**UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE FÍSICA DE SÃO CARLOS**

Larissa Fernandes de Aquino

**Algoritmos bio e socio-inspirados na busca pelos
máximos globais de relevos adaptativos**

São Carlos

2020

Larissa Fernandes de Aquino

**Algoritmos bio e socio-inspirados na busca pelos
máximos globais de relevos adaptativos**

Dissertação apresentada ao Programa de Pós-Graduação em Física do Instituto de Física de São Carlos da Universidade de São Paulo, para obtenção do título de Mestre em Ciências.

Área de concentração: Física Básica

Orientador: Prof. Dr. José Fernando Fontanari

Versão corrigida
(Versão original disponível na Unidade que aloja o Programa)

São Carlos
2020

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Aquino, Larissa Fernandes de
Algoritmos bio e socio-inspirados na busca pelos
máximos globais de relevos adaptativos / Larissa
Fernandes de Aquino; orientador José Fernando Fontanari -
versão corrigida -- São Carlos, 2020.
66 p.

Dissertação (Mestrado - Programa de Pós-Graduação em
Física Básica) -- Instituto de Física de São Carlos,
Universidade de São Paulo, 2020.

1. Problemas de otimização. 2. Algoritmos de busca
estocásticos. 3. Modelo NK. 4. Algoritmos evolucionários.
5. Aprendizado por imitação. I. Fontanari, José Fernando,
orient. II. Título.

AGRADECIMENTOS

Ao meu orientador, Dr. José Fernando Fontanari, pelo conhecimento compartilhado, apoio e extrema prestatividade.

Ao Dr. Sandro M. Reia por toda a ajuda com as simulações, discussões relacionadas à pesquisa e aconselhamentos sobre a vida acadêmica.

Aos meus pais por todo o investimento em minha educação, apoio a minha carreira e confiança na minha capacidade.

Aos colegas de graduação e mestrado pelo suporte emocional, aprendizado e bons momentos compartilhados, em especial à Adriane Leal, Edyvania Emily e Eder Sanchez, que me acompanharam tão de perto ao longo do mestrado.

Aos professores e funcionários do Instituto de Física de São Carlos (IFSC) por tornarem possível a realização de inúmeras pesquisas de qualidade no instituto, mesmo em meio às adversidades. Agradeço especialmente ao Dr. Diogo O. Soares-Pinto pela qualidade das disciplinas e suporte ao aprendizado da turma nas disciplinas de Mecânica Quântica e Mecânica Estatística.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro a esse trabalho.

“Alone we can do so little; together we can do so much.”
(Helen Keller)

RESUMO

AQUINO, L. F. **Algoritmos bio e socio-inspirados na busca pelos máximos globais de relevos adaptativos**. 2020. 66p. Dissertação (Mestrado em Ciências) - Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2020.

Algoritmos de otimização que imitam processos naturais, ditos algoritmos bio-inspirados, prometem encontrar soluções ótimas ou quase-ótimas de problemas de otimização sem utilizar praticamente nenhuma informação acerca desses problemas. Outros exemplos bem menos conhecidos de algoritmos de propósito geral são os algoritmos que imitam processos sociais ou culturais. Isso parece ser paradoxal, já que o modo como humanos solucionam problemas difíceis é exatamente através da colaboração em forças-tarefas e, portanto, algoritmos que imitam a forma como interagimos deveriam ser tão ou mais eficientes e populares quanto os algoritmos bio-inspirados. Nessa dissertação vamos estudar o desempenho de um algoritmo baseado na interação social, o aprendizado por imitação, e de dois tipos de algoritmos evolucionários, o algoritmo genético sexuado e o assexuado, na busca pelo máximo global de relevos de *fitness* ou relevos adaptativos gerados pelo modelo NK. A variação dos parâmetros N e K desse modelo permite a geração de relevos com diferentes dimensionalidades e rugosidades. A medida de desempenho dos algoritmos é proporcional ao número total de atualizações realizadas nos agentes até que o máximo global seja encontrado. A base de comparação é o desempenho da busca cega, onde os agentes são atualizados de modo completamente aleatório. Encontramos que mesmo para relevos sem máximos locais, os algoritmos evolucionários não têm um desempenho muito superior ao da busca cega devido, provavelmente, à estocasticidade do operador de seleção que nem sempre seleciona o agente de maior *fitness*. Como no aprendizado por imitação a escolha do agente de maior *fitness*, o agente modelo, é determinística o seu desempenho é muito superior ao dos algoritmos evolucionários nesse tipo de relevo. O preço a pagar é que o aprendizado por imitação é muito mais suscetível a ficar preso em máximos locais do que os outros algoritmos, gerando desempenhos catastróficos que lembram o fenômeno do ‘pensamento de grupo’ da psicologia social. Entretanto, se os parâmetros do aprendizado por imitação forem ajustados de forma ótima, esse algoritmo é superior aos outros também na busca em relevos rugosos. Finalmente, para relevos muito rugosos, encontramos que a busca cega é superior aos algoritmos cooperativos considerados.

Palavras-chave: Problemas de otimização. Algoritmos de busca estocásticos. Modelo NK. Algoritmos evolucionários. Aprendizado por imitação.

ABSTRACT

AQUINO, L. F.. Bio and socio-inspired algorithms in the search for the global maxima of adaptive landscapes .2020. 66p. Dissertação (Mestrado em Ciências) - Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2020.

Optimization algorithms that mimic processes in nature, so-called bio-inspired algorithms, promise to deliver optimal or near-optimal solutions using hardly any information on the optimization problems they are set to tackle. Another much less known class of general-purpose search algorithms are the algorithms that mimic social or cultural processes. This seems paradoxical as the way humans tackle difficult problems is through the cooperative effort in task-forces and so algorithms inspired on the manner they interact should be more or at least as efficient and popular as the bio-inspired algorithms. In this dissertation, we study the performances of a social-inspired algorithm – the imitative learning search – as well as of asexual and sexual variants of evolutionary algorithms in finding the global maxima of fitness or adaptive landscapes generated by the NK model. The variation of the parameters N e K of that model allows the generation of landscapes of different dimensionalities and ruggedness. The performance measure is proportional to the total number of agent updates required by the algorithms to find the global maximum. The baseline performance is set by the blind search, where the agents are updated in a completely random manner. We find that even for landscapes that exhibit a single maximum, the evolutionary algorithms do not perform much better than the blind search due to the stochastic effects of the selection operator, that is not guaranteed to select the fittest agent. Since in the imitative learning the choice of the fittest agent in the population is deterministic, its performance is much superior to those of the other algorithms in this type of landscape. The tradeoff is that the imitative learning is more prone to be trapped in local maxima than the evolutionary algorithms, leading to catastrophic performances that are reminiscent of the groupthink phenomenon of social psychology. However, if the parameters of the imitative learning are finely tuned, it outperforms the other algorithms in the case of rugged landscapes as well. Finally, for too rugged landscapes, the blind search beats all the cooperative algorithms considered.

Keywords: Optimization problems. Stochastic search algorithms. NK model. Evolutionary algorithms. Imitative learning.

LISTA DE FIGURAS

- Figura 1 – Esquema mostrando a vizinhança adjacente à direita de cada uma das componentes de sequências binárias de tamanho $N = 6$ para valores distintos de K . Nota-se que no caso $K = N - 1$ cada componente sofre influência de todas as outras componentes. 25
- Figura 2 – Número médio de máximos em função do parâmetro de epistasia K para diferentes valores de N como indicado. Cada símbolo representa a média sobre 3000 relevos de *fitness* distintos. As barras de erro são menores que o tamanho dos símbolos. 27
- Figura 3 – *Fitness* média das configurações $\vec{x}$ que diferem de $D_H(\vec{x}, \vec{x}_{max})$ componentes do máximo global $\vec{x}_{max}$ para $N = 15$. Cada símbolo representa a média sobre todas as configurações distintas que têm a mesma distância de Hamming ao máximo global e sobre 3000 relevos de *fitness* distintos. As barras de erro são menores que o tamanho dos símbolos. 28
- Figura 4 – Distância Hamming média entre os máximos locais e o máximo global em função do parâmetro de epistasia K para os valores de N indicados. Cada símbolo representa o resultado sobre 1000 relevos de *fitness* distintos da média da distância de Hamming de todos os máximos locais ao máximo global. As barras de erro são menores que o tamanho dos símbolos. 29
- Figura 5 – *Fitness* médio dos máximos locais e dos máximos globais em função de N para os parâmetros de epistasia $K = 2$ e $K = N - 1$, como indicado. Cada símbolo representa o resultado da média sobre 1000 relevos de *fitness* distintos. As barras de erro são menores que o tamanho dos símbolos. 29
- Figura 6 – Esquema da implementação dos algoritmos de busca para a formulação em que a mutação é independente dos processos que definem os algoritmos cooperativos. A busca cega ocorre para $u = 1$ 34
- Figura 7 – Esquema do movimento elementar ou mutação em uma sequência de $N = 5$ componentes utilizado na busca cega pelos máximos globais de relevos NK. 36
- Figura 8 – Comparação entre a aproximação analítica da função distribuição acumulada (linhas cheias) e os resultados das simulações (símbolos) da busca cega para $M = 1, 10, 100$ como indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas em um único relevo com $N = 12$ e $K = 0$. Os resultados não dependem do parâmetro de epistasia. 39

Figura 9 – Custo computacional médio do AGA em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.	40
Figura 10 – Custo computacional médio do AGA em função do número de agentes M para os valores da probabilidade de mutação u indicados. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.	42
Figura 11 – Esquema do algoritmo genético sexuado (AGS) mostrando os operadores de seleção e de recombinação utilizados para criar agentes da nova geração. Em média $(1 - u)M$ novos agentes são criados segundo esse procedimento.	43
Figura 12 – Custo computacional médio do AGS em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.	43
Figura 13 – Custo computacional médio do AGS em função do número de agentes M para os valores da probabilidade de mutação u indicados. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.	44
Figura 14 – Esquema do algoritmo aprendido por imitação (AI) mostrando o processo de cópia de um único bit do agente modelo pelo agente imitador. Em média $(1 - u)M$ novos agentes são criados segundo esse procedimento.	45
Figura 15 – Custo computacional médio do AI em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.	46

Figura 16 – Custo computacional médio do AI em função do número de agentes M para os valores da probabilidade de mutação u indicados. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.	48
Figura 17 – Custo computacional médio dos quatro algoritmos de busca em função do número de agentes M para a probabilidade de mutação u indicada. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. A dimensionalidade do relevo é $N = 12$ e o parâmetro de epistasia é $K = 0$, ou seja, não há máximos locais.	51
Figura 18 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. A dimensionalidade do relevo é $N = 12$ e o parâmetro de epistasia é $K = 0$, ou seja, não há máximos locais.	51
Figura 19 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para relevos suaves ($K = 0$) com dimensionalidades $N = 9, 12$ e 15 como indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O número de agentes é $M = 10$	53
Figura 20 – Custo computacional médio dos quatro algoritmos de busca em função do número de agentes M para a probabilidade de mutação u indicada. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O relevo com $N = 12$ e $K = 6$ tem 114 máximos.	54
Figura 21 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O relevo com $N = 12$ e $K = 6$ tem 114 máximos.	56
Figura 22 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para relevos rugosos com parâmetros $(N = 9, K = 2)$, $(N = 12, K = 3)$ e $(N = 15, K = 4)$ como indicado e com número de máximos iguais a 11, 39 e 119, respectivamente. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O número de agentes é $M = 10$	57

