (**Equação 1**), a videira após as alterações é chamado de *VideiraFutura,* em seguida será aplicado o método de seleção entre a videira com os fitômeros novos e antigos e são decididos os que permanecerão para a próxima geração de fitômeros, o processo é repetido *N* vezes, até o último fitômero.

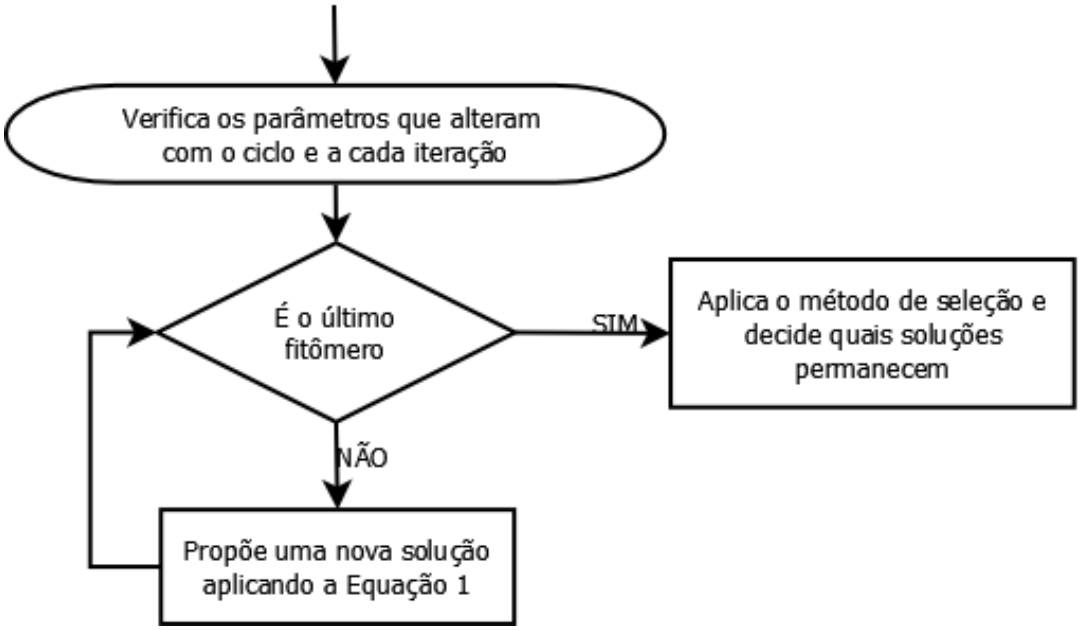

Figura 4 – Fluxograma auxiliar – ETAPA I

A **Figura 5** apresenta a Etapa II do processo e representa a atuação do enxerto onde, de acordo com uma probabilidade pré-estabelecida inicialmente, é selecionado um dos melhores *n* fitômeros com um q_n aleatório e outro fitômero aleatório na videira com q_i. Se o q_n é melhor do que q_i o enxerto é aplicado, com uma probabilidade que indica o quanto aquele fitômero será alterado pelo enxerto, após o fim do enxerto, o método de seleção é aplicado e é decidido se o fitômero que recebeu o enxerto substitui o fitômero antigo.

Depois de encerrado o ciclo com a Etapa I e a Etapa II e a *VideiraAtual* ser completamente definida é novamente calculado a qualidade(q) de cada fitômero, de acordo com a função objetivo e parâmetros definidos pelo problema em questão, é definido o melhor e é verificado se o número máximo de iterações foi alcançado, caso afirmativo o melhor valor é retornado e o algoritmo se encerra, caso negativo os parâmetros são atualizados de acordo com a iteração e novamente será realizado o ciclo com a Etapa I e uma probabilidade baixa da Etapa II acontecer.

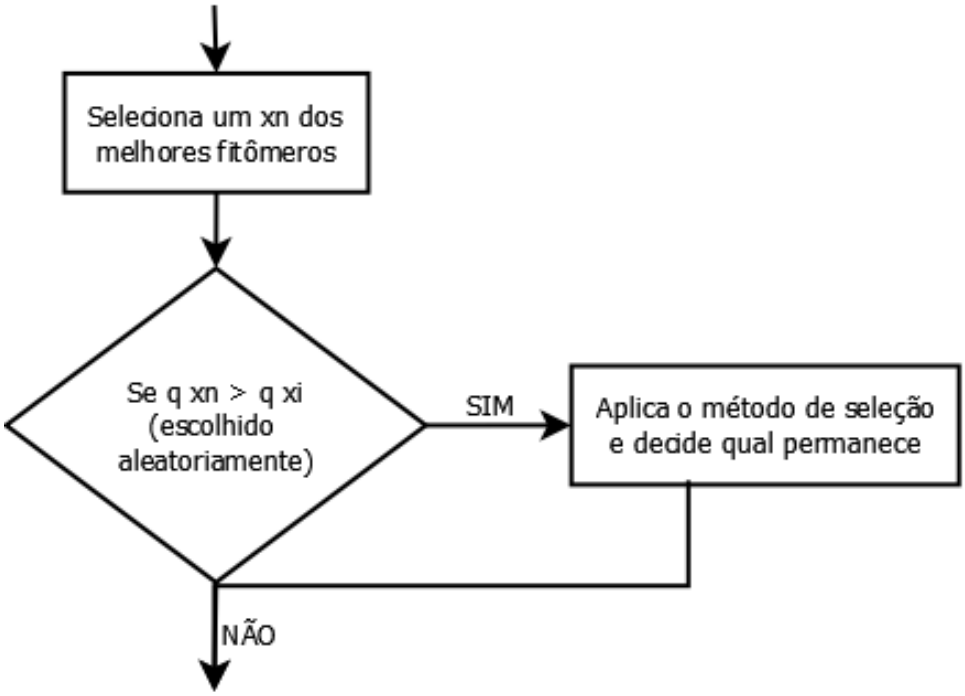

Figura 5 - Fluxograma auxiliar - ETAPA II

1.6 Validação e Comparação

A partir do algoritmo em pseudocódigo e dos fluxogramas apresentados, é relativamente fácil implementar o algoritmo da videira (**MHV**) aqui apresentado em qualquer linguagem de programação. Para os testes, validação e comparação, o mesmo foi implementado em JAVA.

Foram realizados diversos testes para a verificação do funcionamento e para a validação do modelo e algoritmo proposto, para isso foram escolhidas funções de teste que normalmente são utilizadas para testes em metaheurísticas.

1.6.1 Funções de Testes

Existem diversas funções utilizadas para testes e validação de um algoritmo, durante os testes realizados foram utilizadas algumas. Dentre os tipos de funções encontradas na literatura, cada uma traz um desafio diferente, a seguir são apresentadas funções de maximização e minimização que se caracterizam por aplicar cálculos que determinam o valor em cima de uma palavra binária, ou seja, o fitômero é uma palavra binária e as funções são aplicados no fitômero, obtendo-se um valor que deve ser otimizado, sendo esse valor o *q* do fitômero. As funções de maximização utilizam diretamente a palavra binária, enquanto as funções de minimização, por se tratarem de funções que trabalham com valores reais, passam por um processo de conversão, que será explicado mais a frente, onde a

palavra binária é transformada em um número e esse número sim é utilizado para otimização da função.

