Uma Abordagem Baseada em Algoritmos Genéticos para Resolução do Jogo Mastermind: Comparação com Estratégias da Literatura

João Pedro Silva Cunha[1], **Alexandre Tadeu Rossini da Silva**[1]

[1]Curso de Ciência da Computação – Universidade Federal do Tocantins (UFT)

{joao.cunha, arossini}@uft.edu.br

Abstract. *This study proposes a solution to the MasterMind game using Genetic Algorithms (GAs) through a two-stage strategy: identification of the colors in the secret code and determination of their positions. Variations of genetic operators were tested, with emphasis on the Reward Matrix, which implements a collective learning process based on historical memory and exhibits behavior analogous to collective intelligence. The results showed superior performance compared to other strategies, especially in complex scenarios.*

Resumo. *Este estudo propõe uma solução para o jogo MasterMind utilizando Algoritmos Genéticos (AGs) em uma estratégia de duas etapas: identificação das cores do código secreto e determinação de suas posições. Foram testadas variações nos operadores genéticos, com destaque para a Matriz de Recompensa, que adota um processo de aprendizado coletivo baseado em memória histórica e evidencia um comportamento análogo à inteligência coletiva. Os resultados mostraram desempenho superior em relação a outras estratégias, especialmente em cenários complexos.*

1. Introdução

A busca por soluções ótimas e eficientes para problemas complexos tem sido uma constante na história da ciência e da computação. Jogos de lógica e estratégia, como o MasterMind, apresentam desafios para áreas como inteligência artificial e otimização. Criado por Mordecai Meirowitz em 1970 [Abreu and Mendes 2008], o MasterMind é um jogo de tabuleiro que envolve dois jogadores: o codificador, que cria um código secreto com uma combinação de cores, e o decodificador, que tenta descobri-lo dentro de um número limitado de tentativas. A cada tentativa, o codificador fornece pistas por meio de pinos coloridos, onde a cor branca indica um elemento correto, porém em posição errada, e a cor preta sinaliza um elemento correto tanto na cor quanto na posição.

Do ponto de vista computacional, o MasterMind representa um problema de busca e otimização, no qual o objetivo é encontrar a sequência correta de cores dentro de um espaço de busca que cresce exponencialmente com o aumento do número de cores e do comprimento da sequência [Oijen 2018]. O espaço de busca total, representando todas as combinações possíveis de códigos secretos, é definido como N^P. Por exemplo, em cenários simples, como $N = 6$ cores e $P = 4$ posições, há $6^4 = 1.296$ combinações possíveis. No entanto, ao ampliar N para 8 e P para 6, esse número cresce exponencialmente para $8^6 = 262.144$, tornando inviável o uso de abordagens exaustivas para encontrar a solução para valores elevados de P e N.

Essa complexidade foi amplamente discutida na literatura, como nos trabalhos de [Berghman et al. 2009] e [Maestro-Montojo et al. 2014], os quais enfatizam que, embora métodos heurísticos reduzam o esforço de busca, o crescimento exponencial do espaço de soluções continua sendo um desafio fundamental no jogo MasterMind. Ressalta-se que, além do potencial de otimização, abordagens baseadas em Algoritsmo Genéticos (AGs) também revelam um paradigma interessante de colaboração entre agentes computacionais. Ademais, este trabalho propõe o uso de um processo de aprendizado coletivo baseado em memória histórica, denominado Matriz de Recompensa, o qual evidencia um comportamento análogo à inteligência coletiva presente em sistemas colaborativos.