LISTA DE TABELAS

Tabela 1 – Contribuições para <i>fitness</i> de cada uma das componentes de todas as possíveis configurações com $N = 3$ e $K = 1$	26
Tabela 2 – Valores ótimos do tamanho da população para os três algoritmos cooperativos para as probabilidade de mutação u utilizadas na figura 17. O aprendizado por imitação apresenta os menores custos para todos os valores de u	52
Tabela 3 – Valores ótimos da probabilidade de mutação para os três algoritmos cooperativos para os tamanhos da população M utilizados na figura 18. O aprendizado por imitação apresenta os menores custos para todos os valores de M	53
Tabela 4 – Valores ótimos do tamanho da população para os três algoritmos cooperativos para as probabilidade de mutação u utilizadas na figura 20.	55
Tabela 5 – Valores ótimos da probabilidade de mutação para os três algoritmos cooperativos para os tamanhos da população M utilizados na figura 21.	56

SUMÁRIO

1	INTRODUÇÃO	19
2	MODELO NK	23
2.1	Descrição Matemática	24
2.2	Interações Epistáticas	26
2.3	Algumas propriedades estatísticas do relevo de <i>fitness</i>	30
3	ALGORITMOS DE BUSCA NOS RELEVOS NK	33
3.1	Busca cega	35
3.2	Algoritmos evolucionários	38
3.2.1	Algoritmo genético assexuado (AGA)	39
3.2.2	Algoritmo genético sexuado (AGS)	41
3.3	Aprendizado por imitação (AI)	44
4	COMPARAÇÃO DO DESEMPENHO DOS ALGORITMOS	49
4.1	Relevo suave	49
4.2	Relevo rugoso	54
5	CONCLUSÃO	59
	REFERÊNCIAS	63

1 INTRODUÇÃO

Algoritmos de otimização inspirados na biologia¹ ou na física² são hoje ferramentas muito populares em diversas áreas do conhecimento devido a sua versatilidade e capacidade de resolução de problemas complexos utilizando pouca ou nenhuma informação a respeito das características desses problemas. Nessa classe de algoritmos, os algoritmos genéticos talvez sejam os mais conhecidos.³⁻⁵ Em analogia com o processo evolucionário, esses algoritmos exploram o espaço de soluções simulando uma população de agentes que evoluem de acordo com processos estocásticos que modelam a mutação, a seleção e a recombinação.

Nesse aspecto, os algoritmos genéticos popularizam nas décadas de 1980 e 1990 uma importante metodologia proposta inicialmente no estudo de dinâmicas sociais que é a modelagem dos indivíduos através de representações computacionais (agentes), conhecida hoje com modelos baseados em agentes. Esse tipo de artifício foi introduzido nos trabalhos pioneiros sobre autômatos celulares^{6,7} e ganhou força posteriormente na estruturação de trabalhos relacionados à inteligência artificial.⁸ Nessa metodologia, os agentes ou atores são descritos através de um determinado número de estados e evoluem sincronicamente em resposta aos estímulos que produzem uns nos outros.

Deveríamos esperar que houvesse mais algoritmos de otimização inspirados em sistemas sociais do que na evolução biológica. Afinal, sabemos que as soluções da maior parte dos problemas difíceis do mundo real estão além da capacidade de indivíduos isolados e requerem o esforço cooperativo de dezenas, senão milhares, de indivíduos como no programa Apollo⁹ e no projeto Manhattan.¹⁰ Entretanto, algoritmos baseados no aprendizado social (imitação, por exemplo) são muito mais recentes e bem menos populares que os algoritmos genéticos.¹¹⁻¹³ Seriam os algoritmos sociais (ou culturais) menos eficientes que os algoritmos genéticos? Essa é uma possibilidade a se considerar já que os processos de interação social podem levar a comportamentos (soluções) extremamente mal adaptativos,¹⁴ enquanto que os algoritmos evolucionários levam, quase que por construção, a soluções bem adaptadas às condições do meio em que os indivíduos vivem.

Nosso objetivo nessa dissertação é responder a essa questão através de um estudo comparativo do desempenho de duas versões dos algoritmos evolucionários³⁻⁵ – o algoritmo genético assexuado e o algoritmo genético sexuado – com um algoritmo inspirado no aprendizado social proposto recentemente que chamaremos aprendizado por imitação.¹³ A diferença entre os dois algoritmos genéticos considerados é que o algoritmo sexuado utiliza o processo de recombinação (o filho é uma mistura dos pais) além dos processos de mutação e seleção do algoritmo assexuado. Já a essência do aprendizado por imitação é a identificação do agente mais bem sucedido na população numa dada geração que servirá

como modelo para os outros agentes imitarem. Como base de comparação, vamos utilizar a busca cega, onde os agentes exploram o espaço de soluções de modo independente e sem utilizar praticamente nenhuma informação do problema a ser resolvido.

Para entendermos melhor nossos algoritmos, precisamos especificar o problema de otimização em que eles serão testados. Como o ideal é testar os algoritmos em problemas com diferentes graus de complexidade (ou seja, testá-los em problemas ‘simples’ e em problemas ‘difíceis’), nessa dissertação iremos considerar o problema de encontrar o único máximo global de relevos de *fitness* gerados pelo modelo NK.¹⁵ Esse modelo permite a geração de problemas de otimização fáceis onde o relevo tem apenas um máximo e problemas difíceis onde o relevo tem um grande número de máximos locais. A dificuldade do problema é determinada pelo parâmetro K do modelo NK que controla a rugosidade do relevo. Já o parâmetro N do modelo NK determina a dimensionalidade do relevo. O problema de otimização posto aos algoritmos é então a busca pelo único máximo global de um relevo NK. Por isso vamos nos referir aos nossos algoritmos como algoritmos de busca. A cada geração, os agentes assumem estados do modelo NK e portanto têm valores de *fitness* associados a eles. São esses valores que determinarão, através de um processo de escolha estocástico, se esses agentes passarão ou não seus descendentes para a próxima geração nos algoritmos evolucionários. No aprendizado por imitação, o valor de *fitness* determinará se o agente será um modelo ou um imitador.

Outro elemento necessário para entender nossos algoritmos de busca é a medida que usamos para comparar seus desempenhos na tarefa de encontrar o máximo global de um relevo NK. Definimos o custo computacional da busca como uma grandeza proporcional ao número total de atualizações realizadas nos estados dos agentes até que o máximo global seja encontrado. Como implementamos uma dinâmica síncrona, o número total de atualizações é igual ao produto do número de agentes pelo número de gerações transcorridas até que o máximo seja encontrado. Essa definição tem diversas implicações. Por exemplo, vamos ignorar a complexidade das atualizações usadas nos diferentes algoritmos que, naturalmente, refletem-se no tempo de execução de cada algoritmo. Essa escolha desfavorece bastante a busca cega, que tem o mecanismo de atualização mais simples. Outra implicação da nossa definição do custo computacional é que devemos executar os algoritmos até que encontrem o máximo global. Isso limita muito a dimensionalidade N dos relevos NK que vamos tratar, pois pelo menos na busca cega, o tempo para encontrar o máximo global cresce exponencialmente com a dimensionalidade do relevo. Finalmente, vamos supor que exista algum mecanismo externo (supervisor) que identifique quando o máximo global é encontrado e interrompa a busca nesse momento. Por isso dizemos que a busca cega, que não utiliza o valor de *fitness* no seu mecanismo de atualização, ‘praticamente’ não usa nenhuma informação sobre o relevo: de fato usa o conhecimento do máximo global para terminar a busca. Lembramos que o critério de desempenho usado comumente na literatura para comparar algoritmos é o melhor valor de *fitness* encontrado para um mesmo

tempo de execução.

Vamos nos referir aos algoritmos evolucionários e ao aprendizado por imitação como algoritmos cooperativos pois eles permitem a troca de informação entre os agentes, em particular, os valores de *fitness* de cada agente são acessíveis aos outros agentes. Isso é óbvio no caso do aprendizado por imitação, já que o agente modelo é imitado justamente por exibir um *fitness* mais alto que o dos outros agentes. Já nos algoritmos evolucionários esse ponto é mais sutil, e poderíamos muito bem denominá-los algoritmos competitivos ao invés de algoritmos cooperativos. Entretanto, como o processo de seleção utiliza o *fitness* relativo dos agentes fica claro que a probabilidade de reprodução de um agente particular depende dos valores do *fitness* de todos os outros agentes e, desse modo, podemos argumentar que esses valores são transmitidos entre os agentes. Finalmente, na busca cega os agentes realizam a busca de modo realmente independente e nenhuma informação é trocada entre eles.

A análise comparativa do desempenho desses algoritmos revelou alguns fatos interessantes e instrutivos que estão de acordo com nossa intuição sobre influência social e influência genética. Por exemplo, em situações em que a informação (*fitness*) exibida pelos agentes pode ser enganosa, como ocorre em relevos com muitos máximos locais, o algoritmo do aprendizado por imitação pode produzir resultados desastrosos se os seus parâmetros não forem muito bem ajustados. Já os algoritmos evolucionários, especialmente o algoritmo genético sexuado, nunca apresenta um desempenho muito ruim. Entretanto, com o ajuste cuidadoso dos parâmetros, o algoritmo cultural exibe desempenho igual ou superior ao dos algoritmos evolucionários. Um resultado surpreendente que vale a pena mencionar aqui é a vantagem relativamente pequena dos algoritmos cooperativos em relação à busca cega. De fato, para problemas com muitos máximos locais a busca cega é superior a esses algoritmos.

Iniciamos essa dissertação com um estudo bem detalhado dos relevos NK no Capítulo 2, onde apresentamos diversos resultados analíticos e numéricos que caracterizam esses relevos. Após a introdução do modelo NK, no Capítulo 3 apresentamos os quatro algoritmos de busca, a saber, a busca cega, o algoritmo genético assexuado, o algoritmo genético sexuado e o aprendizado por imitação. Além de discutir detalhes de implementação, analisamos o custo computacional desses algoritmos em relevos com vários níveis de complexidade. Já o Capítulo 4 trata da comparação entre esses algoritmos. Finalmente, no Capítulo 5 apresentamos nossas conclusões e comentários finais.

2 MODELO NK

Com os avanços no século passado na descrição de processos genéticos e evolutivos, algoritmos de busca não-determinísticos baseados nos mecanismos de seleção natural ganharam aplicações em diversas áreas. Nesse contexto, o geneticista Sewall Wright introduziu à teoria evolucionária o conceito de relevo adaptativo em 1932, que corresponde a um espaço de genótipos aos quais atribui-se uma medida de performance (*fitness*) e onde os genótipos são ditos vizinhos quando diferem entre si por uma única mutação.¹⁶ No entanto, até meados da década de 80 ainda não havia uma modelagem ou caracterização matemática para os espaços correspondentes às possíveis combinações genéticas nos quais os processos de busca e otimização se desenvolvem.