Das funções de maximização foram utilizadas a função *OneMax,* e uma função enganosa ou armadilha (Deceptive Function ou DecTrap) *TrapFive* (RICHTER; PAXTON, 2005) e das funções de minimização foram utilizadas três clássicas da literatura: *Sphere* (também conhecida como primeira função de De Jong), *Rosenbrock* e *Rastrigin*.

1.6.1.1 Escolha das funções

O processo de escolha das funções para os testes e a validação dessa metaheurística tiveram como base a larga utilização das mesmas na literatura e a dificuldade encontrada para resolução das mesmas. Por exemplo em (LEONARDO RAMOS EMMENDORFER, 2007), (ANDRADE, 2013a) e (GOLDBERG; DEB, 1992), a função *TrapFive* é utilizada para testes com diversos métodos e em todos os trabalhos analisados a função é considerada um obstáculo a ser superado devido as condições por ela imposta. Além dela nesse trabalho é utilizada a função *OneMax* que segue o mesmo padrão, apesar de ser mais simples e já é tradicional nos testes com funções de maximização. As funções de minimização utilizadas são reconhecidas e utilizadas na grande maioria dos testes com metaheurísticas, como em (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007)(AUSTIN, 1987)(YANG, 2010a)(ANMER DASKINA, 2011), geralmente elas são recomendadas para validação e comparação de metaheurísticas.

Existem ainda alguns trabalhos que fazem uma revisão dos problemas de otimização mais comumente utilizados na literatura (JAMIL; YANG, 2013) (YANG, 2010b)(POHLHEIM, 2005).

A função *TrapFive* é uma função armadilha, um conjunto de funções especialmente desenvolvido para verificação da robustez (LEONARDO RAMOS EMMENDORFER, 2007).

Todas as funções utilizadas tinham como intenção verificar a performance geral em um problema, a robustez, a velocidade de convergência e a precisão. E essas análises podem ser vistas de forma bem abrangente principalmente através do comparativo com os outros algoritmos já tradicionais.

1.6.2 Funções de Maximização

As funções de maximização *OneMax* e *TrapFive* têm como objetivo alcançar o máximo global que é igual ao números de 1 no vetor, como o tamanho definido do vetor é 100, esse é o máximo. Ambas funções de maximização não foram comparadas com (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007), visto que elas não são avaliados no trabalho.

1.6.2.1 Função *OneMax*

One Max é uma função que busca o valor máximo de um somatório. Assume-se que, nessa função, N é o número de bits no vetor que será avaliado e x_i representa cada bit desse vetor. Como cada bit só pode assumir os valores 0 e 1, o valor máximo dessa função é N. Segundo Penachin (PENNACHIN, 2010) a complexidade dessa função é = $O\ (N\ log\ N)$. A definição formal é apresentada na **Equação 2**.

$$f(x) = \sum_{i=1}^{N} x_i$$

Equação 2 – Definição Formal Função *OneMax*

1.6.2.2 Função *TrapFive*

Uma função enganosa possui normalmente um ótimo global e um falso ótimo, ou ótimo armadilha, localizado maximamente longe do ótimo global como definido em (GOLDBERG; DEB, 1992). O que torna mais difícil a resolução das *TrapFunctions* é que elas dividem o espaço de busca em duas bacias com espaço de *Hamming*, onde a maior distância existente é entre o ótimo real e o ótimo armadilha. Como mostrado na **Figura 6**, um caso com tamanho 5, a bacia de atração ou armadilha ocupa a maioria do espaço funcional. A função implementada é uma concatenação de diversas armadilhas, conhecida como *TrapFive*.

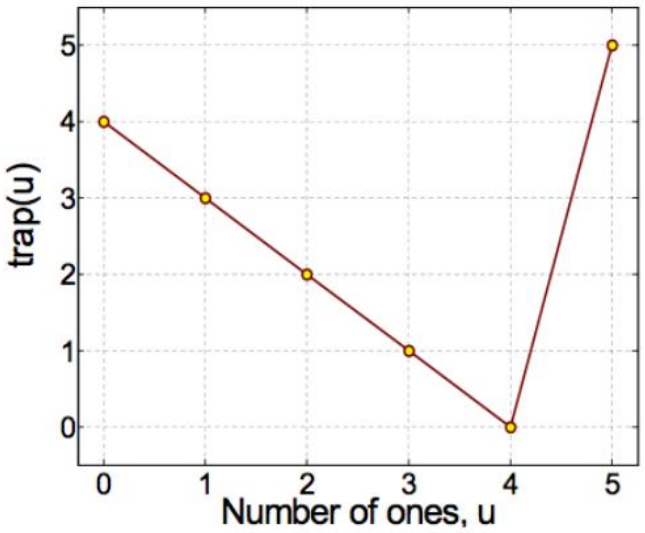

Figura 6 – Exemplo de bacia de atração *TrapFunction(DecTrap)* (PENNACHIN, 2010).

A função *TrapFive* também busca encontrar o valor máximo em um somatório. A diferença entre esta e a função anterior é que aqui o vetor é dividido em blocos de cinco e a soma dos blocos é calculada de acordo com regra de condição abaixo (**Equação 3**). Como os blocos não se sobrepõem e as contribuições de cada bloco para o valor da função total se somam, estes problemas são chamados de aditivamente decomponíveis (LEONARDO RAMOS EMMENDORFER, 2007).

Em (AZIMI; JALALI; FERN, 2011) é dito que a dificuldade mais frequentemente encontrada em problemas *benchmark* é quando um problema é enganoso, como o *TrapFive*. Problemas são enganosos quando estatísticas de primeira ordem levam na direção de um bloco sub-ótimo, mas o bloco ótimo correspondente está na direção contrária.

Uma função enganosa concatenada como essa determina a contribuição de cada bloco para o valor total da função de um indivíduo. No caso o valor máximo que a função pode assumir também é N, onde N é o número de bits no vetor e x_i representa cada bit desse vetor. De acordo com (PENNACHIN, 2010) a complexidade da função é = O (2N).
A definição formal é apresentada na **Equação 3** e a Regra de condição para a mesma é apresentado na **Equação 4**.

$$f(x) = \sum_{i=1}^{N/5} TrapFive(x_{5i} + x_{5i+1} + \cdots + x_{5i+4})$$

Equação 3 – Definição formal Função *TrapFive*.

$$TrapFive(x_1, \ldots x_5) = \begin{cases} 5 & se\ e \sum x_i = 5 \\ 4 - \sum x_i & caso\ contrário \end{cases}$$

Equação 4 – Regra de condição Função *TrapFive*.