2. Trabalhos Relacionados

A aplicação de AGs ao jogo MasterMind foi explorada em diversos trabalhos. O problema foi formalizado como um desafio de otimização restrita por [Bernier et al. 1996], que comparou AGs com *simulated annealing*. Uma abordagem adaptativa foi proposta por [Bento et al. 1999], utilizando operadores de cruzamento e mutação condicionados. Estratégias otimizadas para cenários mais complexos foram exploradas por [Kalisker and Camens 2003], enquanto [Merelo-Guervós et al. 2006] introduziu uma função de aptidão baseada na proximidade das pistas fornecidas. Uma abordagem ágil, com baixo tempo de execução e número reduzido de tentativas, foi apresentada por [Berghman et al. 2009]. A escalabilidade dos AGs foi investigada por [Merelo et al. 2011], que introduziu o conceito de "endgames"para reduzir o espaço de busca. Soluções evolutivas escaláveis foram exploradas por [Merelo et al. 2013], utilizando o algoritmo Evo10, que reduziu o número de avaliações necessárias para encontrar a solução, mostrando-se produtivo em cenários complexos e permitindo aplicações em tempo real, como em jogos para dispositivos móveis. Uma abordagem hierárquica foi proposta por [Maestro-Montojo et al. 2014], combinando duas técnicas evolutivas para selecionar o melhor código jogável.Variações de algoritmos, incluindo a busca local genética, foram examinadas por [Oijen 2018], que combinou heurísticas locais com AGs, demonstrando eficiência em cenários de maior complexidade e reduzindo o tempo de execução sem comprometer a qualidade das soluções.

3. Estratégia em duas etapas para o jogo MasterMind com AGs

A solução desenvolvida neste trabalho foi dividida em duas etapas: identificação das cores e determinação das posições. Na primeira etapa, o AG identificou as cores do código secreto sem considerar suas posições, enquanto na segunda, determinou a posição exata de cada cor. Essa abordagem progressiva permitiu explorar o espaço de busca de forma eficiente, começando pelas cores e refinando as posições.

3.1. Primeira etapa: identificação das cores presentes no código secreto

A população inicial foi composta por cromossomos com um número de genes igual ao número de posições no código secreto P. Cada gene representa uma cor, selecionada de um conjunto de N alelos possíveis, e foi gerado aleatoriamente. A aptidão de cada cromossomo foi calculada com base no número de cores corretas independentemente de suas posições, normalizada em uma escala de 0 a 1. Quando atingiu 1.00, todas as cores do código secreto foram identificadas (ainda que não necessariamente nas posições corretas). A seleção foi realizada por meio do método da *roleta*, que atribuiu a cada indivíduo uma

probabilidade de ser escolhido proporcional à sua aptidão, ou seja, priorizou cromossomos com maior aptidão.

O cruzamento realizou a combinação dos genes de dois cromossomos selecionados em um ou dois pontos de corte, com uma proporção definida entre os dois métodos. Essa abordagem promoveu a criação de novas combinações de indivíduos, aumentando as chances de identificar corretamente todas as cores no código secreto. Após o cruzamento, aplicou-se o operador de mutação, introduzindo variações adicionais substituindo aleatoriamente uma cor em um cromossomo para melhorar a diversidade da população e evitar que o algoritmo ficasse preso em uma solução subótima. A função mutação garantiu que o gene alterado no cromossomo não fosse idêntico ao gene original, assegurando que o cromossomo resultante fosse diferente do anterior.

Para garantir que as melhores soluções fossem preservadas e transmitidas para a próxima geração, o algoritmo empregou a estratégia de substituição por elitismo. Especificamente, a implementação da substituição por elitismo classificou uma população combinada (formada pela geração atual e pela nova geração produzida) e selecionou os cromossomos com maior aptidão até atingir o tamanho da população inicial. Dessa forma, os cromossomos com aptidão mais elevada, que se aproximaram mais da solução ótima, sobreviveram e passaram para a próxima geração. O processo repetiu-se até encontrar um cromossomo com aptidão máxima ou atingir o limite de gerações, quando os melhores indivíduos foram encaminhados para a segunda etapa.

3.2. Segunda etapa: determinação das posições das cores

Uma característica fundamental da segunda etapa foi a reutilização dos cromossomos que atingiram a aptidão máxima na primeira fase como forma de manutenção do fluxo de geração de uma nova população. Esses indivíduos foram mantidos e utilizados na inicialização da população da segunda etapa, fornecendo um ponto de partida otimizado.