De fato, em 1987 Stuart Kauffman trilha os primeiros passos para produzir uma teoria geral das caminhadas adaptativas em relevos rugosos,¹⁵ introduzindo o *fitness* como uma propriedade que depende não apenas dos genes separadamente, mas também de suas interações com os outros genes. Em outras palavras, a contribuição do *fitness* de um dado gene depende não só de suas próprias características, mas também das dos outros genes com que está associado. O foco daquele estudo pioneiro, e de muitos outros que se seguiram, é a caracterização estatística do relevo de *fitness*. O modelo NK tomou a sua forma atual apenas em 1989¹⁷ com a introdução dos parâmetros N e K , onde N , em analogia com sistemas genéticos, indica o número de genes que caracterizam o genótipos e $K = 0, 1, \dots, N - 1$ indica o número de genes que influenciam a contribuição para a performance global associada a cada um dos genes. Em outras palavras, K é um indicativo da complexidade das interações epistáticas entre os genes e, por essa razão, chamamos K de parâmetro de epistasia. Dessa forma, aumentando o valor de K para N fixo passamos de um relevo simples, com poucos máximos para um relevo rugoso, com muitos máximos locais.

O modelo NK encontra aplicações em diversas áreas como, por exemplo, imunologia teórica,¹⁷ genética de populações,^{18–20} modelagem de proteínas,^{21–23} administração de empresas e organização do trabalho^{24–27} e, naturalmente, teste para algoritmos de otimização.²⁸ Essa popularidade se deve ao modelo NK permitir a construção de uma medida de performance para sistemas nos quais cada componente pode estar associada a vários estados e onde há algum tipo de interação entre as componentes, permitindo o controle do tamanho do sistema e do nível de complexidade das interações através dos parâmetros N e K .

Nesse capítulo iremos apresentar de forma sucinta a formulação matemática do modelo NK e fazer uma breve caracterização dos relevos de *fitness* gerados pelo modelo analisando propriedades como número de máximos, *fitness* médio e distância de Hamming

entre os máximos.

Na segunda metade do século passado a ideia de um espaço de proteína começou a se popularizar.^{15,17} Nesse contexto, o modelo NK torna possível a descrição de peptídeos através da associação do parâmetro N à quantidade de aminoácidos que compõe o peptídeo. Como há diferentes tipos de aminoácidos compondo a proteína, considera-se que há p possíveis estados representando os diferentes tipos de aminoácidos. Além disso, os peptídeos são considerados vizinhos se eles diferem entre si por apenas um aminoácido. De forma que há $(p - 1) \times N$ vizinhos para cada um dos peptídeos.

2.1 Descrição Matemática

Considere um sistema que pode assumir configurações representadas por um vetor $\vec{x}$ com N componentes $\vec{x} = (x_0, \dots, x_{N-1})$, onde cada componente x_i pode assumir um número discreto de estados, $x_i \in \{0, 1, \dots, p-1\}$. Nesse caso temos p^N configurações no nosso espaço de vetores. Por exemplo, se $\vec{x}$ representa uma molécula de DNA ou RNA temos $p = 4$, já que os estados das componentes seriam associados às quatro diferentes bases nitrogenadas. Se $\vec{x}$ representa uma proteína, então $p = 20$ e os estados das componentes correspondem aos 20 aminoácidos. Nessa dissertação consideramos apenas o caso mais simples, a saber, o caso de sequências binárias ($p = 2$), de forma que $x_i \in \{0, 1\}$ corresponde a um bit e a dimensão do espaço de sequências binárias é 2^N .

O parâmetro de epistasia K do modelo NK controla o número de componentes que influenciam uma dada componente de modo que K deve assumir os valores inteiros $K \in \{0, 1, \dots, N-1\}$. Há várias maneiras possíveis de escolher as K componentes influenciadoras e essa escolha pode resultar em relevos com características muito distintas.^{29,30} Nessa dissertação vamos considerar que as K componentes que influenciam a componente i são seus vizinhos adjacentes à direita, ou seja, $i + 1, \dots, i + K$ onde a soma é feita módulo N (Lembramos que $N + l = l$ numa soma módulo N). Isso é equivalente a utilizar condições cíclicas de contorno. A figura 1 ilustra essa escolha de influenciadores para uma configuração arbitrária com diferentes valores de K para $N = 6$.

Podemos entender o espaço de configurações (isto é, o espaço de sequências binárias) como sendo uma estrutura na qual cada vértice está associado a uma das 2^N possíveis configurações e os vértices adjacentes correspondem às configurações vizinhas (isto é, que diferem entre si por exatamente um bit). Trata-se, portanto, de um espaço com N dimensões. O modelo NK associa cada um desses vértices a um número real, através da função *fitness*, $\Phi : \{0, 1\}^N \rightarrow \mathbb{R}$, dada por

$$\Phi(\vec{x}) = \frac{1}{N} \sum_{i=0}^{N-1} \phi_i(x_i, \dots, x_{i+K}), \quad (2.1)$$

onde $\phi_i(x_i, \dots, x_{i+K})$ corresponde à contribuição da componente i de $\vec{x}$ para o *fitness*. Vemos então que essa contribuição é uma função dos estados da componente i e dos estados das

Figura 1 – Esquema mostrando a vizinhança adjacente à direita de cada uma das componentes de sequências binárias de tamanho $N = 6$ para valores distintos de K . Nota-se que no caso $K = N - 1$ cada componente sofre influência de todas as outras componentes.

Fonte: Elaborada pela autora.

K componentes vizinhas à direita de i . Novamente, lembramos que a soma que aparece nos índices das componentes é definida módulo N .

Para gerar o relevo de *fitness* precisamos identificar todas as 2^N configurações possíveis, calcular todas as contribuições para o *fitness* distintas e então determinar o *fitness* associado a cada configuração. Como há N componentes $i = 0, \dots, N - 1$ e 2^{K+1} possíveis argumentos para ϕ_i , temos um total de $N \times 2^{K+1}$ contribuições do *fitness* distintas. A tabela 1 mostra todas as possíveis configurações, contribuições e *fitness* para $N = 3$ e $K = 1$.

Em geral, os valores das contribuições ϕ_i são desconhecidos ou difíceis de determinar, devido ao seu vasto número. Por isso, comumente adota-se uma abordagem probabilística para determinar as contribuições do *fitness*, atribuindo-lhes valores aleatórios de acordo com alguma distribuição de probabilidade. De fato, uma das ideias centrais do modelo NK é que os efeitos das interações epistáticas são tão complexos que é possível fazer uma caracterização estatística das contribuições do *fitness* atribuindo à elas valores aleatórios de acordo com uma distribuição específica.¹⁷

Nessa dissertação vamos sempre usar a distribuição uniforme em $(0, 1)$ para gerar os 2^{K+1} valores de $\phi_i(x_i, \dots, x_{i+K})$ para cada $i = 0, \dots, N - 1$. Essa abordagem torna o modelo NK um modelo de interações epistáticas aleatórias e garante que sequências

Tabela 1 – Contribuições para *fitness* de cada uma das componentes de todas as possíveis configurações com $N = 3$ e $K = 1$.

configuração ($\vec{x}$)	componentes do <i>fitness</i>	<i>fitness</i> da configuração ($\Phi(\vec{x})$)
(0, 0, 0)	$\phi_0(0, 0), \phi_1(0, 0), \phi_2(0, 0)$	$\frac{1}{3} (\phi_0(0, 0) + \phi_1(0, 0) + \phi_2(0, 0))$
(1, 0, 0)	$\phi_0(1, 0), \phi_1(0, 0), \phi_2(0, 1)$	$\frac{1}{3} (\phi_0(1, 0) + \phi_1(0, 0) + \phi_2(0, 1))$
(0, 1, 0)	$\phi_0(0, 1), \phi_1(1, 0), \phi_2(0, 0)$	$\frac{1}{3} (\phi_0(0, 1) + \phi_1(1, 0) + \phi_2(0, 0))$
(0, 0, 1)	$\phi_0(0, 0), \phi_1(0, 1), \phi_2(1, 0)$	$\frac{1}{3} (\phi_0(0, 0) + \phi_1(0, 1) + \phi_2(1, 0))$
(0, 1, 1)	$\phi_0(0, 1), \phi_1(1, 1), \phi_2(1, 0)$	$\frac{1}{3} (\phi_0(0, 1) + \phi_1(1, 1) + \phi_2(1, 0))$
(1, 0, 1)	$\phi_0(1, 0), \phi_1(0, 1), \phi_2(1, 1)$	$\frac{1}{3} (\phi_0(1, 0) + \phi_1(0, 1) + \phi_2(1, 1))$
(1, 1, 0)	$\phi_0(1, 1), \phi_1(1, 0), \phi_2(0, 1)$	$\frac{1}{3} (\phi_0(1, 1) + \phi_1(1, 0) + \phi_2(0, 1))$
(1, 1, 1)	$\phi_0(1, 1), \phi_1(1, 1), \phi_2(1, 1)$	$\frac{1}{3} (\phi_0(1, 1) + \phi_1(1, 1) + \phi_2(1, 1))$

Fonte: Elaborada pela autora.

binárias diferentes possuam necessariamente valores de *fitness* distintos. A consequência disso é que o máximo global de um relevo NK é único.

2.2 Interações Epistáticas

Um dos maiores sucessos do modelo NK é possibilitar o estudo dos efeitos de interações epistáticas complexas em sistemas com várias componentes.^{15,17} Essas componentes interagem entre si influenciando o *fitness* de suas componentes vizinhas de modo que quanto mais interações há entre elas, mais complexo se torna o relevo. A modelagem do grau de interação epistáticas entre as componentes do sistema é feita através do parâmetro K . Modificações nesse parâmetro causam grandes mudanças com relação à rugosidade, distribuição dos máximos locais do relevo e dos valores de *fitness* dos máximos. À medida que aumentamos o valor de K passamos de um relevo suave com um único máximo ($K = 0$) para relevos rugosos, com vários máximos locais e, como veremos mais à frente, chegando ao relevo extremo em que não há correlação entre o *fitness* de estados vizinhos ($K = N - 1$).

Podemos entender o efeito do parâmetro de epistasia K na rugosidade dos relevos NK através da identificação e contagem dos máximos. Uma configuração $\vec{x}_m$ é dita um máximo local se o seu *fitness* for maior que o de todas as configurações na sua vizinhança Ω_m , ou seja,

$$\Phi(\vec{x}_m) > \Phi(\vec{x}_i), \quad \forall \vec{x}_i \in \Omega_m. \quad (2.2)$$

Lembramos que a vizinhança Ω_m de $\vec{x}_m$ são as N sequências binárias que diferem de $\vec{x}_m$ por exatamente um bit e que não é necessário cuidado com o sinal de desigualdade, pois duas sequências distintas sempre terão *fitness* distintos. Já o máximo global corresponde ao (único) estado $\vec{x}_{max}$ cujo *fitness* é maior que o de todas as outras configurações possíveis.

Figura 2 – Número médio de máximos em função do parâmetro de epistasia K para diferentes valores de N como indicado. Cada símbolo representa a média sobre 3000 relevos de *fitness* distintos. As barras de erro são menores que o tamanho dos símbolos.

Fonte: Elaborada pela autora.

A figura 2 ilustra como o número médio de máximos varia com o parâmetro K para alguns valores fixos de N . Para um relevo de *fitness* fixo, geramos as 2^N configurações e verificamos quantas delas são máximos. Repetimos esse procedimento para 3000 relevos distintos para cada um dos valores de N e K e apresentamos na figura o resultado da média do número de máximos. Os resultados indicam que $\langle \mathcal{N}_m \rangle$ cresce exponencialmente com K para K grande,²⁹ de acordo com nossa expectativa de que o relevo torna-se mais rugoso (e complexo) com o aumento do alcance das interações epistáticas.

