LVI SBPO SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
DE 04 ATÉ 07 NOVEMBRO 2024
FORTALEZA | CEARÁ

NOVAS VARIAÇÕES DO ALGORITMO GOLDEN BALL COM CHAVES ALEATÓRIAS TENDENCIOSAS: UMA INVESTIGAÇÃO COMPARATIVA APLICADA AO PROBLEMA DO CAIXEIRO VIAJANTE

Bryan Keven Rocha Brauna
Universidade Federal do Ceará
R. Felipe Santiago, 411 - Cidade Universitária, Russas - CE, 62900-000
contato.bryankeven@gmail.com

Bonfim Amaro Junior
Universidade Estadual do Ceará
Av. Dr. Silas Munguba, 1700 - Campus do Itaperi Fortaleza - CE, 60714-903
bonfim.amaro@uece.br

Guilherme Nepomuceno de Carvalho
Universidade Estadual do Ceará
Av. Dr. Silas Munguba, 1700 - Campus do Itaperi Fortaleza - CE, 60714-903
guilhermenepomuceno46@gmail.com

RESUMO

Este artigo apresenta uma investigação sobre meta-heurísticas inspiradas na natureza e baseadas em população. O principal foco está na introdução de três novas variantes do algoritmo Golden Ball (GB). As variantes incluem a integração da técnica Biased Random Key (BRK) ao GB, a criação de instâncias paralelas com o BRKGB-I e a simulação de torneios de futebol no BRKGB-WC e BRKGB-BFC. Os resultados experimentais, utilizando instâncias do Travelling Salesman Problem (TSP), mostram que as novas variantes oferecem uma boa relação entre qualidade da solução e tempo de execução, com destaque para o BRKGB-BFC, que obteve as melhores soluções em termos de qualidade.

PALAVRAS CHAVE. Meta-heurísticas, Algoritmo Golden Ball, Biased Random Key, Otimização Combinatória, Travelling Salesman Problem.

ABSTRACT

This paper presents an investigation into nature-inspired and population-based metaheuristics, focusing on the introduction of three new variants of the Golden Ball (GB) algorithm. The variants include integrating the Biased Random Key (BRK) technique into GB, creating parallel instances with BRKGB-I, and simulating soccer tournaments in BRKGB-WC and BRKGB-BFC. Experimental results using Travelling Salesman Problem (TSP) instances demonstrate that the new variants offer a good balance between solution quality and execution time, with BRKGB-BFC standing out for providing the best solutions in terms of quality.

KEYWORDS. Metaheuristics, Golden Ball Algorithm, Biased Random Key, Combinatorial Optimization, Travelling Salesman Problem.

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

1. Introdução

O uso da inteligência artificial (IA) aumentou significativamente devido à sua importância em diversos setores da sociedade, economia e tecnologia. Na pesquisa científica, anualmente, inúmeros pesquisadores se dedicam ao desenvolvimento de técnicas e métodos para resolver problemas conhecidos no campo da Pesquisa Operacional, como o *Travelling Salesman Problem (TSP)* e o *Capacitated Vehicle Routing Problem (CVRP)*.

Esses problemas são classificados como NP-Hard [Garey and Johnson, 1979], o que implica que soluções triviais de tempo polinomial não são eficientes para encontrar bons resultados. No entanto, existem diversas técnicas focadas em resolver problemas dessa natureza, destacando-se as heurísticas e meta-heurísticas.

As heurísticas são técnicas utilizadas para decidir entre várias alternativas de ações que prometem alcançar os objetivos com maior eficiência [Pearl, 1984]. Entre as heurísticas mais conhecidas estão o Tabu Search Algorithm (TSA) [Gallego et al., 2000] e o Simulated Annealing Algorithm (SAA) [Rutenbar, 1989]. Por outro lado, as meta-heurísticas podem ser compreendidas como processos mestres que guiam operações de heurísticas subordinadas com o objetivo de obter soluções de alta qualidade. Essas heurísticas subordinadas podem variar de simples operações de busca local a algoritmos complexos. Exemplos populares de meta-heurísticas incluem o Artificial Bee Colony (ABC) [Zhang et al., 2010] e o Imperialist Competitive Algorithm (ICA) [Atashpaz-Gargari and Lucas, 2007].

Dada a vasta popularidade dessas técnicas, a comunidade científica desenvolveu e aprimorou inúmeros métodos com diferentes objetivos, características e aplicabilidades em diversos setores da tecnologia, como engenharia [Santana et al., 2021] e saúde [Latifoğlu, 2013], entre outros.

Este artigo foca nas meta-heurísticas classificadas como *Nature-Inspired* e *Population-Based*, que simulam comportamentos da natureza para realizar estratégias de busca e aprimoramento utilizando múltiplos grupos populacionais. O objetivo é apresentar novas variações fundamentadas no algoritmo *Golden Ball*. Esses novos métodos utilizam a concepção do paralelismo para executar processos de forma concorrente, aumentando a exploração do espaço de busca com o objetivo de encontrar soluções destacadas. Além disso, incorporam uma técnica que torna os algoritmos genéricos, fornecendo uma interface para adaptação a diferentes problemas.

Ademais, este artigo está organizado da seguinte forma: nas seções 2 e 3, são apresentados, respectivamente, os trabalhos relacionados e as variantes propostas para o *Golden Ball*. A seção 4 explana os resultados dos experimentos computacionais realizados. Por fim, a seção 5 retrata as considerações finais e possibilidades de trabalhos futuros para o contexto abordado.

2. Abordagens meta-heurísticas fundamentadas em população de soluções

Existem inúmeras meta-heurísticas, cada uma com suas próprias características e objetivos específicos. Apesar das semelhanças entre muitas delas, é possível classificá-las com base em suas conformidades. Uma das principais propriedades a ser considerada é a quantidade de soluções candidatas que as meta-heurísticas utilizam simultaneamente. Essas estratégias podem ser divididas entre aquelas que utilizam uma única solução (*single solution*) e aquelas que se baseiam em um conjunto de soluções (*population-based*).