1.6.3 Funções de Minimização

Nas funções de minimização o objetivo é chegar aos valores mais próximos de 0 (zero), que é o mínimo de todas as funções utilizadas no presente trabalho. Devido a utilização um vetor de valores binários, fez-se necessário o tratamento dos dados desse, para então submetê-los à avaliação.

Para a conversão do valor binário em real, foi realizado previamente uma conversão em inteiro, utilizando-se o bit da esquerda como mais significativo. E então a partir do valor inteiro, a aplicação da **Equação 5** concluiu a conversão para real.

$$a_i + k.\frac{b_i - a_i}{2^n}$$

Equação 5 – Função de conversão binário-real (WRIGHT, 1990).

Na **Equação 5**, *k* é a representação em inteiro do número binário e, para de manter a coerência com (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007), utilizou-se n = 20, isto é, 20 bits para representar cada número e a_i = -50 e b_i = 50, como limite inferior e superior, respectivamente.

Para exemplificar este processo de conversão, suponha que para resolver uma função de dimensão N = 3, em determinada iteração do **MHV**, por exemplo, o fitômero 0 apresente o seguinte genótipo hipotético, representado por um vetor de 60 bits, como exemplificado na **Figura 7** (ANDRADE, 2013a):

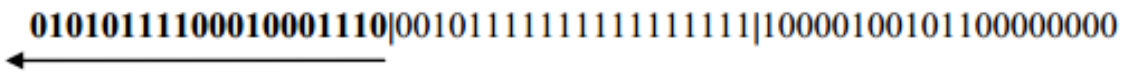

Figura 7 – Exemplo de fitômero para conversão(ANDRADE, 2013a).

Neste caso, cada grupo de 20 bits é convertido para um número real, que será usado em uma dimensão do problema, da esquerda para a direita, respectivamente. Dessa forma, o primeiro grupo de 20 bits (à esquerda) convertido em inteiro pela fórmula que utiliza o bit da esquerda como mais significativo é igual a 358.542 ($0 * 2^0 + 1 * 2^1 + 1 * 2^2 + 1 * 2^3 + \ldots + 0 * 2^{19}$). Então, pela **Equação 5** temos que o número real resultante é $= (-50) + 358.542 \times (100/1048576) \approx -15{,}8067$. Para todas as funções de minimização, *N* representa a dimensão do espaço de busca e x_i é a representação em números reais de um bloco de 20 bits.

A seguir é apresentada uma breve descrição da função, bem como a descrição formal da mesma.

1.6.3.1 Função *Sphere*

Também conhecida como *primeira função de De Jong*, a função *Sphere* é uma referência na literatura para testes de metaheurísticas, caracteriza-se por ser continua, convexa e unimodal (MOLGA; SMUTNICKI, 2005). O mínimo global está localizado $f * = 0\ em\ (0{,}0{,}\ldots{,}0)$ (YANG, 2010b). A definição formal é apresentada na **Equação 6.**

$$f(x) = \sum_{i=1}^{N} {x_i}^2$$

Equação 6 – Definição formal Função *Sphere*.

A **Figura 8** apresenta o gráfico da função *Sphere* usando um domínio das variáveis, outros domínios apresentam gráficos semelhantes, apenas com a escala diferente.

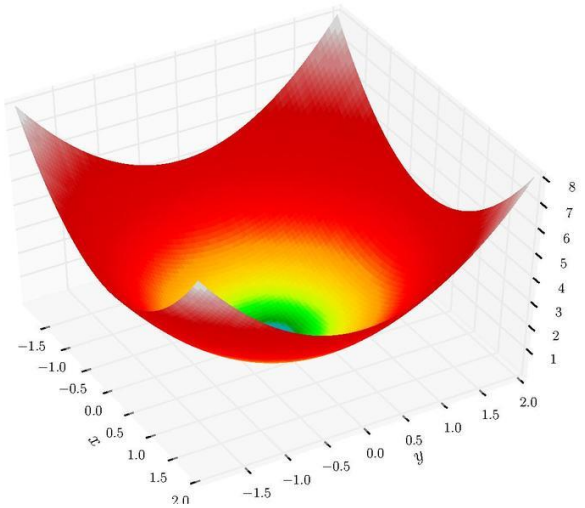

Figura 8 – Exemplo de visualização gráfica da função *Sphere* (MOLGA; SMUTNICKI, 2005).

1.6.3.2 Função *Rosenbrock*

A função *Rosenbrock* é um problema clássico de otimização, também conhecida como função banana. O ótimo global fica situado dentro de um longo e estreito vale plano em forma parabólica, como pode ser visto na **Figura 9**. Encontrar o vale é trivial, no entanto convergência para o ótimo global é difícil e, portanto, este problema tem sido repetidamente usado na avaliação do desempenho de algoritmos de otimização (POHLHEIM, 2005). A definição da função:

$$f(x) = \sum_{i=1}^{N-1} (100(x_{i+1} - {x_i}^2)^2 + (x_i - 1)^2)$$

Equação 7 – Definição formal Função *Rosenbrock*

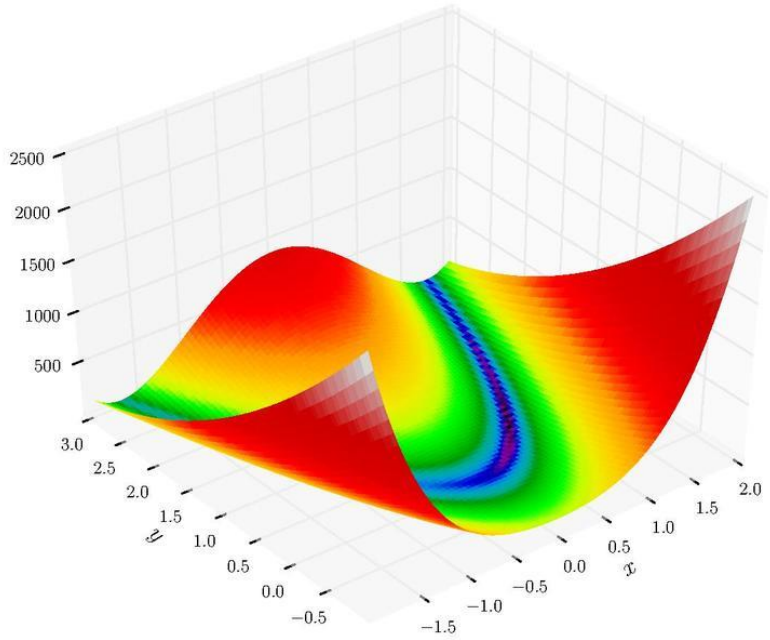

Figura 9 – Exemplo de visualização gráfica da função *Rosenbrock*(MOLGA; SMUTNICKI, 2005).