Durante a geração inicial dos novos cromossomos, os genes foram constituídos, de maneira aleatória, exclusivamente com base nas cores contidas nos genes dos indivíduos recuperados da primeira fase, o que restringiu ainda mais o espaço de busca e garantiu que apenas combinações válidas de cores fossem consideradas. Na segunda etapa, a avaliação de aptidão diferiu da primeira, pois considerou apenas a posição exata dessas cores no código secreto. Assim, o algoritmo utilizou os "pinos pretos" para avaliar a aptidão dos indivíduos, indicando a quantidade de genes cujos valores coincidiram com os respectivos valores e posições no código secreto. A função percorre cada cromossomo da população e, para cada gene, verifica se o valor do gene coincide com o valor correspondente na posição do código secreto. Ao final dessa verificação, é feita uma contagem normalizada pela quantidade total de genes no cromossomo, resultando em um valor de aptidão que variou entre 0 e 1, exatamente como ocorreu na primeira etapa. O valor de aptidão é, então, armazenado no atributo aptidão de cada cromossomo e, posteriormente, utilizado nas etapas de seleção e substituição. Os operadores de seleção e cruzamento na segunda etapa foram idênticos aos da primeira etapa, mantendo o método da roleta e o cruzamento em um ou dois pontos. O operador de mutação seguiu o mesmo princípio, porém com uma restrição importante: as cores utilizadas na substituição aleatória foram limitadas exclusivamente àquelas identificadas como presentes no código secreto durante a primeira etapa, o que preservou a validade das soluções enquanto explorou diferentes arranjos posicionais.

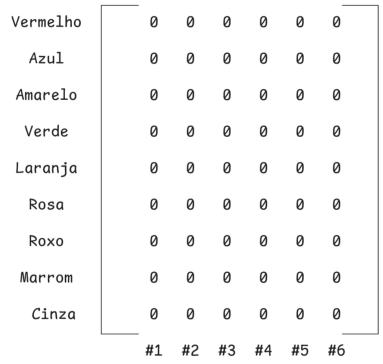

Figura 1. Matriz de recompensa representando as cores disponíveis e suas posições possíveis.

A substituição no algoritmo genético, fundamentada no elitismo, assegurou a preservação dos cromossomos mais aptos. No entanto, à medida que a complexidade do problema aumentou, a diversidade genética tende a estagnar, comprometendo o progresso do algoritmo. Para contornar esse desafio, foram implementadas quatro estratégias de substituição, nomeadas de: Gen, GenPlus, Matriz de Recompensa e Matriz de Recompensa Plus.

As estratégias de substituição implementadas buscaram contornar a estagnação do algoritmo genético, com destaque para a Matriz de Recompensa Plus, que superou as demais em termos de desempenho e convergência, especialmente em cenários complexos. Enquanto as estratégias Gen e GenPlus focaram em reintroduzir diversidade na população por meio de substituições parciais e critérios históricos de aptidão, a Matriz de Recompensa e Matriz de Recompensa Plus trouxeram um mecanismo baseado em princípios de aprendizagem por reforço, introduzindo um ajuste situacional na aplicação das penalizações e recompensas. Para isso, essa técnica utiliza uma tabela para armazenar informações sobre a adequação de cada gene em posições específicas, baseando-se no desempenho histórico dos cromossomos ao longo das gerações. Essa tabela direciona a geração de novos cromossomos, otimizando a busca por combinações que atendam aos critérios desejados. A matriz é estruturada como um dicionário, onde cada cor é uma chave, e seu valor é uma lista de pontuações correspondentes às posições do cromossomo. Inicialmente, todas as pontuações são zeradas, conforme exemplo apresentado na Figura 1 para um cenário hipotético $N = 9$ e $P = 6$.