A abordagem *single solution* mantém e aprimora uma única solução candidata ao longo do processo, empregando métodos específicos para resolver o problema de maneira satisfatória [Blum and Roli, 2003]. Em contraste, os métodos *population-based* gerenciam múltiplas soluções simultaneamente, otimizando-as para identificar a melhor [Beheshti and Shamsuddin, 2013]. Este documento foca em estratégias baseadas em população, destacando sua capacidade de explorar espaços de busca mais amplos e avaliando o equilíbrio entre o tempo de execução e a eficácia das soluções.

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

Nesse contexto, os métodos fundamentados em população mantêm um conjunto de, no mínimo, duas soluções $k \geq 2$ que são usualmente geradas de forma aleatória, condicionado por algum conhecimento sobre o problema ou, ainda, pré-executadas. Assim, ao longo da execução do algoritmo, essas soluções são aprimoradas, frequentemente utilizando características populacionais para guiar a busca e otimizá-las [Moscato and Cotta, 2003].

Esses algoritmos podem explorar k espaços de busca distintos, o que pode elevar a probabilidade de encontrar o máximo global, se conduzido de forma eficiente. No entanto, destaca-se também que esta classe de algoritmos, ao armazenar múltiplas soluções candidatas, exige uma quantidade maior de recursos para o armazenamento de informações quando comparados com métodos de solução única. Numerosas meta-heurísticas e projetos científicos em aplicações práticas empregam algoritmos baseados em população. Entre eles, destacam-se o *Biased Random Key Genetic Algorithm-BRKGA* [Gonçalves and Resende, 2011] e *Golden Ball* [Osaba et al., 2013].

O Biased Random Key Genetic Algorithm (BRKGA) é uma meta-heurística baseada em uma população de soluções, que emprega conceitos fundamentais do renomado método Genetic Algorithm (GA). O GA é uma técnica inspirada pelos princípios da seleção natural de Darwin e pertence à ampla classe dos algoritmos evolucionários [David et al., 1989]. É frequentemente utilizado para gerar resultados eficazes e de alta qualidade.

O GA começa com uma solução candidata gerada aleatoriamente e realiza operações baseadas em processos biológicos, visando desenvolver soluções melhores. As operações principais incluem mutação, crossover e seleção [Holland, 1984]. Além do GA, o BRKGA incorpora uma estratégia conhecida como Biased Random Key (BRK), que consiste em codificar uma solução em um vetor de chaves, mapeando todos os valores da solução para números reais no intervalo de (0, 1], de forma aleatória. Posteriormente, esse vetor pode ser ordenado e decodificado para obter a avaliação de fitness da solução [Bean, 1994].

Inicialmente, o BRKGA cria um conjunto de N soluções, cada uma representando um vetor de chaves aleatórias. O tamanho desses vetores geralmente depende da instância do problema. Estes vetores são decodificados em soluções práticas e são utilizados nas operações conhecidas como troca de informações genéticas para aprimorar e gerar soluções melhores. As manipulações das soluções candidatas, em todas as etapas do algoritmo, são realizadas em função dos vetores (indivíduos). A decodificação para obter o valor de fitness ocorre apenas quando necessário. Essa característica confere ao algoritmo uma natureza genérica e independente do problema específico, permitindo sua adaptação a qualquer problema, desde que se ofereça um método para codificar e decodificar as soluções—aspecto que constitui a parte dependente do problema.

Este algoritmo também ordena todo seu conjunto P de soluções com base na qualidade determinada pelo decodificador e segmenta a população em duas categorias: a elite (P_e), que é a subpopulação copiada diretamente para a próxima geração, e a não elite ($P - P_e$), utilizada para gerar novas soluções por meio do crossover em conjunto com as elites. É importante ressaltar que uma característica fundamental do BRKGA é a capacidade de enviesar as soluções durante o processo de crossover. Esse enviesamento é implementado permitindo que os alelos das novas soluções sejam influenciados pelas características das soluções elite, com uma probabilidade definida pelo parâmetro ProbE (probabilidade de ser elite), um valor real entre 0 e 1. Durante o crossover, um valor aleatório, também no intervalo de 0 a 1, é gerado para cada alelo. Se esse valor for menor ou igual ao ProbE, o alelo da nova solução será herdado da elite; caso contrário, será herdado da solução não elite. Comumente, valores de ProbE entre 0,75 e 0,90 são utilizados para criar novas soluções enviesadas. O crossover utilizado para realizar o processo supracitado é *Parameterised Uniform Crossover* [Spears et al., 1991].

O BRKGA é amplamente utilizado para abordar uma variedade de desafios computacio-

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

nais, especialmente aqueles relacionados a problemas de roteamento. Nesse contexto, é relevante mencionar o artigo [Santos, 2022], que destaca a aplicação eficaz dessa técnica em contextos práticos. Este trabalho consiste em utilizar o BRKGA em conjunto com o *Q-Learning*, um algoritmo de aprendizagem por reforço [Watkins and Dayan, 1992], denominando a meta-heurística híbrida como BRKGA-QL. A técnica é aplicada para resolver o problema *NP-Hard* conhecido como *Capacitated Vehicle Routing Problem* (CVRP). O BRKGA-QL utiliza uma tabela Q para registrar e atualizar todas as ações tomadas a cada configuração de parâmetros da meta-heurística, propondo a seleção do melhor conjunto de parâmetros para executar as ações mais eficientes e alcançar os objetivos.