1.6.3.3 Função *Rastrigin*

A função *Rastrigin* é baseado na função de *De Jong* com a adição da modulação do cosseno, a fim de produzir mínimos locais frequentes. Assim, a função de teste é altamente multimodal. No entanto, a localização dos mínimos são regularmente distribuídos, como mostra a **Figura 10** (MOLGA; SMUTNICKI, 2005). A função tem a definição formal apresentada na **Equação 8.**

$$f(x) = \sum_{i=1}^{N} ({x_i}^2 - 10\cos(2\pi x_i) + 10)$$

Equação 8 – Definição formal Função *Rastrigin*

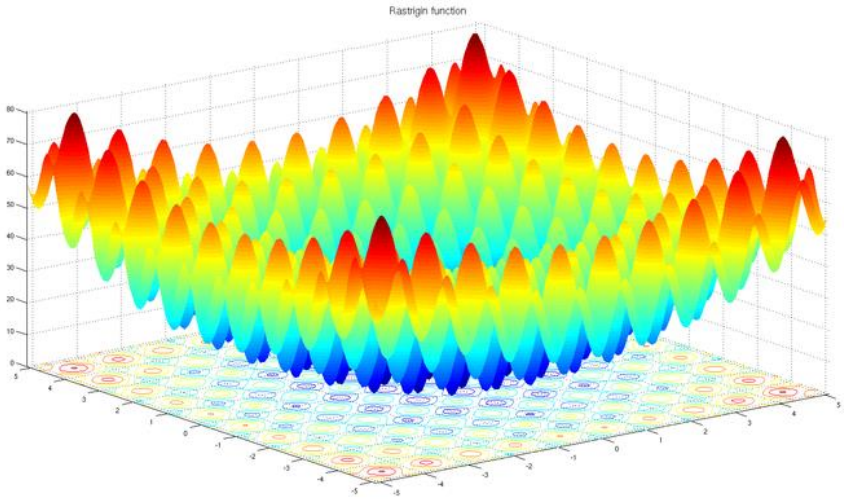

Figura 10 – Exemplo de visualização gráfica da função Rastrigin (MOLGA; SMUTNICKI, 2005).

1.6.4 Técnicas Comparadas

A fim de comparar o desempenho do algoritmo **MHV**, após os testes, os resultados obtidos foram comparados com os obtidos por outros quatro algoritmos heurísticos já tradicionais, o Algoritmo Genético (AG) (MITCHELL, 1995)(GOLDBERG; DEB, 1992), o Otimização por Enxame de Partículas (PSO) (KENNEDY; EBERHART, 1997) (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007), o Population-Based Incremental (PBIL) (BALUJA; CARUANA, 1995) e o Otimização por acasalamento de abelhas (MBO)(ABBASS, 2001). Os resultados utilizados na comparação, obtidos pelo AG, PSO, PBIL e MBO, são os de Andrade Jr em (ANDRADE, 2013a).

Os próximos tópicos farão uma breve introdução de cada metaheurística utilizada para comparação no presente trabalho, devido o foco não ser explicar detalhadamente cada metaheurística, a descrição é bem básica e portanto para uma maior compreensão aconselha-se uma consulta aos artigos referenciados. Os parâmetros específicos de cada metaheurística é apresentado na respectiva sessão onde cada um é descrito, porém alguns parâmetros são comuns a todos os métodos e para realizar a comparação do desempenho dos algoritmos, definiu-se que:

O número de iterações em cada execução seria de 1000 e em cada iteração seria gerado um conjunto de 100 soluções, o que resulta num total de 100.000 soluções avaliadas a cada execução. Os resultados foram coletados a partir de 10 execuções consecutivas de cada algoritmo.

O **MHV** operou com os parâmetros apresentados na **Tabela 2**:

Torneio	10 fitômeros
Temperatura	24
Limites (Máximo e Mínimo)	3 e -3
Enxerto	0.01

Tabela 2 – Parâmetros MHV

1.6.4.1 Algoritmo Genético

Os Algoritmos Genéticos (AG) (MITCHELL, 1995)(GOLDBERG; DEB, 1992) são metaheurísticas que buscam soluções para problemas de otimização de forma análoga ao processo de evolução natural, utilizando procedimentos iterativos que simulam o processo de evolução de uma população de possíveis soluções para um determinado problema. Essa evolução é guiada por um mecanismo de seleção baseado na adaptação de estruturas individuais. A cada iteração do algoritmo (uma geração), um novo conjunto de estruturas é criado através da troca de informações entre estruturas bem adaptadas selecionadas da geração anterior. O resultado tende a ser um aumento da adaptação de indivíduos ao meio, podendo acarretar também um aumento global da aptidão da população a cada nova geração. Nesse caso, a população evolui a cada geração, aproximando-se de uma solução ótima.

Os parâmetros utilizados no AG são apresentados na **Tabela 3**:

Torneio	8 indivíduos
Crossover	1 ponto
Coeficiente de cruzamento	0.75
Mutação	0.01

Tabela 3 – Parâmetros AG

1.6.4.2 Otimização por enxame de partículas (PSO)

A Otimização por enxame de partículas(PSO) foi originalmente desenvolvido e introduzido por Eberhart e Kennedy (EBERHART; KENNEDY, 1995). O PSO é um algoritmo de busca populacional baseado na simulação do comportamento social dos pássaros, abelhas ou cardumes de peixes. Cada indivíduo dentro do enxame é representado por um vetor no espaço de busca multidimensional. Este vector tem também um vector designado que determina o movimento seguinte da partícula e é chamado o vector de velocidade. O algoritmo PSO também determina como atualizar a velocidade de uma partícula. Cada partícula atualiza sua velocidade baseado na velocidade atual e a melhor posição que tem explorado até agora e também com base na melhor posição global

explorado pelo enxame (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007).

O processo de PSO, em seguida, é iterativo um determinado número de vezes ou até outro critério de parada pré-estabelecido. Demonstrou-se que este modelo simples pode lidar com problemas de optimização difíceis de forma eficiente. PSO foi originalmente desenvolvido para espaços de valores contínuos, mas muitos problemas são definidos para espaços de valores discretos, nos quais o domínio das variáveis é finito. Exemplos clássicos desses problemas são: programação inteira, escalonamento e roteamento.

Em 1997, Kennedy introduziu uma versão binária discreta do PSO para problemas de otimização discreta. Nesse chamado PSO binário (KENNEDY; EBERHART, 1997), cada partícula representa sua posição em valores binários. O valor de cada partícula pode então ser mudado ou sofrer mutação do 1 para 0 e vice versa. Nessa versão binária, a velocidade de uma partícula é definida como a probabilidade que uma partícula pode mudar seu estado para 1 (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007).

Porém, essa versão original do PSO binário mostrou alguns problemas, sendo assim, outros autores propuseram variações para obtenção de uma melhor performance. Aqui utilizaremos a versão proposta por Khanesar et al (MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, 2007), cujo trabalho propôs algumas alterações à versão original de (KENNEDY; EBERHART, 1997), obtendo resultados interessantes que foram utilizados em estudo comparativo.