Como vimos, o aumento de K afeta a distribuição dos máximos locais no relevo e consequentemente a caminhada em direção ao máximo global em determinados processos de busca. Para entendermos melhor esse efeito, analisamos a dependência do *fitness* das configurações em função de suas distâncias de Hamming ao máximo global $\vec{x}_{max}$. Lembramos que a distância de Hamming $D_H(\vec{x}_a, \vec{x}_b)$ entre as configurações $\vec{x}_a$ e $\vec{x}_b$ é dada pelo número de componentes distintas entre elas. Como mostra a figura 3, para $K = 0$, o *fitness* médio diminui linearmente com a distância ao máximo global, indicando que a presença desse máximo influencia todo o relevo. Porém, conforme K aumenta essa influência fica limitada às configurações muito próximas ao máximo global, já que o *fitness* médio das configurações mais afastadas independe da distância de Hamming $D_H(\vec{x}, \vec{x}_{max})$. Lembremos que para $K = N - 1$ essa influência é nula, mesmo para configurações vizinhas a $\vec{x}_{max}$. Outro resultado interessante da figura 3 é que o *fitness* médio do máximo global (obtido para $D_H(\vec{x}, \vec{x}_{max}) = 0$) aumenta com K .

Figura 3 – *Fitness* média das configurações $\vec{x}$ que diferem de $D_H(\vec{x}, \vec{x}_{max})$ componentes do máximo global $\vec{x}_{max}$ para $N = 15$. Cada símbolo representa a média sobre todas as configurações distintas que têm a mesma distância de Hamming ao máximo global e sobre 3000 relevos de *fitness* distintos. As barras de erro são menores que o tamanho dos símbolos.

Fonte: Elaborada pela autora.

Outra informação interessante é a distância média entre os máximos locais e o máximo global, mostrada na figura 4. Para K pequeno, temos a situação em que os (poucos) máximos locais estão próximos ao máximo global e, conseqüentemente, devem ter também um *fitness* alto. Para K grande, os (muitos) máximos locais estão praticamente uniformemente distribuídos pelo espaço de configurações e têm, em média, metade das suas componentes diferentes das do máximo global ($D_H \approx 1/2$).

Em princípio, os valores de *fitness* dos máximos locais e do máximo global deveriam ser influenciados tanto pelo número de componentes do sistema N , quanto pelo parâmetro de epistasia K . A figura 5 mostra que a situação é mais interessante. Para K pequeno, o *fitness* médio dos máximos locais não é afetado pelo parâmetro N e mantém-se sempre acima do valor médio de *fitness* do relevo ($\langle \Phi(\vec{x}) \rangle = 0.5$, como veremos na seção seguinte). O mesmo acontece com o *fitness* médio do máximo global. No entanto, para valores altos de K ocorre a chamada catástrofe da complexidade,¹⁵ na qual o valor médio do *fitness* dos máximos locais tende ao valor médio de *fitness* do relevo, tornando difícil distingui-los das demais configurações, mas mesmo nesse cenário o *fitness* médio dos máximos globais mantém-se acentuadamente acima de $\langle \Phi(\vec{x}) \rangle = 0.5$ e aumenta com o aumento de N . Note que usamos a expressão *fitness* médio dos máximos globais embora cada relevo tenha um único máximo global pois os resultados exibidos na figura 5 representam uma média sobre diferentes relevos de *fitness*.

Figura 4 – Distância Hamming média entre os máximos locais e o máximo global em função do parâmetro de epistasia K para os valores de N indicados. Cada símbolo representa o resultado sobre 1000 relevos de *fitness* distintos da média da distância de Hamming de todos os máximos locais ao máximo global. As barras de erro são menores que o tamanho dos símbolos.

Fonte: Elaborada pela autora.

Figura 5 – *Fitness* médio dos máximos locais e dos máximos globais em função de N para para os parâmetros de epistasia $K=2$ e $K=N-1$, como indicado. Cada símbolo representa o resultado da média sobre 1000 relevos de *fitness* distintos. As barras de erro são menores que o tamanho dos símbolos.

Fonte: Elaborada pela autora.

2.3 Algumas propriedades estatísticas do relevo de *fitness*

Podemos facilmente calcular analiticamente algumas grandezas estatísticas que caracterizam os relevos de *fitness* gerados pelo modelo NK. A primeira delas é o valor médio do *fitness* de uma sequência $\vec{x}$,

$$\langle \Phi(\vec{x}) \rangle = \frac{1}{N} \sum_{i=0}^{N-1} \langle \phi_i(x_i, \dots, x_{i+K}) \rangle, \quad (2.3)$$

onde usamos que a média da soma é a soma das médias. Aqui a notação $\langle \dots \rangle$ refere-se à média sobre relevos diferentes, ou seja, sobre as diferentes realizações das contribuições das componentes de *fitness* $\phi_i(x_i, \dots, x_{i+K})$. Lembrando que essas componentes são variáveis aleatórias independentes e identicamente distribuídas com distribuição uniforme em $(0, 1)$, temos $\langle \Phi(\vec{x}) \rangle = 1/2$, independente de N e K . Calculemos agora o segundo momento $\Phi(\vec{x})$, que é necessário para a determinação da variância. Temos

$$\langle \Phi^2(\vec{x}) \rangle = \frac{1}{N^2} \left\langle \left(\sum_i \phi_i \right)^2 \right\rangle = \frac{1}{N^2} \sum_i \langle \phi_i^2 \rangle + \frac{1}{N^2} \sum_i \sum_{j \neq i} \langle \phi_i \phi_j \rangle, \quad (2.4)$$

onde omitimos o argumento de ϕ_i por simplicidade de notação. Lembrando que $\langle \phi_i^2 \rangle = 1/3$ e que $\langle \phi_i \phi_j \rangle = \langle \phi_i \rangle \langle \phi_j \rangle = 1/4$ se $i \neq j$, obtemos $\langle \Phi^2(\vec{x}) \rangle = 1/4 + 1/12N$. Portanto,

$$\text{var}(\Phi(\vec{x})) = \langle \Phi^2(\vec{x}) \rangle - \langle \Phi(\vec{x}) \rangle^2 = \frac{1}{12N}, \quad (2.5)$$

que também não depende do parâmetro de epistasia K . Escrevendo $\sigma^2 = \langle \phi_i^2 \rangle - \langle \phi_i \rangle^2 = 1/12$ e usando a desigualdade de Chebyshev³¹ temos

$$P(|\Phi(\vec{x}) - \langle \Phi(\vec{x}) \rangle| < \epsilon) = P\left(|\Phi(\vec{x}) - \frac{1}{2}| < \epsilon\right) \geq 1 - \frac{\sigma^2}{N\epsilon^2}, \quad (2.6)$$

para qualquer $\epsilon > 0$. Portanto, para N muito grande os relevos de *fitness* do modelo NK são praticamente planos, ou seja, independentemente da configuração $\vec{x}$ temos $\Phi(\vec{x}) \approx 1/2$.

O efeito do parâmetro de epistasia K aparece somente quando calculamos correlações entre configurações diferentes. De particular importância são configurações vizinhas, já que a definição de máximo local depende apenas delas. Sem perda de generalidade, vamos calcular agora a correlação entre as duas configuração vizinhas $\vec{x}_a = (0, 0, \dots, 0)$ e $\vec{x}_b = (1, 0, \dots, 0)$. Os resultados anteriores mostram que $\langle \Phi(\vec{x}_a) \rangle = \langle \Phi(\vec{x}_b) \rangle = 1/2$ e $\text{var}(\Phi(\vec{x}_a)) = \text{var}(\Phi(\vec{x}_b)) = 1/12N$. A covariância de $\Phi(\vec{x}_a)$ e $\Phi(\vec{x}_b)$ é definida como

$$\text{cov}(\Phi(\vec{x}_a), \Phi(\vec{x}_b)) = \langle \Phi(\vec{x}_a) \Phi(\vec{x}_b) \rangle - \langle \Phi(\vec{x}_a) \rangle \langle \Phi(\vec{x}_b) \rangle. \quad (2.7)$$

Observando que, independentemente de seus argumentos, as contribuições do *fitness* ϕ_i e ϕ_j das componentes i e j são independentes para $i \neq j$ temos

$$\text{cov}(\Phi(\vec{x}_a), \Phi(\vec{x}_b)) = \frac{1}{N^2} \sum_{i=0}^{N-1} \text{cov}(\phi_i^a, \phi_i^b). \quad (2.8)$$

Aqui $\phi_i^a = \phi_i(0, \dots, 0) \forall i$ e o argumento da função tem $K + 1$ zeros. Por outro lado, $\phi_0^b = \phi_0(1, 0, \dots), \phi_1^b = \phi_1(0, 0, \dots), \dots, \phi_{N-1}^b = \phi_{N-1}(0, 1, \dots)$. Observando que $\text{cov}(\phi_i^a, \phi_i^b) = 0$ para as componentes i em que os argumentos de ϕ_i^a e ϕ_i^b forem diferentes e $\text{cov}(\phi_i^a, \phi_i^b) = \text{var}(\phi_i^a) = 1/12$ para as componentes i em que os argumentos de ϕ_i^a e ϕ_i^b forem iguais, precisamos apenas estimar o número de componentes com argumentos diferentes para efetuarmos a soma na equação (2.8). Faremos isso por indução. Para $K = 0$ apenas a componente $i = 0$ tem argumentos diferentes nas configurações $\vec{x}_a$ e $\vec{x}_b$. Para $K = 1$, são as componentes $i = 0$ e $i = N - 1$. Para $K = 2$ são as componentes $i = 0, i = N - 2$ e $i = N - 1$, e assim por diante. Assim, número de componentes diferentes é $K + 1$. Com isso, podemos escrever

$$\text{cov}(\Phi(\vec{x}_a), \Phi(\vec{x}_b)) = \frac{1}{12N^2} [N - (K + 1)]. \quad (2.9)$$

Finalmente, podemos calcular a correlação

$$\text{corr}(\Phi(\vec{x}_a), \Phi(\vec{x}_b)) = \frac{\text{cov}(\Phi(\vec{x}_a), \Phi(\vec{x}_b))}{[\text{var}(\Phi(\vec{x}_a))\text{var}(\Phi(\vec{x}_b))]^{1/2}} = 1 - \frac{K + 1}{N}. \quad (2.10)$$

Esse resultado mostra que a correlação é máxima para $K = 0$ e diminui linearmente com K até $K = N - 1$, onde a correlação entre os estados vizinhos se anula. Para o caso $K = N - 1$ cada componente da configuração tem seu *fitness* influenciado pelo estado de todas as outras componentes (veja a tabela 1), de forma que quando mudamos um bit em uma das componentes alteramos o valor de todas as contribuições para o *fitness* da configuração e consequentemente gerando um valor de *fitness* sem correlação com o valor anterior a mudança do bit. Esse caso extremo é conhecido como relevo casa de cartas²¹ e tem grande semelhança com o clássico modelo das energias aleatórias (REM) da física estatística.³² A diferença é que no modelo REM as energias (o negativo da função *fitness*) são distribuídas por uma distribuição normal, enquanto que no relevo casa de cartas o *fitness* está limitado ao intervalo $(0, 1)$, sendo fortemente concentrado no centro desse intervalo para N grande de acordo com a equação (2.6).