A abordagem supracitada foi comparada com o método *Branch & Cut* (B & C) [Naddef and Rinaldi, 2002], utilizando 11 instâncias de 30, 60, 90 e 120 vértices, totalizando 44 testes para as duas abordagens. Os resultados mostram que, em amostras de 30 e algumas de 60 vértices, ambos os métodos obtêm métricas semelhantes. No entanto, à medida que o número de vértices aumenta, o tempo de execução do B&C cresce exponencialmente, com alguns testes durando mais de uma hora e, em alguns casos, excedendo o limite de tempo (120 minutos) sem encontrar uma solução. Por outro lado, o BRKGA-QL, na maioria das execuções, não excedeu 40 segundos, demonstrando sua capacidade de encontrar soluções de qualidade tanto para amostras pequenas quanto grandes.

Além do BRKGA, outro algoritmo baseado em população amplamente conhecido na literatura é o *Golden Ball* (GB) [Osaba et al., 2013], uma meta-heurística que simula o comportamento de um torneio de futebol para a seleção de um melhor jogador(solução). O GB fragmenta a população em Nt subpartições denominadas times, cada uma composta por N soluções referidas como jogadores. Além disso, todos os times possuem um *coach* que incorpora, um ou mais, técnicas de treinamento com o propósito de aprimorar a qualidade dos jogadores, tais métodos são: treinamento comum e customizado. O algoritmo em questão contém uma série de etapas cruciais para a execução adequada do método. A figura 1 ilustra o fluxo do GB.

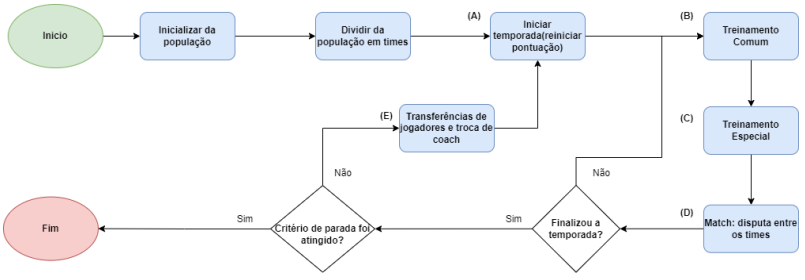


Figura 1: Fluxograma do algoritmo Golden Ball

A fase de inicialização é marcada pela criação e divisão da população em times, onde os jogadores de cada equipe são inicializados com valores aleatórios que representam suas qualidades. Além das propriedades dos jogadores, os *coaches* são definidos de forma esporádica para os diferentes times. É importante destacar que o *coach* representa um conjunto de treinamento comum e customizado.

Após o estágio de iniciação, a tabela de pontuação das equipes é criada na Etapa (A) da figura 1, atribuindo pontuações iniciais de zero a todos os participantes, marcando o início do torneio. A tabela é reiniciada a cada nova temporada.

A etapa (B) da figura 1 refere-se ao treinamento comum, onde todos os jogadores de todas as equipes são submetidos a um algoritmo de melhoria de qualidade (busca local). Nesta etapa, os principais métodos utilizados são 2-OPT e 3-OPT [Lin, 1965]. Os métodos mencionados geram uma nova solução candidata a partir da solução atual, que a substituirá somente se a qualidade

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

da alternativa for superior; caso contrário, a solução alternativa é simplesmente ignorada. Esse procedimento se repete um número previamente definido de vezes.

Na etapa C (figura 1), o treinamento customizado é destinado apenas a jogadores estagnados que não obtêm mais melhorias de qualidade por meio do treinamento comum por uma quantidade x de vezes. Este treinamento visa evitar que os jogadores fiquem limitados a máximos ou mínimos locais, realizando procedimentos especiais de treinamento em dupla com o melhor jogador da equipe, o capitão. Após o processo, os jogadores podem adquirir características do capitão, resultando em melhorias na sua qualidade ou desbloqueando a capacidade de aprimoramento em treinamentos futuros. Os principais métodos utilizados para o treinamento customizado são: *Order Crossover* [Davis et al., 1985] e *Very Greedy Crossover* [Ahmed, 2010].

Após finalizar os estágios de treinamento, as partidas (etapa D) entre todas as equipes ocorrem. A competição envolve confrontos entre todos os times participantes, decidindo as partidas conforme a qualidade dos jogadores. A equipe com a maior quantidade de jogadores de melhor qualidade na disputa obtém 3 pontos na tabela de classificação, enquanto os perdedores ganham 0 pontos. Em caso de empate, 1 ponto é atribuído a cada equipe.

O torneio consiste em duas iterações, nas quais as etapas A, B, C e D ocorrem duas vezes consecutivas, formando uma temporada completa. Após esse processo, inicia-se a fase de transferência de jogadores entre as equipes, etapa E. Nessa fase, os jogadores mais destacados dos times da metade inferior da tabela são transferidos para as equipes da metade superior, enquanto os jogadores de desempenho inferior dessas últimas são movidos para as equipes da metade inferior. Além disso, as equipes da parte inferior selecionam novos coaches aleatoriamente, semelhante ao que ocorre em equipes de futebol.

O torneio ocorre repetidamente até que o critério de aceitação seja alcançado. Este critério consiste em três condições que precisam ser satisfeitas para concluir o algoritmo: ausência de melhorias na aptidão do melhor indivíduo, falta de aprimoramento no somatório das qualidades dos capitães de cada time e estagnação do agregado de qualidade de todas as equipes participantes.

Ao finalizar o algoritmo, a melhor solução para o problema é determinada pelo capitão da equipe que ocupa o topo da tabela de classificação.