Os parâmetros utilizados no PSO podem ser vistos na **Tabela 4**:

Peso do melhor estado atual	6
Peso do melhor estado global	5
Peso de inércia	0.05

Tabela 4 – Parâmetros PSO

1.6.4.3 Population-Based Incremental Learning (PBIL)

O PBIL é uma abstração do AG, que mantém explicitamente as estatísticas contidas na população de um AG, mas que abstrai o operador de *crossover* e redefine o papel da população. Isto resulta que o PBIL é mais simples, tanto computacionalmente e, teoricamente, do que o AG (BALUJA; CARUANA, 1995). Basicamente é uma combinação dos mecanismos do AG com o aprendizado incremental.

O objetivo do algoritmo PBIL é criar um vetor de probabilidade de valores reais que, quando amostrado, revela os vetores de solução melhores avaliados com uma alta probabilidade. O vetor de probabilidade pode ser visto como um vetor de protótipos que

gera soluções bem avaliadas de acordo com o conhecimento disponível sobre o espaço de busca (ANDRADE, 2013b).

Os parâmetros utilizados no PBIL estão expostos na **Tabela 5**:

Taxa de aprendizado	0.1
Taxa de mutação	0.3
Coeficiente de mutação	0.01

Tabela 5 – Parâmetros PBIL

1.6.4.4 Otimização por acasalamento de abelhas (MBO)

A Otimização por acasalamento de abelhas (MBO) (ABBASS, 2001) é inspirado no comportamento de sociabilidade em Hymenoptera, como abelhas, formigas e vespas, focando no processo de acasalamento das abelhas produtoras de mel.

As abelhas apresentam diversas características que justificam seu uso como modelo de comportamento inteligente. Essas características incluem divisão do trabalho, comunicação a nível individual e de grupo. Nesse algoritmo, é definido um modelo unificado para o acasalamento de abelhas no contexto da otimização que simula a evolução das abelhas começando com uma colônia solitária (rainha solteira sem uma família) para o surgimento de uma colônia social (uma ou mais rainhas com uma família).

Os parâmetros utilizados no MBO, **Tabela 6**:

Torneio	2 indivíduos
Quantidade de Rainhas	3
Quantidade de trabalhadores	3
Tamanho Espermateca	21
Redutor de Velocidade	0.9
Redutor de energia	0.5
Coeficiente de mutação	0.01
Número de melhorias por filhote	100

Tabela 6 – Parâmetros MBO

RESULTADOS

1.7 Apresentação dos resultados

Para se obter um comparativo melhor dos resultados, foram utilizadas funções específicas para testes, que normalmente são utilizadas na literatura, como vimos anteriormente. Além disso, por se tratar de um conjunto de metaheurísticas com princípios e estratégias diferentes, foram estabelecidos alguns quesitos de comparação, podendo assim evidenciar as características mais interessantes de cada abordagem. Os quesitos são: O melhor resultado obtido nas execuções, o pior resultado obtido nas execuções, a média de todos os resultados obtidos nas iterações e por fim o tempo médio gasto.

A discussão sobre os resultados obtidos é apresentado a seguir, separadamente função por função

1.7.1 Resultados *OneMax*

Os resultados obtidos a partir dos testes realizados com a função *OneMax*, como pode ser observado na **Tabela 7**, foram satisfatórios. O máximo foi encontrado nas execuções, mesmo não ocorrendo em todas as iterações, como no AG e no PBIL, porém a média obtida foi boa e com o melhor tempo médio de todos os algoritmos analisados, levando pouco mais de 1 segundo na execução, um resultado bastante notável, visto que é menos da metade do segundo melhor, a saber o PBIL.

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	**100**	100	**100**	14.08
PBIL	**100**	100	**100**	5.31
MBO	**100**	98	98.5	44.78
MHV	**100**	97	98.7	**1.64**

Tabela 7 - Comparação de resultados Função OneMax

1.7.2 Resultados *TrapFive*

A função *TrapFive* possui um nível de complexidade muito mais elevado que a função OneMax. Grande parte do problema para uma solução ótima está na incapacidade de identificar blocos que sigam um padrão, uma vez que eles não são formados por elementos sequenciais.

Os resultados estão listados na **Tabela 8**. É possível observar que o **MHV** conseguiu obter o melhor valor de todas as metaheurísticas nessa função (88), visto que nenhum algoritmo foi capaz de encontrar o ótimo global, além disso obteve uma boa média de resultados e o segundo melhor tempo de execução (12.16), perdendo apenas para o PBIL.

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	87	82	**84.2**	13.73
PBIL	87	82	83.9	**5.52**
MBO	84	79	80.6	39.43
MHV	**88**	82	83.8	12.16

Tabela 8 - Comparação de resultados Função *TrapFive*

1.7.3 Resultados *Sphere*

Das funções de minimização, a função *Sphere* é a que apresenta menor grau de dificuldade, especialmente para dimensões menores. Com N = 3, onde N é a dimensão do espaço de busca, os resultados obtidos foi o 3º melhor num comparativo geral, ficando atrás apenas do AG e do PBIL, que alcançaram o ótimo global. O melhor resultado do **MHV** equiparou-se em ordem de grandeza aos das outras metaheurísticas, com valores de 2^{-12}. Os piores resultados e as médias obtidas mostram a curta variação nas 10 execuções, visto que a média dos resultados é muito próxima ao ótimo global. Com relação ao tempo médio de execução o **MHV** supera todos os outros com 3.63, passando novamente o PBIL, conforme pode ser observado na **Tabela 9**.

Com a dimensão N = 5, o desempenho foi muito bom, visto que nenhuma metaheurística alcançou o mínimo global. O melhor valor atingido foi de 5^{-7}, mas os resultados não variaram muito durante as execuções, obtendo também uma boa média em relação aos outros. A média de tempo permaneceu abaixo de todos os outros, com 6.13. Os dados são apresentados na **Tabela 10**.

Em N = 10, novamente o **MHV** obtém o 3º melhor resultado entre as metaheurísticas, a 3ª melhor média e permanece também com o melhor tempo médio, 12.49. Os testes para N=10 encontram-se na **Tabela 11**.

Levando em conta os dados apresentados nas **Tabelas 9**,**10** e **11**, concluímos que o **MHV** aplicado à função *Sphere*, obteve uma ótima média de resultados, situando-se normalmente em 3º em questão de resultados, porém retoma a frente no quesito tempo de execução de forma absoluta em todas as dimensões (N = 3, N=5 e N = 10).