Para a estratégia nominada Matriz de Recompensa, a tabela é atualizada com base nos resultados obtidos na avaliação de aptidão, conforme descrito a seguir, e a Figura 2a apresenta um exemplo.

- Recompensa positiva: Quando um cromossomo recebe uma pontuação de pinos pretos, todos os genes têm suas posições correspondentes incrementadas proporcionalmente à quantidade de pinos pretos atribuída ao cromossomo.
- Recompensa negativa: Quando um cromossomo recebe uma pontuação de pinos brancos, as posições correspondentes aos genes parcialmente corretos são penalizadas proporcionalmente, subtraindo unidades homogêneas à quantidade de pinos brancos atribuída ao cromossomo.

Para a estratégia nomeada Matriz de Recompensa Plus, a atualização das

pontuações é aplicada com base em condições situacionais em relação ao valor de pinos pretos (presença ou ausência) alcançado pelo cromossomo (os pinos brancos são ignorados). O processo considera o número de pinos pretos atribuídos ao cromossomo avaliado, que indica a quantidade de genes corretamente posicionados. A recompensa positiva é igual à da estratégia Matriz de Recompensa, porém a recompensa negativa é diferente, conforme apresentado a seguir e ilustrado na Figura 2b, que apresenta um exemplo.

- Recompensa negativa: Caso o cromossomo não receba nenhum pino preto, todos os genes em suas respectivas posições têm uma unidade subtraída. Esse método castiga configurações improdutivas sem comprometer excessivamente as possibilidades futuras, evitando a estagnação da busca.

Nas duas estratégias baseadas em matriz de recompensa, após a avaliação da aptidão dos cromossomos da população, as recompensas positivas e negativas são aplicadas.

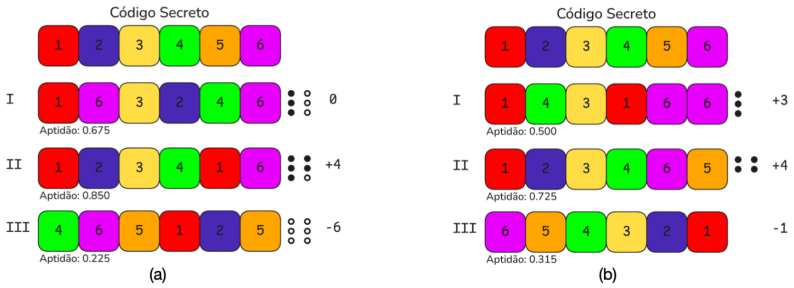

Figura 2. Exemplo de cromossomos avaliados.

4. Testes e Resultados

Foram realizados testes em dez cenários, variando o número de posições (P) e cores (N), desde configurações simples $\{P = 4, N = 6\}$ até complexas $\{P = 8, N = 12\}$. Cada cenário foi executado 500 vezes, com parâmetros fixos: população de 150 indivíduos, 15 gerações máximas, taxa de mutação de 0.1 e cruzamento de 0.5. Na primeira etapa, foram identificadas as cores presentes no código secreto, o algoritmo mostrou-se competente em todos os cenários. A média de gerações necessárias aumentou gradualmente com a complexidade, variando de 1.06 $\{P = 4, N = 6\}$ a 5.94 $\{P = 8, N = 12\}$. A flexibilidade desta etapa contribuiu para o bom desempenho, estabelecendo uma base sólida para a segunda etapa. Na segunda etapa, que determinou a posição exata das cores, foram testadas quatro variações do algoritmo: Gen, GenPlus, Matriz de Recompensa e Matriz de Recompensa Plus. A Matriz de Recompensa Plus destacou-se, com os melhores resultados em todos os cenários, especialmente nos mais complexos ($\{P = 7, N = 10\}$ e $\{P = 8, N = 12\}$), onde o Gen não convergiu em algumas execuções. A média de gerações variou de 1.00 $\{P = 4, N = 6\}$ a 2.35 $\{P = 8, N = 12\}$.