O Golden Ball (GB) é aplicado em diversos campos do conhecimento, resolvendo uma ampla variedade de problemas cotidianos. O artigo de Osaba et al. [Osaba et al., 2013] destaca a eficácia do GB e o comparara com os métodos Genetic Algorithm (GA) e Distributed Genetic Algorithm (DGA). Além disso, o GB foi testado em quatro versões diferentes, variando os parâmetros da quantidade de jogadores por time e o número de equipes. Da mesma forma, os algoritmos GA e DGA foram configurados em duas versões cada, GA1, GA2, DGA1 e DGA2, alterando os valores dos parâmetros de probabilidade de cruzamento (Pc) e probabilidade de mutação(Pm).

Os resultados obtidos indicam que, na maioria das instâncias, o algoritmo GB se destaca em relação aos demais algoritmos. Além disso, ao comparar o tempo de execução, as versões GA1 e DGA1 demandaram mais tempo em todas as instâncias, enquanto as versões GA2 e DGA2 obtiveram tempos semelhantes ao GB.

3. Novas variações na aplicação da meta-heurística *Golden Ball*

Nesta seção, apresentamos três novas variantes baseadas no Golden Ball Algorithm. A subseção 3.1 apresenta a integração do *Biased Random Key* com o algoritmo GB. As subseções subsequentes (3.2, 3.3, 3.4) descrevem as novas estratégias utilizando o BRKGB.

3.1. Biased Random Key Golden Ball (BRKGB)

A adaptação da técnica do *Biased Random Key* (BRK) ao Golden Ball (GB) modifica apenas as etapas de inicialização das soluções candidatas e do treinamento comum das equipes. No processo de inicialização, são geradas Nt equipes, cada uma contendo P jogadores. O jogador

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

é representado por um vetor de chaves aleatórias de tamanho N no intervalo $[0, 1]$. Em outras palavras, os valores que compõem a solução são números reais, em vez de dados específicos do problema. Dessa forma, os vetores podem ser ordenados e mapeados aos valores das instâncias, permitindo a decodificação para obter a qualidade da solução. Os jogadores, ou soluções, são definidos como $P = \{R_1, R_2, R_3, ..., R_n\}$, onde R_i representa um número real entre 0 e 1. O *Algoritmo* 1 descreve o BRKGB.

Algorithm 1 Pseudocódigo do BRKGB

```
Inicialização da população como vetor de chaves aleatórias
Divisão dos jogadores em times
repeat
    Reiniciar a tabela da temporada
    N ← 1
    while N ≠ 2 do
        Treinamento comum das equipes
        Treinamento especial das equipes
        Calcular a qualidade das equipes
        Realizar partidas entre as equipes
        N ← N + 1
    end while
    Etapa de transferência
until critério de parada
```

Algorithm 2 Treinamento Comum

```
for each Ti in Times do
    for each Pi(jogador) in Ti do
        Crie um novo jogador(Pi') como cópia de Pi
        Decodificar o Pi'(vetor de chave) na solução real
        Aplicar Pi' ao 2OPT
        Codificar o Pi' em vetor de chaves
        if Pi.fitness > Pi'.fitness then
            Pi ← Pi'
            contador ← 0
        else
            contador ← contador + 1
        end if
    end for
end for
```

Na fase de treinamento comum, o procedimento de busca local é semelhante ao original; contudo, o método de treino utilizado é o 2-OPT, que necessita da solução real para realizar o processo de melhoria. Portanto, é necessário decodificar cada jogador, ou vetor de chaves aleatórias, em uma solução real antes de submetê-lo ao algoritmo de aprimoramento, conforme demonstrado no *Algoritmo* 2. Vale ressaltar que o procedimento é realizado em uma cópia da solução, sendo a original substituída somente se houver uma melhoria na qualidade e cada tentativa falha de aprimorar um jogador que não pode ser mais melhorado resulta na adição de um contador. Esse contador é utilizado no treinamento personalizado, que só será realizado se o jogador alcançar um número predefinido de tentativas falhas. Após o treinamento, a solução final é codificada novamente em vetor de chaves aleatórias, seguindo dessa forma nas demais etapas e fomentando o algoritmo Biased Random Key Golden Ball (BRKGB).

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

3.2. BRKGB-Instanced

O Biased Random Key Golden Ball Instanced (BRKGB-I) é uma nova variação baseada no método mencionado na seção 3.1, herdando todas as suas características. No entanto, a premissa do BRKGB-I é ampliar a exploração no espaço de busca para encontrar soluções mais eficientes e manter a consistência nos resultados.

Além disso, o método realiza múltiplas buscas simultâneas, executando os processos de forma paralela e independente umas das outras. Ao final do BRKGB-I, a solução mais eficiente entre todas as instâncias é selecionada como a solução para o problema. Vale notar que a quantidade de instâncias criadas pelo algoritmo pode ser controlada por um parâmetro T, onde T é pelo menos 1 e no máximo igual ao número de threads que a máquina fornece. O *Algoritmo* 3 descreve o processo do BRKGB-I.

Algorithm 3 Pseudocódigo do BRKGB-I

```
1: brkgb-instancias[T]
2: for each Ti in brkgb-instancias do
3:     Thread(Ti.ExecutarProcesso())
4: end for
5: Esperar T threads do brkgb concluir
6: melhorFitness ← ∞
7: atualFitness ← ∞
8: for each Ti in brkgb-instancias do
9:     atualFitness ← Ti.melhorFitness
10:    if atualFitness < melhorFitness then
11:        melhorFitness ← atualFitness
12:        melhorSolucao ← Ti
13:    end if
14: end for
```

3.3. BRKGB-World Championship

A próxima meta-heurística apresentada neste artigo é o *Biased Random Key Golden Ball World Championship* (BRKGB-WC). Similar ao BRKGB-I, esse método utiliza diversas instâncias do BRKGB para ampliar a exploração do espaço de busca e encontrar soluções de melhor qualidade. No entanto, o BRKGB-WC simula um torneio mundial de clubes de futebol, onde diversas equipes competem por uma vaga em um campeonato global, culminando em um único campeão: o melhor time mundial.