Seguem as tabelas que apresentam os resultados das comparações do **MHV** e os outros algoritmos utilizando a função *Sphere* para N = 3, N = 5 e N = 10, respectivamente:

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	**0**	2.72848e-8	1.81899e-8	13.07
PBIL	**0**	1.81899e-8	**1.27329e-8**	5.85
MBO	9.09495e-9	2.86491e-6	7.93990e-7	16.34
PSO	6.8212e-9	-	2.5739e-8	-
MHV	2.21962e-12	7.27617e-8	1.4714e-8	**3.63**

Tabela 9 - Comparação de resultados Função *Sphere(De Jong)*, N=3

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	**9.09495e-9**	4.54747e-8	**2.091835e-8**	17.31
PBIL	**9.09495e-9**	2.3465e-6	2.537494e-7	8.24
MBO	0.00124297	0.07692590	0.03070175	25.16
PSO	1.9213e-6	-	5.2909e-4	-
MHV	5,99285e-7	6,54929e-4	1,09259e-4	**6.13**

Tabela 10 - Comparação de resultados Função *Sphere(De Jong)*, N=5

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	4.54747e-8	1.24601e-6	5.28416e-7	32.95
PBIL	**3.63798e-8**	2.38288e-6	**3.49246e-7**	14.76
MBO	15.7480	64.6136	52.3674	161.51
PSO	0.1121	-	1.9819	-
MHV	0,05926	3,32186	5,2937e-1	**12.49**

Tabela 11 - Comparação de resultados Função *Sphere(De Jong)*, N=10

1.7.4 Resultados *Rosenbrock*

Na função *Rosenbrock* para N=3 a primeira vista é onde o algoritmo teve o pior desempenho , visto que na comparação do melhor resultado o **MHV** conseguiu apenas um 0.52, sendo o pior na comparação, porém isso não se confirma ao analisar-se que ele alcançou a segunda melhor média de resultados e isso também é evidenciado nos piores resultados, visto que a variação entre o melhor e o pior resultado obtido pelo **MHV** tem uma variação menor do que a da maioria dos outros algoritmos apresentados, além disso em termos de resultados o tempo de execução foi menor do que qualquer dos outros, com 3.92.

Para N = 5, é perceptível, que a metaheurística proposta continua a obter bons resultados, encontrou maiores dificuldades para uma dimensão maior, como era de se esperar, mas o pior resultado encontrado não se distanciou muito do melhor, mostrando uma coerência e apesar de o melhor resultado só ter sido melhor do que o obtido pelo MBO, a média foi muito próxima aos resultados obtidos pelos demais algoritmos e como antes, o tempo de execução permanece inferior a todos os demais com 6.59.

Na dimensão N = 10, o melhor resultado encontrado volta a estar em 3º, sendo ultrapassado apenas pelo AG e pelo PBIL e o tempo de execução cravando os 13 segundos, mostrando novamente que mesmo para dimensões grandes, o tempo gasto para encontrar uma boa solução é muito curto.

De forma geral na função *Rosenbrock* o **MHV** mostrou uma boa média de resultados e é uma ótima escolha principalmente levando em consideração o tempo de execução, as **Tabelas 12**, **13** e **14** mostram, respectivamente, os resultados obtidos para N= 3, 5 e 10.

	Melhor Resultado	**Pior Resultado**	**Média dos resultados**	**Tempo médio**
AG	0.46838	6.63989	2.5381714	15.44
PBIL	0.107154	9.89095	2.347752	5.67
MBO	**0.0491312**	0.6482640	**0.35350382**	106.27
PSO	0.0934	-	0.5164	-
MHV	0,52465	3,70672	1,5012	**3.92**

Tabela 12 - Comparação de resultados Função *Rosenbrock*, N=3

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	**1.4137**	4.77349	3.599487	21.76
PBIL	3.14446	4.68945	3.915263	8.02
MBO	3.79254	4.98461	4.231863	117.64
PSO	2.2470	-	**2.5162**	-
MHV	3,49360	4,51205	4,0419	**6.59**

Tabela 13 - Comparação de resultados Função *Rosenbrock*, N=5

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	**6.63907**	16.0315	**10.10238**	34.77
PBIL	8.08279	15.9249	13.0808	13.86
MBO	1887.70	36671.5	21772.249	199.69
PSO	32.8310	-	367.8391	-
MHV	11,6719	1746,821	495.62	**13.00**

Tabela 14 - Comparação de resultados Função *Rosenbrock*, N=10.

1.7.5 Resultados *Rastrigin*

Na função *Rastrigin* o **MHV** apresentou resultados excelentes, visto que a função tem um nível alto de complexidade, os resultados na grandeza de 10^{-6} se aproximaram muito do ótimo global, igualando ao AG e ao MBO. O tempo médio de execução foram surpreendentes 3.88, vide **Tabela 15**.

Na **Tabela 16** pode ser visto os resultados em N = 5, foram bons resultados, novamente permanecendo em 3º, superando PSO e MBO, manteve uma média razoável e mesmo com a complexidade maior o tempo médio de execução não subiu muito e continuou o melhor no quesito, com 6.26.

Na dimensão de N = 10, apresentado na **Tabela 17**, os resultados foram ainda melhores, mostrando uma grande efetividade para problemas de grandes dimensões e alta

complexidade, pois foi alcançada a segunda posição em relação ao melhor resultado, perdendo apenas para o AG, a média dos resultados foi bastante satisfatória e é ainda mais admirável visto que os resultados foram bons, sem perder qualidade no tempo médio de execução, sendo ainda o mais rápido com 12.71.

Analisando todas as dimensões fica evidente os bons resultados encontrados pelo **MHV** principalmente levando em conta que em todas as dimensões ele obteve sempre o melhor tempo médio de execução, contudo não perdeu a qualidade das soluções encontradas.

	Melhor Resultado	**Pior Resultado**	**Média dos resultados**	**Tempo médio**
AG	1.80437e-6	3.50046e-4	4.83569e-5	16.7
PBIL	**0**	1.06261	0.3100641	5.4
MBO	1.80437e-6	7.62429e-3	8.7682499e-4	6.32
PSO	1.3533e-6	-	**6.5138e-6**	-
MHV	1,80437e-6	1,07022	2,3512e-2	**3.88**

Tabela 15 - Comparação de resultados Função *Rastrigin,* N = 3.

	Melhor Resultado	**Pior Resultado**	**Média dos resultados**	**Tempo médio**
AG	**3.60873e-6**	1.16602	0.5387128	23.55
PBIL	**3.60873e-6**	8.2265	2.982462	7.75
MBO	0.549241	1.82835	1.483502	99.63
PSO	0.0034	-	**0.3799**	-
MHV	0,00317	7,05978	2,9755	**6.26**

Tabela 16 - Comparação de resultados Função *Rastrigin,* N = 5.

	Melhor Resultado	Pior Resultado	Média dos resultados	Tempo médio
AG	**0.00118907**	6.37324	**2.97374418**	33.43
PBIL	4.07353	18.6522	11.7417	14.85
MBO	95.2457	130.769	109.8309	166.34
PSO	10.3925	-	39.1396	-
MHV	3.1870	40.4714	19.51	**12.71**

Tabela 17 - Comparação de resultados Função *Rastrigin,* N = 10.