Para o relevo casa de cartas ($K = N - 1$) em que o *fitness* das configurações são variáveis aleatórias independentes e identicamente distribuídas, podemos calcular o número médio de máximos locais analiticamente. Considere uma determinada configuração $\vec{x}_m$ cujo *fitness* é $\Phi(\vec{x}_m) = \Phi_m$. A probabilidade de que $\vec{x}_m$ seja um máximo local é a probabilidade de que seu *fitness* seja maior que o *fitness* de seus N vizinhos, ou seja,

$$\left[\int_0^{\Phi_m} P(\Phi) d\Phi \right]^N, \quad (2.11)$$

onde $P(\Phi)$ satisfaz $\int_0^1 d\Phi P(\Phi) = 1$. Observe que $P(\Phi)$ não é uniforme no intervalo $(0, 1)$, já que é concentrada em $\Phi = 1/2$ para N grande. Agora, se quisermos a probabilidade de que uma dada configuração seja máximo local independentemente do valor de Φ_m devemos

integrar a expressão acima sobre os possíveis valores dessa grandeza, ou seja,

$$\mathcal{P}_m = \int_0^1 P(\Phi_m) \left[\int_0^{\Phi_m} P(\Phi) d\Phi \right]^N d\Phi_m. \quad (2.12)$$

Felizmente, essa integral é muito simples de ser calculada qualquer que seja a distribuição $P(\Phi)$. De fato, se fizermos $z = \int_0^{\Phi_m} P(\Phi) d\Phi$ de modo que $z = 0$ para $\Phi_m = 0$ e $z = 1$ para $\Phi_m = 1$ teremos

$$\mathcal{P}_m = \int_0^1 z^N dz = \frac{1}{N+1}. \quad (2.13)$$

Podemos então estimar o número médio de máximos locais como o produto da probabilidade de uma determinada configuração ser máximo local ($\mathcal{P}_m$) pelo número total de configurações (2^N), ou seja,

$$\langle \mathcal{N}_m \rangle = \frac{2^N}{N+1}. \quad (2.14)$$

É interessante que mesmo no caso de um relevo extremamente rugoso como o obtido fazendo-se $K = N - 1$ no modelo NK, a fração de máximos locais vai a zero à medida que a dimensão N do espaço de configurações aumenta.

Outro resultado interessante relativo ao relevo casa de cartas ($K = N - 1$) é a estimativa do valor médio do máximo global. De fato, as equações anteriores permitem escrever

$$\langle \Phi_m \rangle = (N+1) \int_0^1 d\phi_m \phi_m \left[\int_0^{\Phi_m} P(\Phi) d\Phi \right]^N, \quad (2.15)$$

cujo cálculo requer o conhecimento de $P(\Phi)$. Podemos ver como esse resultado reproduz qualitativamente o comportamento de $\langle \Phi_m \rangle$ ilustrado na figura 5 supondo, erroneamente, que $P(\Phi)$ seja uniforme no intervalo $(0, 1)$. Com essa hipótese temos $\langle \Phi_m \rangle = (N+1)/(N+2)$ que é uma função crescente de N .

3 ALGORITMOS DE BUSCA NOS RELEVOS NK

Nesse capítulo, estudaremos o desempenho de quatro algoritmos de busca estocásticos que exploram relevos NK (isto é, relevos de *fitness* gerados pelo modelo NK) em busca do único máximo global. São eles, a busca cega, duas versões dos algoritmos evolucionários³⁻⁵ – o algoritmo genético assexuado (AGA) e o algoritmo genético sexuado (AGS) – e um algoritmo inspirado na evolução cultural³³ proposto recentemente¹³ e que denominaremos aprendizado por imitação (AI). Como já mencionado, a vantagem de se utilizar o modelo NK nesse tipo de estudo é a possibilidade de se controlar a dificuldade dessa busca através da variação do número de máximos locais. Todos os quatro algoritmos considerados utilizam grupos ou populações de agentes na busca pelo máximo global. Os algoritmos evolucionários e o cultural são ditos algoritmos cooperativos pois permitem a troca de informação entre os agentes. Já na busca cega os agentes exploram o relevo de *fitness* de modo independente e sem utilizar nenhuma informação local desse relevo. Naturalmente, o desempenho da busca cega será a referência para avaliarmos a eficiência dos algoritmos cooperativos. Em princípio, esperamos que esses algoritmos apresentem desempenho muito superior ao da busca cega.

Independentemente do algoritmo considerado, a busca é realizada por um grupo de M agentes, cada um representado por uma sequência binária de tamanho N , de modo que podemos imediatamente associar cada agente com uma configuração do modelo NK. Inicialmente, os agentes são escolhidos de modo aleatório. Esse grupo inicial evolui ao longo das gerações de acordo com as regras dos algoritmos de busca, até que ao menos um dos agentes encontre o máximo global, ou seja, sua configuração seja idêntica à do máximo global. O número de gerações necessárias para que os agentes encontrem a solução corresponde ao tempo de busca.

Nesta dissertação usaremos quase que exclusivamente sequências de tamanho $N = 12$ de modo que o número de configurações do modelo NK ($2^N = 4096$) não é proibitivamente grande a ponto de impedir as muitas repetições das buscas necessárias para obtermos uma estatística confiável. Ainda, com essa escolha de N podemos estudar os casos em que o número de agentes M é da mesma ordem ou mesmo maior que o número de configurações, já que tipicamente o tamanho do espaço de busca é muitas ordens de grandeza maior que o número de agentes utilizados nos algoritmos cooperativos ou na busca cega. Somente no Capítulo 4 iremos considerar brevemente sequências de tamanhos $N = 9$ e $N = 15$.

Devido a formulação do algoritmo cultural AI¹³ na qual os processos de mutação e cópia (imitação) ocorrem de forma independente, vamos utilizar uma formulação análoga para os algoritmos evolucionários fazendo com que os processos de mutação e seleção sejam

Figura 6 – Esquema da implementação dos algoritmos de busca para a formulação em que a mutação é independente dos processos que definem os algoritmos cooperativos. A busca cega ocorre para $u = 1$.

Fonte: Elaborada pela autora.

também independentes. Esse esquema corresponde ao modelo clássico de Crow-Kimura da genética de populações,³⁴ que é muito popular entre a comunidade de física estatística pois em alguns casos especiais a dinâmica evolucionária pode ser mapeada em um Hamiltoniano de spins quânticos.^{35–37} Lembramos que em sua formulação usual,⁵ a mutação é vista como um erro no processo de replicação ou reprodução e, portanto, os dois processos estão acoplados e ocorrem de forma sequencial. Já no esquema em que mutação e seleção (ou imitação) são processos independentes, primeiro decidimos qual processo deve ocorrer – mutação com probabilidade u ou seleção com probabilidade $1 - u$, onde $u \in [0, 1]$ é a probabilidade de mutação – e então executamos o processo selecionado. Adiantamos que a mutação consiste em simplesmente inverter um bit escolhido aleatoriamente da sequência binária do agente que passa pelo processo de mutação.

A figura 6 ilustra a implementação dos algoritmos de busca no relevo NK utilizando a formulação em que o processo de mutação é independente dos processos de seleção (AGA e AGS), de recombinação (AGS) e de imitação (AI).

Finalmente, vamos fazer a hipótese usual de que não ocorra superposição de gerações de modo que apenas atualizamos os estados dos agentes após havermos determinado os estados de todos eles usando a informação da geração anterior. Essa é a atualização síncrona ou paralela que faz com que todos os M agentes de uma geração sejam simultaneamente substituídos pelos M agentes da próxima geração.

Além de uniformizar o procedimento de busca pelo máximo global para os diferentes algoritmos, precisamos estabelecer uma medida quantitativa que nos permita comparar seus

desempenhos para relevos de diferentes dimensionalidades N e populações de tamanhos distintos M . Com esse objetivo, vamos definir o custo computacional C de uma busca como uma grandeza proporcional ao número total de atualizações efetuadas até que o máximo global seja encontrado e, portanto, a busca seja encerrada. Supondo que isso ocorra na geração t^* e lembrando que a cada geração são realizadas exatamente M atualizações então $C \propto Mt^*$. Note que nessa medida estamos ignorando a complexidade das atualizações que podem corresponder a tempos reais de execução muito distintos. Como esperamos que t^* cresça linearmente como o número de configurações no relevo NK, definimos o custo computacional de uma busca como

$$C = \frac{Mt^*}{2^N}. \quad (3.1)$$

Como veremos a seguir, todos os quatro algoritmos considerados são estocásticos de modo que devemos executar um número grande de buscas independentes (tipicamente 5×10^4 buscas) e calcular o custo computacional médio $\langle C \rangle$. A notação $\langle \dots \rangle$ representa a média sobre buscas independentes. Neste capítulo e em boa parte do capítulo seguinte vamos considerar apenas três relevos NK, com parâmetros $(N = 12, K = 0)$, $(N = 12, K = 3)$ e $(N = 12, K = 6)$. Todas as buscas ocorrem nos mesmos relevos. Em particular, a realização do relevo suave $(N = 12, K = 0)$ tem apenas um máximo que, portanto, é o máximo global, a realização de relevo $(N = 12, K = 3)$ tem 40 máximos e a realização de relevo $(N = 12, K = 6)$ tem 114 máximos.

Nas seções seguintes vamos descrever os detalhes de implementação dos nossos algoritmos de busca e estudar seus desempenhos quando o tamanho da população M e a probabilidade de mutação u variam. Em todas as figuras apresentadas neste e no próximo capítulo, as barras de erro são menores que o tamanho dos símbolos.

3.1 Busca cega

O algoritmo não-determinístico mais simples (e geral) que podemos utilizar para explorar o espaço de configurações de um problema é a busca cega. Como mencionado, nesse algoritmo de busca temos M agentes cada um representado por uma sequência binária equivalente à alguma das possíveis configurações do relevo NK. A cada geração, os M agentes executam sincronicamente o movimento elementar, que consiste simplesmente em sortear com probabilidade uniforme uma das N componentes da sequência binária e inverter o seu estado (bit), ou seja, fazer $1 \rightarrow 0$ ou $0 \rightarrow 1$. Naturalmente, o movimento elementar nada mais é do que o processo de mutação. Esse procedimento, que está esquematizado na figura 7, é repetido até que algum dos agentes encontre o máximo global. Na busca cega, um agente não exerce influência alguma sobre os demais (não há troca de informação entre eles) e o nível de complexidade do relevo em que a busca ocorre não influencia no tempo de busca t^* , já que o *fitness* do agente não influencia na escolha de seu próximo estado.

Figura 7 – Esquema do movimento elementar ou mutação em uma sequência de $N = 5$ componentes utilizado na busca cega pelos máximos globais de relevos NK.

Fonte: Elaborada pela autora.

Note que na formulação de mutação e seleção independentes, os algoritmos cooperativos reduzem-se a busca cega para $u = 1$.

O tempo médio necessário para encontrar o máximo global $\langle t^* \rangle$ na busca cega pode ser calculado analiticamente²⁸ e depende apenas da dimensão N do relevo e do número M de agentes realizando a busca. A título de completeza, a seguir vamos reproduzir esse cálculo.