O método mencionado cria T instâncias do BRKGB, representando campeonatos por continente que ocorrem de forma simultânea e independente. Além disso, o número de torneios pode ser controlado por um parâmetro T, sendo pelo menos 1 e no máximo igual ao número de *threads* para processamento que a máquina de execução fornece.

Todavia, o melhor time de cada continente é selecionado para a população de um torneio mundial, uma nova instância do BRKGB, que contém apenas os melhores de cada campeonato continental. Assim, o processo do algoritmo elege a melhor equipe (campeão dos campeões) e o melhor jogador, cuja solução é considerada a melhor para o problema. O *Algoritmo* 4 apresenta o pseudocódigo do BRKGB-WC.

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

Algorithm 4 Pseudocódigo do BRKGB-WC

```
brkgb-instancias[T]
for each Ti in brkgb-instancias do
    Thread(Ti.ExecutarProcesso())
end for
Esperar os processos finalizar
camp-mundial[T]
idx ← 0
while idx < brkgb-instancias.tamanho do
    camp-mundial[idx] = brkgb-instancias[idx].melhorTime
    idx ← idx + 1
end while
camp-mundial.ExecutarProcesso()
campeao = camp-mundial.melhorSolucao
```

3.4. BRKGB-BFC

A última vertente apresentada nesse artigo é a *Biased Random Key Golden Ball Brazilian Football Championship* (BRKGB-BFC). Este algoritmo é inspirado no campeonato brasileiro de futebol conhecido como "Brasileirão". A técnica utiliza diversas instâncias do BRKGB similar aos demais algoritmos. Entretanto, o método mencionado simula as divisões dos times por séries para ranquear as equipes por seções.

Dessa forma, o BRKGB-BFC cria S séries, cada qual representa um algoritmo BRKGB. Portanto, as séries criadas são categorizadas em ordem alfabética, começando por "A"como a melhor, ou seja, os times com melhor qualidade formam a série "A". As séries subsequentes conterão equipes com qualidade inferior, tal qual o Brasileirão em séries "A", "B"e "C". No início, os times são distribuídos aleatoriamente nas séries, cada um contendo NT equipes. Conforme a temporada de jogos avança, as tabelas de classificação das séries são formadas. Ao fim de cada temporada, os times da metade inferior de uma série, exceto a última, são rebaixados para a séria posterior, enquanto os times da metade superior da tabela, com exceção da série "A", são promovidos para um nível melhor. O *Algorithm* 5 demonstra o processo do método mencionado.

Algorithm 5 Pseudocódigo do BRKGB-BFC

```
series[S]
Inicialização dos times por série
repeat
    for each Si in series do
        Thread(Si.BrkgbProcesso())
    end for
    Esperar a temporada das series finalizar
    Trocar times de series
until critério de parada
melhorTime ← series.melhorSerie.melhorTime
melhorSolucao ← melhorTime.melhorJogador
```

Além disso, vale notar também que as séries ocorrem em paralelo e a quantidade de séries é controlado por um parâmetro denominado S. O valor de S deve ser pelo menos 2, e no máximo a quantidade de *threads* suportada pela máquina que executa o algoritmo.

O processo continua até que as seguintes condições sejam satisfeitas: não ocorrer melhoria no capitão da melhor equipe da série "A", não acontecer melhorias em nenhuma série, e não existir melhoria em nenhuma equipe (convergência). Quando essas condições são atendidas, o algoritmo finaliza, e a solução para o problema em questão é o capitão da melhor equipe da série "A".

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

4. Resultados

Esta seção é destinada a apresentar os resultados dos testes realizados com as estratégias meta-heurísticas. Os métodos propostos foram comparados com os algoritmos BRKGB e BRKGA em 25 instâncias do TSP, cujas amostras foram coletadas do repositório de benchmarking TSPLIB [Reinelt, 1991] e organizadas em ordem crescente (dificuldade/tamanho), conforme o número de vértices, que está refletido no nome da instância. A máquina utilizada para realizar os testes possui processador AMD Ryzen 5 3500U (4 núcleos, 8 *threads*) e 8 GB de memória RAM.

A configuração de parâmetros para o BRKGA: $NP = 48$ (tamanho da população), $P_e = 20\%$ (fração da população para o conjunto de elites), $P_m = 10\%$ (fração da população para o conjunto de mutantes) e $ProbE = 70\%$ (probabilidade de que a prole herde um alelo do pai elite). Da mesma forma, a configuração para os métodos, BRKGB, BRKGB-I, BRKGB-WC e BRKGB-BFC: $Nt = 4$ (o número de times), $Np = 12$ (número de jogadores por time), $Ni = 4$ número de instâncias, $Ns = 4$ (número de séries), $ProbE = 85\%$ (probabilidade de que a prole herde um alelo do pai elite). Além disso, é importante destacar que os parâmetros são selecionados com base em testes preliminares realizados com diversas configurações, visando identificar o conjunto de parâmetros que oferece o melhor equilíbrio entre custo, qualidade e desempenho.

A tabela 1 apresenta os resultados para 20 execuções de testes computacionais em 25 amostras. Os atributos escolhidos foram: a média da qualidade da solução (Avg), O valor percentual que representa a diferença necessária para atingir a qualidade máxima (Gap) conforme TSPLIB [Reinelt, 1991], e o tempo médio de execução ($Time$). A tabela 2 descreve uma análise qualitativa entre os métodos de acordo com a relação dos atributos da tabela 1.