CONCLUSÃO E PERSPECTIVAS

Os resultados obtidos evidenciam que a metaheurística foi bem sucedida e apontam o seu grande potencial para resolver problemas de otimização, visto que teve um ótimo desempenho em todas as funções utilizadas para teste.

Para problemas menos complexos, como as menores dimensões (N= 3) dos problemas de minimização e a função *OneMax*, além de alcançar excelentes resultados, incluindo o ótimo como visto na **Tabela 7**, mostrou-se imbatível no tempo de execução convergindo de forma muito rápida.

Em dimensões intermediárias, nas funções de minimização para N=5, o **MHV** manteve uma boa média no desempenho, mostrando-se preciso e tendo um desempenho geral normalmente em terceiro melhor, porém ainda superior quando a questão é o tempo de execução.

Para problemas mais complexos o **MHV** surpreendeu ainda mais, mantendo bons resultados e um tempo de execução surpreendente, superando todos os outros algoritmos e perdendo apenas na **Tabela 8**, na função *TrapFive* porém é exatamente nela onde o desempenho é superior aos demais, comprovando uma robustez impressionante, e apesar de não ter sido o melhor no quesito tempo, ficou com o segundo lugar, mostrando que permanece com uma média de tempo acima da maioria.

Desde a formulação, implementação e comparação é possível observar que trata-se de um algoritmo promissor e potencialmente poderoso, o refinamento dos parâmetros ainda mais detalhadamente, bem como algumas melhorias na estrutura poderão trazer ganhos significativos para os resultados da metaheurística.

Numa análise geral, a **MHV** se mostrou promissora em todas as funções, situando-se, em média, em terceiro lugar dos melhores resultados encontrados, porém, sendo quase imbatível no quesito tempo de execução, fator impressionante e que mostra o quanto a técnica é promissora.

Em trabalhos futuros pretende-se realizar uma série de testes com grande parte das funções existentes na literatura, para que se tenha uma visão geral do funcionamento do algoritmo com outras classes de problemas.

Há ainda a intenção de um comparativo com um grande grupo de metaheurísticas tradicionais na literatura, com variações nos parâmetros e maior diversidade de problemas, para que se obtenha uma visão geral dos melhores campos de atuação do **MHV**.

REFERÊNCIAS

ABBASS, H. A. **MBO: marriage in honey bees optimization-a Haplometrosis polygynous swarming approach**. Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546). **Anais**...IEEE, 2001Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=934391>

ALGORITHMS, G. Genetic Algorithms - John H . Holland understand Genetic Algorithms - John H . Holland. p. 12–15, 2005.

ANDRADE, J. V. DE. ANÁLISE COMPARATIVA DE META-HEURÍSTICAS BIOINSPIRADAS QUANDO APLICADAS A PROBLEMAS DE OTIMIZAÇÃO. In: INTERGOVERNMENTAL PANEL ON CLIMATE CHANGE (Ed.). . **Climate Change 2013 - The Physical Science Basis**. Cambridge: Cambridge University Press, 2013a. v. 53p. 1–30.

ANDRADE, J. V. DE. ANÁLISE COMPARATIVA DE META-HEURÍSTICAS BIOINSPIRADAS QUANDO APLICADAS A PROBLEMAS DE OTIMIZAÇÃO. In: INTERGOVERNMENTAL PANEL ON CLIMATE CHANGE (Ed.). . **UFVJM**. Cambridge: Cambridge University Press, 2013b. v. 53p. 1–30.

ANMER DASKINA, S. K. Group Leaders Optimization Algorithm. **IEEE**, v. 1, n. 00, p. 1–18, 2011.

AUSTIN, M. Illuminated gloves—a cycling safety aid. **Electronic Systems News**, v. 1987, n. 1, p. 25, 1987.

AVANTHAY, C.; HERTZ, A.; ZUFFEREY, N. A variable neighborhood search for graph coloring. **European Journal of Operational Research**, v. 151, n. 2, p. 379–388, 2003.

AVIGAD, J.; DONNELLY, K. Formalizing O notation in Isabelle/HOL. **Automated Reasoning**, 2004.

AZIMI, J.; JALALI, A.; FERN, X. Dynamic Batch Bayesian Optimization. **Genetic and Evolutionary Computation**, v. 1, p. 525–532, 14 out. 2011.

BALAKRISHNAN, V.; BOYD, S.; BALEMI, S. Branch and bound algorithm for computing the minimium stability degree of parameter-dependent linear systems. **Int. J. of Robust and Nonlinear Control**, v. 1, n. 4, p. 295–317, 1991.

BALUJA, S.; CARUANA, R. Removing the genetics from the standard genetic algorithm. **Icml**, p. 1–11, 1995.

BERTSIMAS, D.; TSITSIKLIS, J. Simulated Annealing. **Statistical Science**, v. 8, n. 1, p. 10–15, fev. 1993.

BOYD, S.; MATTINGLEY, J. Branch and bound methods. **Notes, Stanford University, Stanford, CA**, p. 1–18, 2003.

CARNEIRO, T. et al. Um Levantamento Na Literatura Sobre a Resolução De Problemas De Otimização Combinatória Através Do. n. February 2016, 2014.

CARRANO, F. M. AND J. J. P. **Data Abstraction & Problem Solving with C++**. Cambridge: Pearson; 6 edition, 1995. v. 1

CASTRO, L. N. DE; ZUBEN, F. J. V. **Recent Developments in Biologically Inspired Computing**. [s.l.] Idea Group Publishing, 2005.

CHRISTIANE, A.; ZAPELINI, Z. Universidade de São Paulo Instituto de Matemática e Estatística Departamento de Ciências da Computação Um Estudo Abrangente sobre