A Tabela 1 compara os resultados obtidos com os dos trabalhos relacionados. A estratégia Matriz de Recompensa Plus apresentou o melhor desempenho, com médias inferiores às reportadas na literatura, especialmente em cenários complexos. Essa comparação reforça a eficácia da abordagem proposta, que superou consistentemente as soluções existentes. A Figura 3 resume graficamente os testes. Conclui-se que, mesmo em cenários de maior complexidade combinatória, a solução nominada de Matriz de Recompensa Plus apresentou desempenho superior em diferentes configurações testadas.

Tabela 1. Comparação dos resultados com os trabalhos relacionados

Algoritmo	$P=4, N=6$	$P=4, N=8$	$P=5, N=7$	$P=5, N=8$	$P=5, N=9$	$P=6, N=6$	$P=6, N=9$	$P=6, N=10$	$P=7, N=10$	$P=8, N=12$
[Bernier et al. 1996]	-	-	5.08	-	-	5.62	-	7.00	-	-
[Bento et al. 1999]	-	-	-	7.54	-	-	-	-	-	-
[Kalisker and Camens 2003]	4.75	-	-	6.39	-	-	7.27	-	8.62	-
[Merelo-Guervós et al. 2006]	4.13	-	-	5.90	-	-	-	-	-	-
[Berghman et al. 2009]	4.39	-	-	5.61	-	-	6.47	-	-	8.37
[Merelo et al. 2011]	4.44	5.24	-	5.77	6.12	-	-	-	-	-
[Merelo et al. 2013]	-	5.21	-	5.65	6.01	-	6.50	6.88	7.42	-
[Maestro-Montojo et al. 2014]	-	-	-	5.55	-	-	6.37		-	-
[Oijen 2018]	4.32	-	-	-	-	-	6.61	-	-	8.58
AG: Gen	2.06	2.52	3.08	3.51	3.85	4.97	6.51	7.42	*	*
AG: GenPlus	2.06	2.52	3.15	3.48	3.92	4.09	5.80	6.47	7.88	11.45
AG: Matriz de Recompensa	2.06	2.52	2.96	3.45	3.95	3.16	6.19	6.92	9.46	12.95
AG: Matriz de Recompensa Plus	2.06	2.52	2.70	3.15	3.51	2.94	4.63	5.37	6.01	8.29

P: posições para cores; N: cores disponíveis; O símbolo '-' indica que não foi executado o cenário para o algoritmo; * O algoritmo não encontrou solução para as 500 execuções realizadas.

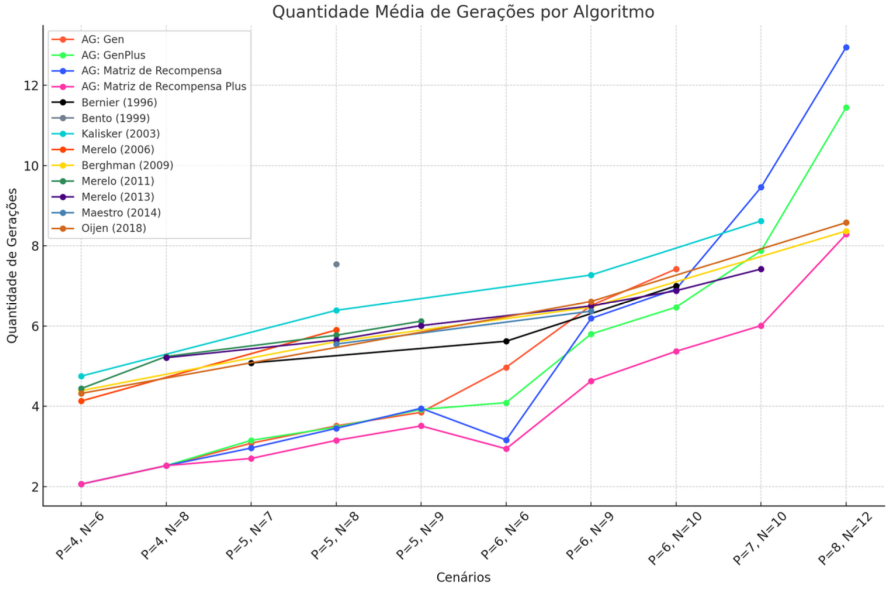

Figura 3. Comparação da quantidade média de gerações necessárias para resolver cenários do jogo Mastermind.