Instância	BRKGB		BRKGA		BRKGB-I		BRKGB-WC		BRKGB-BFC	
Nome	Avg/Gap	Time	Avg/Gap	Time	Avg/Gap	Time	Avg/Gap	Time	Avg	Time
oliver30	424.672/0.2	0.02305	478.185/11.3	0.14885	423.887/0.03	0.0648	423.949/0.04	0.148	**423.741/0.0**	0.27125
berlin52	7727.89/2.4	0.1386	10245/26.38	0.2645	7656.03/1.4	0.37555	7669.33/1.66	0.74345	**7608.61/0.87**	1.0745
eil51	438.75/2.90	0.1259	556.368/23.4	0.3977	436.324/2.36	0.31375	436.306/2.36	0.67645	**434.457/1.94**	0.9718
st70	693.818/2.71	0.51615	1110.7/39.2	0.6273	688.498/1.96	1.4137	689.073/2.04	3.1438	**685.502/1.53**	3.7401
pr76	110019/1.69	0.41115	182387/68.6	0.37085	109481/1.20	1.3466	109817/1.50	2.5943	**109430/1.16**	3.3219
kroA100	21848.5/2.59	1.09965	49373.8/56.8	0.58275	21718.9/2.01	3.3595	21662.3/1.75	6.97735	**21617.8/1.55**	7.92785
kroB100	22846.5/3.08	1.07035	49344.1/55.1	0.61605	22729.9/2.59	3.0791	22755.9/2.70	7.06435	**22616.4/2.1**	8.3029
kroC100	21409/3.08	1.2456	48029.6/56.7	0.59325	21296/ 2.5	3.2253	21241.8/2.3	7.0933	**21079.1/1.56**	7.99245
kroD100	22068.9/3.51	1.11965	48518/56.1	0.5521	21913.2/2.82	3.18065	21896.1/2.74	6.873	**21813.4/2.38**	8.41715
kroE100	22796.2/3.19	1.106	49823.7/55.7	0.59065	22686.1/2.72	3.08805	22721/2.87	6.84985	**22577.8/2.25**	8.13355
eil101	667.64/5.78	0.99005	1171.65/46.3	0.66125	666.111/5.57	3.11705	664.885/5.39	6.38075	**659.878/4.67**	7.76345
lin105	14744.2/2.47	1.2463	35012.8/58.9	0.56605	14646.2/1.82	3.6161	14678.7/2.04	7.9172	**14604.3/1.54**	9.3689
pr107	45508.5/2.72	1.21885	136522/208.1	0.57595	45179.7/1.97	3.6709	45294.4/2.23	7.2546	**44931.2/1.41**	8.7457
pr124	60392.4/2.30	1.89445	195996/232.0	0.74835	59842/1.37	6.54795	60019.9/1.67	12.1539	**59817.1/1.33**	14.3096
ch130	6356.28/4.03	2.5355	15600.4/155	0.76795	6307.32/3.22	7.819	6306.18/3.21	15.9297	**6270.14/2.62**	18.996
pr136	101110/4.48	2.83215	266797/175	0.77315	100151/3.49	8.6377	100293/3.63	17.6465	**99668.5/2.99**	20.7911
pr144	59467.1/1.58	3.3061	234182/300	0.83285	59113.5/0.98	10.4858	59167.3/1.07	19.7569	**58920/0.65**	23.594
ch150	6809.32/4.13	3.124	18730.2/186.9	0.9196	6765.76/3.51	22.37	6790.33/3.86	23.1119	**6724.53/2.92**	49.0626
kroA150	27536.6/3.67	3.4066	81072.1/205.6	0.9276	27251.6/2.66	25.1707	27498.8/3.54	26.4151	**27245.7/2.64**	54.2976
kroB150	27169.4/3.82	3.65275	79424.5/203.9	0.9476	26809.7/2.53	26.029	26988.4/3.18	26.5767	**26808.5/2.53**	52.6559
pr152	75576.8/2.57	3.65035	318699/332	0.9142	75363.3/2.28	11.8106	75065.9/1.87	23.8583	**74627.6/1.28**	26.5819
d198	16234.6/2.88	9.51315	57506.3/264	1.31175	16174.8/2.5	34.9642	16163.9/2.43	66.3586	**16137.1/2.26**	72.4386
kroA200	30605.6/4.04	9.06955	113735/287.2	1.3084	30318.1/3.13	64.4398	30529.3/3.8	65.2474	**30280.2/3.01**	113.075
kroB200	30952.5/4.89	8.96755	114205/287.9	1.26475	30586.4/3.75	63.6056	30786.9/4.38	65.7566	**30458.2/3.35**	110.588
pr226	81900.3/1.86	9.7798	537198/568.4	1.4697	81246.2/1.07	79.3342	81381.9/1.24	78.8753	**81185.6/1.0**	161.497
media gap	3.06		158.74		2.37		2.54		2.0	

Tabela 1: Resultados dos métodos BRKGB, BRKGA, BRKGB-I, BRKGB-WC e BRKGB-BFC. Para cada instância, a média dos resultados, o gap em relação ao ótimo do TSPLIB e a média de tempo de execução são mostradas.