Metaheurística , incluindo um Histórico. 2009.
COELHO, F.; NETO, J. P. Teoria da Computação, Computabilidade e Complexidade. 2010.
CORDENONSI, A. Z.; MÜLLER, F. M.; DE, F. P. O LOBO e o Caixeiro Viajante. **SBIE - Simpósio Brasileiro de Informática na Educação**, p. 71–81, 2007.
DA SILVA, M. J. R. et al. Composição físico-química do mosto e do vinho branco de cultivares de videiras em resposta a porta-enxertos. **Pesquisa Agropecuaria Brasileira**, v. 50, n. 11, p. 1105–1113, 2015.
DULCE, A.; SUCENA, M. Desenvolvimento e Crescimento. n. Módulo 3, p. 1–15, [s.d.].
EBERHART, R.; KENNEDY, J. A new optimizer using particle swarm theory. **MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science**, p. 39–43, 1995.
ELISA TULER; NIVIO ZIVIANI; CHARLES ORNELAS; LEONARDO ROCHA; LEONARDO MATA. Problemas NP-Completo e Algoritmos. 2006.
GENDREAU, M.; POTVIN, J.-Y. Metaheuristics in Combinatorial Optimization. **Annals of Operations Research**, v. 140, n. 1, p. 189–213, nov. 2005.
GLOVER, F. Tabu Search—Part I. **ORSA Journal on Computing**, v. 1, n. 3, p. 190–206, 1989.
GLOVER, F. Scatter Search. p. 519–537, 2003.
GLOVER, F.; LAGUNA, M.; MARTI, R. Principles of tabu search. **Approximation Algorithms and Metaheuristics**, v. 23, p. 1–12, 2007.
GOLDBERG, D.; DEB, K. Massive multimodality, deception, and genetic algorithms. **Parallel Problem Solving from Nature - PPSN**, p. 37–48, 1992.
HERTZ, A.; WIDMER, M. Guidelines for the use of meta-heuristics in combinatorial optimization. **European Journal of Operational Research**, v. 151, n. 2, p. 247–252, 2003.
JAMIL, M.; YANG, X.-S. A Literature Survey of Benchmark Functions For Global Optimization Problems Citation details: Momin Jamil and Xin-She Yang, A literature survey of benchmark functions for global optimization problems. **Int. Journal of Mathematical Modelling and Numerical Optimisation**, v. 4, n. 2, p. 150–194, 2013.
JAMILSON, M. et al. Inteligência Computacional para Otimização Sumário. 2011.
KENNEDY, J.; EBERHART, R. C. **A discrete binary version of the particle swarm algorithm**. 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation. **Anais**...IEEE, 1997Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=637339>
LEONARDO RAMOS EMMENDORFER. ˜ Entre Genes Em Aprendizado Da Ligac ¸ Ao ˜ Evolutiva : Uma Nova Abordagem Computac ¸ Ao. 2007.
LOURENÇO, H.; MARTIN, O.; STUTZLE, T. **Iterated local search**. [s.l: s.n.].
MANDELLI, F. **Comportamento Meteorológico e sua Influência na Vindima de 2009 na Serra Gaúcha**. Disponível em: <http://revistagloborural.globo.com/GloboRural/0,6993,EEC1664546-4529,00.html>. Acesso em: 21 mar. 2016.
MARCO DORIGO, MAURO BIRATTARI, AND T. S. **Ant Colony Optimization**. [s.l: s.n.].
MARTÍ, R.; LAGUNA, M.; GLOVER, F. Principles of scatter search. **European Journal of Operational Research**, v. 169, n. 2, p. 359–372, 2006.

MCGOVERN, P. E. Stone Age Wine. **Ancient Wine**, p. 1–15, 2003.
MITCHELL, M. Genetic algorithms: An overview. **Complexity**, v. 1, n. 1, p. 31–39, 1995.
MLADENOVIĆ, N.; HANSEN, P. Variable neighborhood search. **Computers & Operations Research**, v. 24, n. 11, p. 1097–1100, nov. 1997.
MOJTABA AHMADIEH KHANESAR, MOHAMMAD TESHNEHLAB, M. A. S. A novel binary particle swarm optimization. **2007 Mediterranean Conference on Control & Automation**, v. 1, n. 1, p. 1–6, 2007.
MOLGA, M.; SMUTNICKI, C. Test functions for optimization needs. **Test functions for optimization needs**, n. c, p. 1–43, 2005.
MOREIRA, N. Computabilidade - Uma introdução. 2003.
MYLES, S. et al. Genetic structure and domestication history of the grape. **Proceedings of the National Academy of Sciences of the United States of America**, v. 108, n. 9, p. 3530–3535, 2011.
NILSON, T. S. Influência do clima sobre os estádios fenológicos da videira e sobre a quantidade da produção. **Journal of Chemical Information and Modeling**, v. 53, n. 9, p. 1689–1699, 2010.
PALLAS, B. et al. Influence of intra-shoot trophic competition on shoot development in two grapevine cultivars (Vitis vinifera). **Physiologia Plantarum**, v. 134, n. 1, p. 49–63, 2008.
PALLAS, B. et al. A stochastic growth model of grapevine with full interaction between environment, trophic competition and plant development. **Plant Growth Modeling, Simulation, Visualization and Applications, Proceedings - PMA09**, p. 95–102, 2010.
PENNACHIN, C. Algoritmos de Estimativa de Distribuição. 2010.
POHLHEIM, H. Examples of objective functions. **GEATbx version**, v. 8, n. December, 2005.
RICH, E.; KNIGHT, K. **Artificial Intelligence**, 1990.
RICHTER, J. N.; PAXTON, J. Adaptive Evolutionary Algorithms on Unitation, Royal Road and Longpath Functions. **Computational Intelligence**, p. 381–386, 2005.
SILVÉRIO, H. L.; RODRIGUES, L. C. DE A.; STEINER, M. T. A. **Meta-Heurísticas em Pesquisa Operacional**. [s.l: s.n.].
TAKAHASHI, A. G.-C.; C. H. A.; R. **Manual de computação evolutiva e meta-heurística**. [s.l: s.n.].
TALBI, E.; POPULACIONAIS, M. Metaheurísticas Populacionais Baseado no livro METAHEURISTICS - From Design to Implementation Conteúdo. 2014.
TEIXEIRA; ANTONIO HERIBERTO DE C. **Cultivo da videira – clima**. Disponível em: <https://sistemasdeproducao.cnptia.embrapa.br/FontesHTML/Uva/CultivodaVideira/clima.htm>. Acesso em: 2 fev. 2016.
WESTERKAMP, C. Glossário Plantas-com-Flores: Forma e função. 2012.
WRIGHT, A. H. Genetic algorithms for real parameter optimization. **Foundations of Genetic Agorithms**, p. 205–220, 1990.
XANTRE, W. **Apostila_Informática Aplicada 2015.2**. Disponível em: <https://pt.scribd.com/doc/275385492/Apostila-Informatica-Aplicada-2015-2>. Acesso em: 19 fev. 2016.
YAGIURA, M.; IBARAKI, T.; GLOVER, F. A path relinking approach with ejection chains for the generalized assignment problem. **European Journal of Operational**

Research, v. 169, n. 2, p. 548–569, 2006.
YANG, X.-S. **Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010a. v. 284
YANG, X.-S. Test Problems in Optimization. **Engineering Optimization: An Introduction with Metaheuristic Applications**, n. 2010, 2010b.
ZUBEN, F. J. VON. Introdução à Inteligência Artificial. n. 09, p. 1–48, 2013.

AUTORIZAÇÃO

Autorizo a reprodução e/ou divulgação total ou parcial do presente trabalho, por qualquer meio convencional ou eletrônico, desde que citada a fonte.

(Fonte Times New Roman, tamanho 12, letras minúsculas, justificado, espacejamento 1,5cm entrelinhas)

_____

Nome do autor
e-mail
Nome da Instituição
Endereço institucional

(Fonte Times New Roman, tamanho 12, letras minúsculas, centralizado, espacejamento 1,5cm entrelinhas)