5. Considerações Finais

Este estudo investigou a eficiência de algoritmos genéticos na resolução do jogo MasterMind, utilizando uma abordagem experimental e comparativa. A solução proposta dividida em duas etapas permitiu uma exploração gradual e eficiente do espaço de busca. A estratégia da Matriz de Recompensa Plus destacou-se, superando não apenas as outras variações testadas, mas também as soluções encontradas na literatura, como as propostas por [Bernier et al. 1996, Kalisker and Camens 2003, Merelo et al. 2013]. Dessa forma, os achados desta pesquisa fornecem uma contribuição relevante.

Ao interpretar essa abordagem sob a ótica da colaboração entre agentes computacionais, evidencia-se que o modelo empregado transcende a otimização. Cada cromossomo atua como um agente autônomo, compartilhando conhecimento de forma distribuída, enquanto o uso da Matriz de Recompensa representa um processo de aprendizado coletivo baseado em memória histórica. Esses elementos reforçam o potencial de integração de princípios de inteligência coletiva em sistemas colaborativos adaptativos. Além disso, a divisão do problema em duas fases remete a um modelo de repartição de tarefas colaborativas, característico de ambientes de resolução coletiva de problemas.

Referências

Abreu, P. and Mendes, P. (2008). Mastermind: an augment reality approach: [porting a legacy game to new interaction paradigms]. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, pages 205–210.

Bento, L., Pereira, L., and Rosa, A. (1999). Mastermind by evolutionary algorithms. In *Proceedings of the 1999 ACM symposium on Applied computing*, pages 307–311.

Berghman, L., Goossens, D., and Leus, R. (2009). Efficient solutions for mastermind using genetic algorithms. *Computers & operations research*, 36(6):1880–1885.

Bernier, J. L., Herráiz, C. I., Merelo, J., Olmeda, S., and Prieto, A. (1996). Solving mastermind using gas and simulated annealing: a case of dynamic constraint optimization. In *Parallel Problem Solving from Nature—PPSN IV: International Conference on Evolutionary Computation—The 4th International Conference on Parallel Problem Solving from Nature Berlin, Germany, September 22–26, 1996 Proceedings 4*, pages 553–563. Springer.

Kalisker, T. and Camens, D. (2003). Solving mastermind using genetic algorithms. In *Genetic and Evolutionary Computation Conference*, pages 1590–1591. Springer.

Maestro-Montojo, J., Salcedo-Sanz, S., and Merelo, J. (2014). New solver and optimal anticipation strategies design based on evolutionary computation for the game of mastermind. *Evolutionary Intelligence*, 6:219–228.

Merelo, J. J., Cotta, C., and Mora, A. (2011). Improving and scaling evolutionary approaches to the mastermind problem. In *European Conference on the Applications of Evolutionary Computation*, pages 103–112. Springer.

Merelo, J.-J., Mora, A. M., Castillo, P. A., Cotta, C., and Valdez, M. (2013). A search for scalable evolutionary solutions to the game of mastermind. In *2013 IEEE Congress on Evolutionary Computation*, pages 2298–2305. IEEE.

Merelo-Guervós, J., Castillo, P., and Rivas, V. (2006). Finding a needle in a haystack using hints and evolutionary computation: the case of evolutionary mastermind. *Applied Soft Computing*, 6(2):170–179.

Oijen, V. v. (2018). Genetic algorithms playing mastermind. B.S. thesis.