A tabela 3 demonstra a comparação do método BRKGB-BFC com BRKGA, utilizando uma população 10 e 100 vezes maior que na tabela 1.O método BRKGB-BFC mostrou vantagem, obtendo melhores resultados em qualidade de solução, o gap em relação ao ótimo do TSPLIB. Em tempo de execução, na maioria das instâncias, o algoritmo BRKGB-BFC apresenta resultados com menores tempos em comparação ao BRKGA 100x, mas ao comparar com o BRKGA 10x, a maioria dos resultados demonstra um tempo maior de execução; contudo, a qualidade da solução

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

Método	Qualidade da Solução	Tempo de Execução
BRKGB	**Alta Qualidade:** O BRKGB consistentemente fornece soluções de alta qualidade na maioria das instâncias, muitas vezes alcançando soluções próximas das melhores. No entanto, para algumas instâncias como pr76 e st70, as soluções não são as melhores comparadas a outros métodos.	**Muito Rápido:** O BRKGB apresenta tempos de execução excepcionalmente rápidos, sendo frequentemente o mais rápido ou um dos mais rápidos em muitas instâncias. Isso o torna um método muito eficiente em termos de velocidade.
BRKGA	**Qualidade Moderada a Baixa:** O BRKGA tende a fornecer soluções de qualidade moderada a baixa em comparação com outros métodos. Para instâncias como pr76 e pr107, as soluções são significativamente inferiores em qualidade.	**Extremamente Rápido:** O BRKGA, consistentemente, mostra os tempos de execução mais rápidos na maioria das instâncias, tornando-o uma excelente escolha quando a velocidade é uma prioridade.
BRKGB-I	**Alta Qualidade:** O BRKGB-I geralmente oferece soluções de alta qualidade, às vezes alcançando a melhor solução entre todos os métodos, como em oliver30. No entanto, nem sempre supera o BRKGB-BFC em todos os casos.	**Rápido a Moderado:** Os tempos de execução do BRKGB-I são geralmente rápidos, mas podem variar significativamente, com algumas instâncias apresentando tempos mais longos, como em d198.
BRKGB-WC	**Alta Qualidade:** O BRKGB-WC fornece soluções de alta qualidade, muitas vezes muito próximas das melhores soluções encontradas por outros métodos. Ele consistentemente apresenta um bom desempenho em diferentes instâncias.	**Moderado a Lento:** Os tempos de execução do BRKGB-WC são de moderados a lentos, frequentemente demorando mais do que o BRKGB e o BRKGA, mas ainda fornecendo uma qualidade de solução competitiva.
BRKGB-BFC	**Melhor Qualidade:** O BRKGB-BFC fornece geralmente as melhores soluções, alcançando os menores valores de solução na maioria das instâncias, tornando-o o método mais confiável para a qualidade da solução.	**Lento:** Os tempos de execução do BRKGB-BFC são tipicamente os mais longos, indicando que, embora forneça as melhores soluções, faz isso à custa da velocidade.

Tabela 2: Análise dos métodos, a partir das informações da tabela 1, baseados nos critérios médios de qualidade e tempo de execução.

obtida é superior. Portanto, considerando a relação custo-benefício, o BRKGB-BFC se revela mais vantajoso. Adicionalmente, vale notar que a última linha das tabelas 1 e 3 apresentam a média dos valores de Gap para todas as instâncias, com o objetivo de avaliar a eficiência do método de forma independente de cada instância.

5. Conclusão e Trabalhos Futuros

Neste estudo, desenvolvemos uma investigação sobre meta-heurísticas do grupo *Nature-Inspired* e *Population-Based*. Como principal contribuição, focamos na introdução de três novas variantes fundamentadas no algoritmo *Golden Ball*. Essas estratégias aplicam a ideia de paralelismo visando aumentar a exploração do espaço de busca na tentativa de encontrar soluções de destaque.

Mediante comparação realizada (tabela 1), assumindo instâncias do TSP, com outra abordagem que aplica os conceitos de chaves aleatórias (BRKGA), concluímos que: para alta qualidade da solução, o BRKGB-BFC é geralmente a melhor escolha, apesar de seu tempo de execução mais longo. Já para rapidez na execução, o BRKGA é o método mais eficiente em termos de tempo, mesmo que sacrifique alguma qualidade na solução.

Considerando uma questão de equilíbrio, o BRKGB e BRKGB-I oferecem uma boa relação entre qualidade da solução e tempo de execução, sendo adequados para casos onde ambos os fatores são importantes. Esta análise qualitativa proporciona uma visão clara das vantagens e desvantagens de cada método, auxiliando na escolha do método mais adequado conforme os requisitos específicos do problema. Finalmente, ao tentar considerar uma população maior para o BRKGA (tabela 3), verificamos o BRKGB-BFC em melhores condições de escolha para todas as instâncias da execução dos testes.

Como trabalhos futuros, pretendemos realizar estudos detalhados sobre a escalabilidade das variações propostas, avaliando o desempenho dos algoritmos em problemas de diferentes escalas e complexidades, bem como em diversos ambientes computacionais, incluindo arquiteturas