Começamos supondo, sem perda de generalidade, que a sequência $\vec{x}_m = (1, \dots, 1)$ seja o único máximo global do relevo. Lembremos que o tempo de busca da busca cega não depende de detalhes do relevo, desde que se garanta que exista apenas um máximo global. Consideremos primeiramente o caso de um único agente, isto é, $M = 1$. Suponhamos que na geração t , a sequência associada ao agente tenha j componentes 1 e $N - j$ componentes 0. Após o movimento elementar (mutação de um bit), essa sequência terá $j + 1$ ou $j - 1$ componentes 1 com probabilidades $(j - N)/N$ e j/N , respectivamente. Dessa forma, a probabilidade da sequência de nosso agente exibir exatamente i componentes 1 na geração $t + 1$ é

$$T_{ij} = (1 - j/N) \delta_{i,j+1} + (j/N) \delta_{i,j-1} \quad (3.2)$$

para $j < N$, onde $\delta_{i,j}$ é a delta de Kronecker. No caso $j = N$ todas as componentes da sequência são 1 e, portanto, o agente encontrou o máximo global e o processo de busca deve ser interrompido. Implementamos essa condição impondo que o máximo global seja um estado absorvente, ou seja, $T_{iN} = \delta_{i,N}$.

Podemos verificar que a matriz $\mathbb{T}$ cujos elementos são definidos acima atende a condição de normalização

$$\sum_{i=0}^N T_{ij} = \sum_{i=0}^N \left[\left(1 - \frac{j}{N}\right) \delta_{i,j+1} + \frac{j}{N} \delta_{i,j-1} \right] = \left(1 - \frac{j}{N}\right) + \frac{j}{N} = 1 \quad (3.3)$$

para todo j e, como $0 \leq T_{ij} \leq 1$, concluímos que $\mathbb{T}$ é uma matriz estocástica tridiagonal³⁸ que descreve um processo estocástico com o estado absorvente $i = N$ que corresponde ao máximo global. A probabilidade $\pi_N(t)$ de que o agente tenha N componentes 1 na geração t é

$$\pi_N(t) = \sum_{i=0}^N T_{Ni} \pi_i(t-1) \quad (3.4)$$

onde $\pi_i(t)$ é a probabilidade de que a sequência do agente tenha i componentes 1 na geração t . A expansão de $\pi_i(t)$ e das componentes T_{ij} em termos dos autovetores à direita e à esquerda da matriz $\mathbb{T}$ permite-nos escrever $\pi_N(t) \approx 1 - \lambda^t$ para t grande, onde λ é o segundo maior autovalor de $\mathbb{T}$.³⁹ Lembramos que o maior autovalor de uma matriz estocástica é 1 e que todos os seus autovalores têm módulo menor ou igual 1. Por essa razão podemos manter apenas o maior termo não trivial da expansão nos autovetores. Assim, a probabilidade de que o agente encontre o máximo global precisamente na geração $t = t^* = 1, 2, \dots$ é $p_1(t^*) = \pi_N(t^*) - \pi_N(t^* - 1)$, ou seja,

$$p_1(t^*) = (1 - \lambda) \lambda^{t^*-1} \quad (3.5)$$

que é uma distribuição geométrica com probabilidade de sucesso $1 - \lambda$. Devemos enfatizar que esse resultado é válido apenas para t^* grande, devido ao truncamento da expansão nos autovetores de $\mathbb{T}$. Usamos o subíndice 1 em $p_1(t^*)$ para indicar que $M = 1$.

Para o caso de M agentes independentes explorando o relevo NK, a geração t^* em que o máximo global é encontrado é a variável aleatória

$$t^* = \min(t_1^*, t_2^*, \dots, t_M^*) \quad (3.6)$$

onde t_i^* são variáveis aleatórias independentes distribuídas pela distribuição geométrica da equação (3.5). Nosso problema agora é obter a distribuição de probabilidade de t^* . Para $M = 2$, temos

$$p_2(t^*) = 2p_1(t^*) \sum_{t=t^*+1}^{\infty} p_1(t) + [p_1(t^*)]^2 = (1 - \lambda^2)(\lambda^2)^{t^*-1} \quad (3.7)$$

e para $M = 3$,

$$p_3(t^*) = p_1(t^*) \sum_{t=t^*+1}^{\infty} p_2(t) + p_2(t^*) \sum_{t=t^*+1}^{\infty} p_1(t) + p_1(t^*)p_2(t^*) = (1 - \lambda^3)(\lambda^3)^{t^*-1}, \quad (3.8)$$

de modo que podemos generalizar

$$p_M(t^*) = (1 - \lambda^M)(\lambda^M)^{t^*-1}. \quad (3.9)$$

Assim, a distribuição de probabilidade da variável aleatória t^* definida pela equação (3.6) é uma distribuição geométrica com probabilidade de sucesso $1 - \lambda^M$. Portanto, o número médio de gerações para encontrar o máximo global é

$$\langle t^* \rangle = \sum_{t^*=1}^{\infty} t^* p_M(t^*) = \frac{1}{1 - \lambda^M}, \quad (3.10)$$

que nos permite obter o custo computacional médio

$$\langle C \rangle = \frac{M \langle t^* \rangle}{2^N} = \frac{M}{2^N(1 - \lambda^M)}. \quad (3.11)$$

O valor do autovalor λ da matriz de transição $\mathbb{T}$ necessita ser calculado numericamente. Por exemplo, para $N = 12$ obtemos $\lambda \approx 0.99978$ e observamos que $\lambda \rightarrow 1$ muito rapidamente com o aumento de N .²⁸ Mas, como já mencionado, nesta dissertação vamos considerar apenas o caso $N = 12$. Como $\lambda \approx 1$ podemos escrever $\lambda^M \approx e^{-M(1-\lambda)}$ de modo que $\langle C \rangle \approx M/2^N$ se $M \gg 1/(1 - \lambda) \approx 4545$ e $\langle C \rangle \approx 1/[2^N(1 - \lambda)] \approx 1.12$, independente de M , se $M \ll 4545$. Assim, observamos dois comportamentos do custo médio computacional com o número de agentes M . Para M pequeno, $\langle C \rangle$ é praticamente independente de M , de modo que a equação (3.1) implica que $\langle t^* \rangle \propto 1/M$. Esse resultado é esperado quando os agentes exploram regiões distintas do espaço de configurações, ou seja, quando não há sobreposição das regiões exploradas pelos agentes. Já para M muito grande, há uma grande chance que o máximo global seja encontrado durante a criação da população inicial, ou em poucas gerações, de modo que $\langle t^* \rangle \propto \text{cte}$.

Outra grandeza de interesse é a função distribuição acumulada definida como

$$\Pi_M(t^*) = \sum_{t=1}^{t^*} p_M(t) = 1 - \lambda^{Mt^*} \quad (3.12)$$

que dá a probabilidade de que a busca cega encontre o máximo global numa geração $t \leq t^*$. Na figura 8 mostramos a comparação entre os resultados obtidos com a equação (3.12) e as simulações da busca cega. Como esperado, a aproximação não é boa para t^* pequeno, uma vez que a manutenção apenas dos dois maiores autovalores na expansão nos autovetores da matriz $\mathbb{T}$ só é justificável para t^* grande.

3.2 Algoritmos evolucionários

Os algoritmos evolucionários foram provavelmente os primeiros algoritmos de inspiração biológica¹ propostos. De fato, os algoritmos evolucionários foram propostos originalmente por John Holland em meados da década de 60³ com objetivo de explicar de maneira formal os processos adaptativos dos sistemas naturais e desenvolver sistemas artificiais capazes de reter propriedades importantes desses sistemas, além de ter um bom nível de eficiência na busca por soluções de problemas de otimização. Desde então esses algoritmos foram aplicados em diversas áreas como matemática, ciências naturais e engenharias na solução de problemas como reconhecimento de padrões, processamento de imagens, otimização de funções, teoria de jogos e aprendizado de máquina entre outros.^{4,5}

Como o nome indica, os algoritmos evolucionários baseiam-se no princípio da evolução pela seleção natural, adaptando a ideia da reprodução dos indivíduos mais adaptados a uma esfera abstrata onde os indivíduos ou agentes são representados por

Figura 8 – Comparação entre a aproximação analítica da função distribuição acumulada (linhas cheias) e os resultados das simulações (símbolos) da busca cega para $M = 1, 10, 100$ como indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas em um único relevo com $N = 12$ e $K = 0$. Os resultados não dependem do parâmetro de epistasia.

Fonte: Elaborada pela autora.

sequências binárias, que têm suas chances de reprodução determinadas por uma função *fitness*. A evolução da população ocorre através da troca de informações entre os agentes, que alteram os bits de suas sequências de modo estruturado, ainda que não-determinístico. Cada geração é formada a partir de recombinações e modificações dos agentes da geração anterior. Dessa forma, ao longo do tempo, os agentes exploram o espaço de configurações utilizando o conhecimento do relevo de fitness transmitido pelas gerações precedentes através da herança genética.

A seguir faremos uma breve descrição da implementação adotada para as duas variantes dos algoritmos evolucionários analisadas, a saber, a variante sexuada em que ocorre recombinação e a variante assexuada em que não ocorre recombinação. Vale ressaltar a importância do operador de seleção para ambas variantes. Ao aplicar esse operador estamos selecionando quais agentes da geração anterior contribuirão na composição da nova geração. Essa seleção ou sorteio é feita com probabilidade proporcional ao *fitness* dos agentes, permitindo que a população utilize as informações adquiridas ao longo de sua evolução para influenciar as futuras gerações.

3.2.1 Algoritmo genético assexuado (AGA)

Esse é o algoritmo evolucionário mais simples que faz uso apenas do mecanismo de reprodução diferenciada, ou seja, do operador seleção. Na versão simplista da reprodução

Figura 9 – Custo computacional médio do AGA em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.

Fonte: Elaborada pela autora.

assexuada usada nesta dissertação, consideramos um único pai que dá origem a um único filho. Lembramos que esse processo ocorre na natureza (em bactérias, por exemplo) através da divisão por mitose e, portanto, um pai deveria produzir dois filhos.⁴⁰ Adotamos essa variante simplista da reprodução assexuada pois nosso objetivo é estudar a influência do operador seleção na eficiência da busca pelo máximo global de relevos NK e não modelar processos biológicos com fidelidade.

Assim, no AGA considerado o pai é um dos M agentes da geração atual sorteado com probabilidade proporcional ao seu *fitness*. Essa forma de escolher os agentes que passarão seus genes (ou bits) para a próxima geração é o operador seleção.⁴ A configuração (ou cromossomo) do agente filho é idêntica à do agente pai. Lembramos que na formulação em que mutação e seleção são processos independentes, em média apenas $(1 - u)M$ membros da nova geração são produzidos por esse processo e os membros restantes (uM em média) são agentes da geração anterior com apenas um de seus bits invertidos. Como veremos a seguir, essa ineficiência em produzir diversidade, já que em média $(1 - u)M$ agentes da nova geração também estavam na geração anterior, terá consequências drásticas para o desempenho do AGA.

A figura 9 mostra a influência da probabilidade de mutação no custo computacional médio do AGA para relevos com diferentes rugosidades $K = 0, 3$ e 6 . O comportamento qualitativo do custo computacional é pouco influenciado pela rugosidade do relevo ou pelo número de agentes M . Chama a atenção nessa figura, o péssimo desempenho do AGA em relação à busca cega. Isso é devido a ausência de um mecanismo de geração de

diversidade, além do processo de mutação. O desempenho do AGA é particularmente ruim para valores pequenos da probabilidade de mutação u . De fato, para $u = 0$ o algoritmo pode simplesmente não encontrar o máximo global se a população inicial tiver uma componente fixada com o bit contrário ao do máximo global. Dizemos que uma componente está fixada quando todos os agentes exibem o mesmo estado (bit) naquela componente. Como já mencionado, para $u = 1$ esse algoritmo reduz-se à busca cega. Essa figura mostra também o comportamento não monotônico do custo computacional médio com o tamanho da população que iremos explorar a seguir.

A figura 10 mostra a dependência do custo computacional médio com o número de agentes M . Para o relevo suave ($K = 0$) e para o relevo pouco rugoso ($K = 3$) existe um número ótimo de agentes que maximiza o desempenho do algoritmo no caso de u não ser muito grande. Provavelmente, a razão porque o custo é alto para M pequeno é a maior probabilidade de ocorrerem fixações na população inicial, o que exige um grande número de iterações para que essa situação seja revertida quando u é pequeno. Mais interessante é a razão porque o custo computacional é alto quando M é grande, resultado que observamos também na figura 9. Esse é um efeito da dinâmica tornar-se mais determinística com o aumento de M e, portanto, favorecer fortemente os agentes com maior *fitness* o que resulta em uma drástica diminuição da diversidade.³⁴ Em outras palavras, o aumento de M amplifica o efeito da seleção. Lembremos que o custo computacional definido pela equação (3.1) é proporcional à M . Portanto, se o tamanho da população efetiva for muito menor que M , pois há muitas cópias dos mesmos agentes, o algoritmo paga o preço no custo sem ter as vantagens de explorar regiões distintas do relevo que uma população diversa traria.

3.2.2 Algoritmo genético sexuado (AGS)

Esse algoritmo evolucionário é mais sofisticado que o anterior, pois, além da reprodução diferenciada, incorpora o mecanismo de recombinação, que consegue manter a diversidade da população por longos períodos. O algoritmo genético sexuado busca imitar o processo no qual dois indivíduos dão origem a um novo através da mistura de seus genes. Em nossa analogia, os pais correspondem a dois agentes distintos da geração atual sorteados com probabilidade proporcional aos seus *fitness* (operador seleção),⁴ enquanto o agente filho corresponde ao agente que será passado para a próxima geração. A configuração (ou cromossomo) do agente filho é determinada pelo operador recombinação.⁴ Escolhe-se aleatoriamente um inteiro $P_c \in \{1, 2, \dots, N - 2\}$ para ser o ponto de recombinação e forma-se a configuração do agente filho usando as componentes 0 até P_c do primeiro pai e as componentes $P_c + 1$ até $N - 1$ do segundo pai. Note que com essa escolha do ponto de recombinação P_c garantimos que o agente filho sempre será um recombinante, ou seja, terá componentes herdadas de ambos os pais. A figura 11 mostra o esquema desse procedimento.

Figura 10 – Custo computacional médio do AGA em função do número de agentes M para os valores da probabilidade de mutação u indicados. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.

Fonte: Elaborada pela autora.

Enfatizamos que esse procedimento – seleção e recombinação – contribui em média com $(1 - u)M$ agentes para a nova geração. Como no caso do AGA, o restante dos agentes da nova geração passam apenas pelo processo de mutação e, portanto, passam para a próxima geração com um bit invertido independentemente de seus valores de *fitness*.

A figura 12 mostra como a probabilidade de mutação u influencia o desempenho do AGS para um número fixo de agentes. O operador recombinação mostra-se um poderoso mecanismo de geração de diversidade, uma vez que garante que os filhos sejam recombinantes, capaz de amenizar um pouco o efeito redutor de diversidade do operador seleção para M grande. Mas, ainda assim, a mutação é essencial para impedir a fixação das componentes, especialmente para M pequeno. Por exemplo, a probabilidade de que a população inicial tenha uma componente fixa com o bit oposto ao do máximo global é $N/2^M$, o que dá em torno de 0.012 para $M = 10$ e em torno de 10^{-30} para $M = 100$. É por essa razão que vemos o custo computacional divergir para $M = 10$ quando $u \rightarrow 0$ no caso do relevo suave ($K = 0$), mas não vemos a divergência para $M = 100$ e $M = 1000$. Naturalmente, se considerarmos 10^{30} amostras veríamos a divergência para $M = 100$ também. A presença de máximos locais nos relevos para $K = 3$ e $K = 6$ reduz bastante a eficiência do AGS que necessita da mutação para escapar dessas armadilhas, mas o aumento de u torna o algoritmo mais similar a busca cega.

A figura 13 mostra o efeito do tamanho da população M no desempenho do AGS. Como esperado, para M e u pequenos o desempenho é prejudicado pela dificuldade em lidar com a baixa diversidade da população inicial e reverter possíveis fixações. Para M

Figura 11 – Esquema do algoritmo genético sexuado (AGS) mostrando os operadores de seleção e de recombinação utilizados para criar agentes da nova geração. Em média $(1 - u)M$ novos agentes são criados segundo esse procedimento.

Fonte: Elaborada pela autora.

Figura 12 – Custo computacional médio do AGS em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.

Fonte: Elaborada pela autora.

Figura 13 – Custo computacional médio do AGS em função do número de agentes M para os valores da probabilidade de mutação u indicados. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.

Fonte: Elaborada pela autora.

grande o problema é a redução de diversidade produzida pelo operador seleção usado na escolha dos pais. Devemos notar que, na ausência de mutação, o AGS eventualmente produzirá uma população isogênica (isto é, todos os agentes são iguais) apesar do operador recombinação.⁴¹ Afinal, se os pais tiverem sequências muito parecidas, o filho terá também uma sequência parecida. Quando os pais tiverem sequências idênticas, o filho será idêntico aos pais. Assim, o operador recombinação retarda mas não elimina o efeito do operador seleção. Assim, para u fixo sempre existe um tamanho de população ótimo que minimiza o custo computacional. É interessante que, ao contrário do AGA, sempre é possível ajustar os parâmetros M e u tal que o desempenho do AGS seja superior ao da busca cega, embora a vantagem torne-se cada vez menos significativa à medida que a rugosidade do relevo aumenta.

Esses resultados ilustram a importância da escolha dos parâmetros M e u para o desempenho do AGS e revelam o fato importante de que os parâmetros ótimos dependem do relevo, de modo que prescrições genéricas aplicáveis para todos os problemas não são confiáveis.

3.3 Aprendizado por imitação (AI)

Os algoritmos evolucionários abordados na seção anterior tornaram-se bastante populares devido à ampla variedade de aplicações possíveis, já que podem, em princípio, ser aplicados em qualquer problema de otimização.⁴ Essa generalidade pode ser vista como indicação de que a eficiência desses algoritmos não deve ser comparável com a de

Figura 14 – Esquema do algoritmo aprendizado por imitação (AI) mostrando o processo de cópia de um único bit do agente modelo pelo agente imitador. Em média $(1 - u)M$ novos agentes são criados segundo esse procedimento.

Fonte: Elaborada pela autora.

algoritmos criados para tratar um problema de otimização específico, mas não iremos tratar diretamente dessa questão nesta dissertação.

Uma alternativa aos algoritmos evolucionários, de mesma generalidade mas bem menos conhecida, são os algoritmos inspirados nos mecanismos sociais que embasam a evolução cultural.^{11,12,42} Embora nas últimas décadas tenha ocorrido um crescimento do interesse nas abordagens da mecânica estatística e sistemas dinâmicos ao estudo de sistemas sociais, focando principalmente na emergência de fenômenos coletivos,^{43,44} a perspectiva dos ditos algoritmos culturais é distinta, pois usa a dinâmica social para solucionar problemas. Desse modo, o estudo do desempenho desses algoritmos tem relevância para a área da organização do trabalho, cujo objetivo é ajustar ou regular as interações sociais no ambiente de trabalho de modo a maximizar o desempenho do grupo.^{25,27,45,46} De fato, uma das questões básicas é entender quando a cooperação pode ser prejudicial ao desempenho do grupo, no sentido que o grupo encontraria a solução mais rapidamente se os agentes trabalhassem de forma independente.¹³

Os humanos são sabidamente os melhores imitadores da natureza⁴⁷ e é possível que essa capacidade de imitação tenha sido responsável pela sua cultura complexa.⁴⁸ Assim, o algoritmo cultural que iremos implementar utiliza como ingrediente principal a cópia ou imitação do indivíduo (ou agente) de maior *fitness* no grupo, daí o nome de aprendizado por imitação (AI).¹³

O primeiro passo na implementação do AI é identificar a cada geração qual é o

Figura 15 – Custo computacional médio do AI em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.

Fonte: Elaborada pela autora.

agente que tem o maior *fitness* do grupo. Esse agente é denominado de agente modelo. O agente modelo tem grande impacto na dinâmica do algoritmo, pois em média $(1 - u)M$ agentes do grupo (os imitadores) serão alterados ao copiar uma de suas componentes na tentativa de se tornarem mais parecidos com o modelo. O processo de imitação entre um agente imitador e o modelo começa com a identificação das componentes que diferem nas configurações desses agentes. Uma dessas componentes é então sorteada aleatoriamente e o agente imitador copia o bit dessa componente. Na prática isso quer dizer que o imitador inverte o bit dessa componente em sua configuração, de modo que agora essa componente seja igual a do modelo. Como esperado, após a imitação o agente imitador e o modelo ficam mais parecidos entre si. Caso o agente imitador seja idêntico ao próprio modelo, o que não é uma situação incomum já que o processo de imitação reduz a diversidade da população que pode conter várias cópias de uma mesma sequência, então o agente imitador não é atualizado. Esse mecanismo de cópia é inspirado no mecanismo usado para estudar a influência da mídia^{49,50} no modelo de Axelrod de disseminação cultural.⁵¹ Essa, aliás, é uma das razões porque dizemos ser o AI inspirado na evolução cultural.

A figura 14 ilustra os detalhes da implementação do algoritmo AI. Como nos algoritmos evolucionários descritos anteriormente, apenas uma fração dos agentes passa pelo processo de imitação, o restante passa pelo processo de mutação.