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

Instancia	BRKGB-BFC		BRKGA 10x		BRKGA 100x	
Nome	Avg/Gap	Time	Avg/Gap	Time	Avg/Gap	Time
oliver30	**423.741/0.0**	0.27125	438.004/3.25	1.6368	423.978/0.05	14.9092
berlin52	**7608.61/0.87**	1.0745	8590.59/12.2	2.91295	8024.82/6.01	26.2695
eil51	**434.457/1.94**	0.9718	477.773/10.8	3.69945	452.118/5.77	34.4002
st70	**685.502/1.53**	3.7401	848.528/20.4	4.0903	760.875/11.2	36.7998
pr76	**109430/1.16**	3.3219	139421/22.4	2.8818	122204/11.4	27.5919
kroA100	**21617.8/1.55**	7.92785	33734/36.9	6.15005	26443.9/19.5	53.7122
kroB100	**22616.4/2.1**	8.3029	33316.4/33.5	6.18365	27084.2/18.2	53.8668
kroC100	**21079.1/1.56**	7.99245	32759/36.6	6.11645	26280.5/21.04	53.9254
kroD100	**21813.4/2.38**	8.41715	32429.3/34.3	6.0956	25835.4/17.5	54.0552
kroE100	**22577.8/2.25**	8.13355	34370.7/35.7	6.15065	26882.2/17.9	53.8015
eil101	**659.878/4.67**	7.76345	859.675/26.8	6.0539	723.493/13.06	53.708
lin105	**14604.3/1.54**	9.3689	23849.8/39.7	5.52875	18598.3/22.6	58.3056
pr107	**44931.2/1.41**	8.7457	87648.3/49.4	5.78225	62150.6/28.7	61.2431
pr124	**44931.2/1.41**	14.3096	124915/52.7	7.75765	85676/31.1	66.8929
ch130	**6270.14/2.62**	18.996	10466.6/41.6	7.97055	7990.39/23.5	78.8717
pr136	**99668.5/2.99**	20.7911	176679/45.2	8.19285	129388/25.2	87.4326
pr144	**58920/0.65**	23.594	156613/62.6	9.34695	100774/41.9	91.0585
ch150	**6724.53/2.92**	49.0626	12570.6/48.0	8.74935	8923.1/26.8	94.067
kroA150	**27245.7/2.64**	54.2976	54281.3/51.1	8.65945	37721.1/29.6	87.9364
kroB150	**26808.5/2.53**	52.6559	52093.3/49.8	9.3331	36929.8/29.2	87.9068
pr152	**74627.6/1.28**	26.5819	197752/62.7	9.57045	125935/41.4	96.2446
d198	**16137.1/2.26**	72.4386	37870.3/58.3	13.522	23537.9/ 32.9	120.721
kroA200	**30280.2/3.01**	113.075	77231.9/61.9	12.9992	48046.5/38.8	123.449
kroB200	**30458.2/3.35**	110.588	76408.7/61.4	12.8765	48837.7/39.7	123.313
pr226	**81185.6/1.0**	161.497	345187/76.7	14.7649	204006/60.6	140.868
media gap	2.0		41.3		24.5	

Tabela 3: Resultados dos métodos BRKGB-BFC, BRKGA 10X a população, BRKGA 100x a população. Para cada instância, a média dos resultados, o gap em relação ao ótimo do TSPLIB e a média de tempo de execução são mostradas.

paralelas e distribuídas. Além disso, investigaremos algoritmos de treinamento comum mais rápidos para substituir o 2OPT ao executar instâncias grandes.

Referências

Ahmed, Z. H. (2010). Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. *International Journal of Biometrics & Bioinformatics (IJBB)*, 3(6): 96.

Atashpaz-Gargari, E. and Lucas, C. (2007). Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In *2007 IEEE Congress on Evolutionary Computation*, p. 4661–4667.

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA journal on computing*, 6(2):154–160.

Beheshti, Z. and Shamsuddin, S. M. H. (2013). A review of population-based meta-heuristic algorithms. *Int. j. adv. soft comput. appl*, 5(1):1–35.

Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM computing surveys (CSUR)*, 35(3):268–308.

David, E. G. et al. (1989). Genetic algorithms in search. *Optimization and Machine Learning, Reading, Massachusetts*.

Davis, L. et al. (1985). Applying adaptive algorithms to epistatic domains. In *IJCAI*, volume 85, p. 162–164. Citeseer.

Gallego, R., Romero, R., and Monticelli, A. (2000). Tabu search algorithm for network synthesis. *IEEE Transactions on Power Systems*, 15(2):490–495.

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br

Garey, M. R. and Johnson, D. S. (1979). Computers and intractability: A guide to the theory of np-completeness. *W. H. Freeman*, p. 209–217. Discusses various NP-hard problems including the Travelling Salesman Problem (TSP) and the Capacitated Vehicle Routing Problem (CVRP).

Gonçalves, J. F. and Resende, M. G. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525.

Holland, J. H. (1984). Genetic algorithms and adaptation. *Adaptive control of ill-defined systems*, p. 317–333.

Latifoğlu, F. (2013). A novel approach to speckle noise filtering based on artificial bee colony algorithm: An ultrasound image application. *Computer Methods and Programs in Biomedicine*, 111 (3):561–569. ISSN 0169-2607. URL https://www.sciencedirect.com/science/article/pii/S0169260713001661.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *The Bell System Technical Journal*, 44(10):2245–2269.

Moscato, P. and Cotta, C. (2003). A gentle introduction to memetic algorithms. In *Handbook of metaheuristics*, p. 105–144. Springer.

Naddef, D. and Rinaldi, G. (2002). Branch-and-cut algorithms for the capacitated vrp. In *The vehicle routing problem*, p. 53–84. SIAM.

Osaba, E., Díaz, F., and Onieva, E. (2013). Golden ball: A novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts. *Applied Intelligence*, 41:145–166.

Pearl, J. (1984). Heuristics: Intelligent search strategies for computer problem solving. URL https://www.osti.gov/biblio/5127296.

Reinelt, G. (1991). TSPLIB–a traveling salesman problem library. *ORSA Journal on Computing*, 3 (4):376–384.

Rutenbar, R. (1989). Simulated annealing algorithms: an overview. *IEEE Circuits and Devices Magazine*, 5(1):19–26.

Santana, G. A. et al. (2021). Halton-chaos-beta-pso: uma nova abordagem metaheurística baseada em enxame de partículas aplicada à problemas de engenharia elétrica.

Santos, A. G. (2022). Uma meta-heurística adaptativa aplicada ao problema de roteamento de veículos capacitados: estudo de caso de uma transportadora de pacotes. URL https://repositorio.unifesp.br/handle/11600/63539.

Spears, W. M., De Jong, K. A., et al. (1991). On the virtues of parameterised uniform crossover. In *ICGA*, p. 230–236.

Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8:279–292.

Zhang, C., Ouyang, D., and Ning, J. (2010). An artificial bee colony approach for clustering. *Expert Systems with Applications*, 37(7):4761–4767. ISSN 0957-4174. URL https://www.sciencedirect.com/science/article/pii/S0957417409009452.

DOI: 10.59254/sbpo-2024-193754
https://proceedings.science/p/193754?lang=pt-br