A figura 15 explora a variação do custo computacional do AI com a probabilidade de mutação. Para o relevo suave ($K = 0$) o desempenho do AI é muito menos sensível a falta de geração de diversidade resultante de uma pequena probabilidade de mutação u do

que os algoritmos evolucionários. Por exemplo, para $M = 10$ apenas para $u < 10^{-3}$ o custo computacional do AI torna-se maior que o da busca cega. Como já mencionado, devemos ter $\langle C \rangle \rightarrow \infty$ para $u \rightarrow 0$ devido a possibilidade de que a população inicial tenha uma componente fixa contrária a do máximo global. Entretanto, para relevos rugosos ($K = 3$ e $K = 6$) a situação se inverte e o desempenho do AI é o mais negativamente afetado pela diminuição da probabilidade de mutação. O problema aqui é o aprisionamento nos máximos locais. De fato, uma probabilidade de imitação $1 - u$ grande faz com que os agentes copiem o modelo com mais frequência e, caso o modelo esteja num máximo local, acaba por levar toda a população para uma armadilha. Há uma correspondência desse fenômeno na psicologia social – o pensamento de grupo – onde os membros do grupo acabam por adotar o pensamento do líder com resultados frequentemente desastrosos.⁵² Há várias formas de amenizar esse efeito como permitir que cada agente tenha uma propensão para imitar $1 - u$ distinta,⁵³ diminuir a influência do agente modelo limitando a conectividade entre os agentes⁵⁴ ou permitindo que os agentes se movimentem no plano.⁵⁵

A figura 16 mostra como o custo computacional do AI é influenciado pelo tamanho da população M . O aspecto interessante dessa figura é o desempenho catastrófico do algoritmo para relevos rugosos ($K = 3$ e $K = 6$). Fica claro que populações grandes (mas não tão grandes a ponto de haver uma probabilidade razoável de que o máximo global seja encontrado já na montagem da população inicial) intensificam o fenômeno do pensamento de grupo, ou seja, o aprisionamento nos máximos locais. Não é difícil entender a razão disso: quando há muitos clones do agente modelo cuja configuração é um máximo local, mesmo que alguns deles sofram mutações e se afastem do máximo local, eles serão atraídos novamente ao imitarem os clones que não sofreram mutação. O desempenho do AI não é tão afetado pelos máximos locais quando M é pequeno, pois existe uma chance de todos os clones sofrerem mutação.

Apesar da grande sensibilidade do desempenho do AI com a escolha dos parâmetros u e M , o ajuste fino desses parâmetros produz um desempenho comparável ou superior ao dos algoritmos evolucionários como veremos no próximo capítulo. Finalizamos essa seção observando que o fracasso do AI em certas circunstâncias é justamente o que torna o algoritmo interessante sob a perspectiva da solução coletiva de problemas, uma vez que sabemos que transmissão cultural e social pode passar informação falsa que resulta em comportamentos mal adaptativos.⁵⁶ Isso é mais difícil, senão impossível, de acontecer no contexto evolucionário já que se o organismo sobreviveu até a idade reprodutiva e conseguiu um parceiro fértil, seus genes são provavelmente muito bem adaptados.

Figura 16 – Custo computacional médio do AI em função do número de agentes M para os valores da probabilidade de mutação u indicados. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. Os painéis mostram os resultados para relevos de diferentes rugosidades K como indicado. A dimensionalidade dos relevos é $N = 12$ em todos os casos.

Fonte: Elaborada pela autora.

4 COMPARAÇÃO DO DESEMPENHO DOS ALGORITMOS

No capítulo anterior, discutimos o desempenho dos algoritmos de busca em função dos parâmetros u (probabilidade de mutação) e M (tamanho da população) separadamente. Agora vamos comparar os seus desempenhos de uma forma mais conveniente, exibindo o custo computacional dos quatro algoritmos nas mesmas figuras. Lembramos que usamos o desempenho da busca cega como medida da eficiência dos algoritmos cooperativos. Como mencionado anteriormente, esperamos que esses algoritmos apresentem desempenho muito superior ao da busca cega, já que permitem a troca de informação entre os agentes e privilegiam configurações com maior *fitness*.

Além da capacidade de guiar o sistema na direção de um *fitness* maior, é necessário manter a diversidade dos estados dos agentes ao longo desse processo, pois a redução da diversidade da população pode causar o aprisionamento do algoritmo em configurações que não correspondem à solução ótima, especialmente ao tentar solucionar problemas mais complexos com diversos máximos locais (relevos rugosos). Uma das formas de garantir a diversidade da população é aumentar o valor da probabilidade de mutação u , mas isso diminui a assertividade da busca e o algoritmo cooperativo tende a um desempenho próximo ao da busca cega conforme u se aproxima de 1. Por assertividade entendemos a tendência a realizar mudanças que aumentem o *fitness*. Curiosamente, o aumento do tamanho da população M não é uma maneira muito eficiente de se aumentar a diversidade nos algoritmos cooperativos. Embora a população inicial seja muito diversa e por essa razão os algoritmos conseguem encontrar o máximo global já nas primeiras gerações se $M \gg 2^N$ (todos os algoritmos têm um desempenho similar nesse caso), ela rapidamente perde diversidade com a geração de muitos clones que não contribuem para a busca (apenas duplicam trabalho) mas que pesam negativamente no custo computacional.

Nosso objetivo é entender quantitativamente em quais situações os algoritmos cooperativos superam significativamente o desempenho da busca cega e encontrar os parâmetros u e M que otimizam o *trade off* entre diversidade e assertividade para cada um dos algoritmos cooperativos.

4.1 Relevo suave

Nesta seção vamos considerar o caso $K = 0$, de modo que o relevo possui apenas um máximo e, portanto, não há chance da busca ficar estagnada em máximos locais. É claro que encontrar o único máximo em um relevo suave é um problema de busca extremamente fácil que um algoritmo guloso (ou seja, um algoritmo que escolhe a atualização que resulta no maior aumento do *fitness*) resolve em N passos no máximo. Embora a diversidade sempre tenha um papel importante, nesse cenário a eficiência do algoritmo estará muito

associada a sua capacidade de gerar alterações das configurações que aumentem o *fitness* dos agentes, levando a busca rapidamente ao máximo global, visto que, para esse tipo de relevo, o *fitness* de um agente é uma medida confiável da sua proximidade ao máximo global.

A figura 17 mostra o desempenho dos algoritmos em função do número de agentes M para probabilidade de mutação u fixa e a figura 18 o desempenho em função de u para M fixo. O algoritmo de melhor desempenho é o aprendizado por imitação (AI), independentemente da escolha dos parâmetros u e M . É surpreendente que tanto o algoritmo genético assexuado (AGA) como o sexuado (AGS) tenham desempenhos não muito superiores ao da busca cega mesmo para um relevo sem máximos locais. Para valores baixos de u , os algoritmos evolucionários têm desempenho inferior ao da busca cega, conforme percebe-se na figura 18, ao contrário do AI que apresenta desempenho inferior ao da busca cega apenas para $u < 10^{-3}$ no caso $M = 10$. O mau desempenho dos algoritmos evolucionários nesses casos é devido a baixa eficácia do operador seleção (ou da roleta genética⁴) para escolher o agente com maior *fitness* entre agentes com *fitness* próximos. Lembramos que a correlação entre os *fitness* de configurações vizinhas é $1/N$ (veja equação 2.10) no relevo com $K = 0$, de modo que a roleta genética terá dificuldades em escolher o melhor agente. Essa situação piora no caso de populações pequenas, pois agora há uma possibilidade razoável do agente com *fitness* mais alta não ser escolhido para passar a próxima geração. Já o AI realiza uma escolha determinística do melhor agente e, portanto, não encontra dificuldade alguma nessa escolha, o que explica seu desempenho superior no relevo suave.

Como podemos verificar na figura 17 existe um tamanho ótimo da população que minimiza custo computacional, de forma que com um ajuste fino do parâmetro M todos os algoritmos cooperativos podem exibir um desempenho superior ao da busca cega. A tabela 2 mostra o valor ótimo de M para cada um dos algoritmos cooperativos, bem como o valor do custo computacional associado. O desempenho dos algoritmos cooperativos começa a se deteriorar para populações maiores que esse ótimo devido a redução de diversidade inerente a esses algoritmos, que produzem muitas cópias dos mesmos agentes, que não oferecem contribuição adicional para o sucesso da busca. Esse efeito de duplicação de trabalho também é observado na busca cega mas apenas para populações muito grandes ($M \gg 2^N$) pois esse algoritmo não reduz a diversidade da população de agentes. Observa-se também na tabela 2 que o tamanho ótimo da população é consideravelmente menor para o AI. Isso ocorre porque esse algoritmo tende a diminuir a diversidade mais intensamente que os outros algoritmos.

Analogamente, também há um valor ótimo para a probabilidade de mutação, como pode ser observado na figura 18 e na tabela 3. Esse resultado é esperado visto que no limite em que $u \rightarrow 1$ os algoritmos cooperativos são equivalentes à busca cega e que para $u \rightarrow 0$

Figura 17 – Custo computacional médio dos quatro algoritmos de busca em função do número de agentes M para a probabilidade de mutação u indicada. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. A dimensionalidade do relevo é $N = 12$ e o parâmetro de epistasia é $K = 0$, ou seja, não há máximos locais.

Fonte: Elaborada pela autora.

Figura 18 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. A dimensionalidade do relevo é $N = 12$ e o parâmetro de epistasia é $K = 0$, ou seja, não há máximos locais.

Fonte: Elaborada pela autora.

Tabela 2 – Valores ótimos do tamanho da população para os três algoritmos cooperativos para as probabilidade de mutação u utilizadas na figura 17. O aprendizado por imitação apresenta os menores custos para todos os valores de u

Algoritmo	u	$M_{\text{ótimo}}$	$\langle C \rangle_{\min}$
AGA	0.3	44	0.89
AGS	0.3	47	0.35
AI	0.3	5	0.04
AGA	0.5	34	0.90
AGS	0.5	47	0.47
AI	0.5	8	0.06
AGA	0.7	34	0.97
AGS	0.7	33	0.65
AI	0.7	34	0.16
AGA	0.9	12	1.07
AGS	0.9	39	0.94
AI	0.9	120	0.55

Fonte: Elaborada pela autora.

a falta de diversidade dificulta as buscas e pode até mesmo não ser possível encontrar o máximo global para determinadas condições iniciais dos agentes.

Até esse momento consideramos relevos NK com dimensionalidade fixa $N = 12$. A figura 19 mostra como o desempenho dos algoritmos é afetado pela variação da dimensão N do espaço de configurações, mantendo o tamanho da população M fixo. Nota-se que, enquanto os algoritmos evolucionários apresentam uma melhora tímida com o aumento da dimensionalidade, o custo computacional do AI decresce exponencialmente com o aumento de N . Isso quer dizer que, ao contrário da busca cega cujo tempo médio de busca escala com 2^N , o tempo médio de busca do AI escala com alguma potência de N . Lembramos que o tempo médio de busca da estratégia ótima para o relevo com $K = 0$, a busca gulosa, escala linearmente com a dimensionalidade N .

Em resumo, para o relevo sem máximos locais ($K = 0$), o bom desempenho do aprendizado por imitação deve-se a sua capacidade de sempre escolher o agente com melhor *fitness*, enquanto a roleta genética do operador seleção faz essa escolha de modo muito ineficiente quando os agentes têm valores de *fitness* muito próximos entre si. Para a roleta ser eficiente é necessário que haja uma diferença considerável entre o *fitness* dos agentes. Vale notar que, sob o aspecto da influência social, a imitação não trouxe problemas para a busca pois no relevo considerado o *fitness* é um excelente indicativo da proximidade ao máximo global. Ou seja, a informação difundida pelo agente modelo é honesta nesse caso.

Tabela 3 – Valores ótimos da probabilidade de mutação para os três algoritmos cooperativos para os tamanhos da população M utilizados na figura 18. O aprendizado por imitação apresenta os menores custos para todos os valores de M

Algoritmo	M	$u_{\text{ótimo}}$	$\langle C \rangle_{\min}$
AGA	10	0.6	1.04
AGS	10	0.85	1.07
AI	10	0.25	0.04
AGA	100	0.4	0.94
AGS	100	0.1	0.38
AI	100	0.01	0.11
AGA	1000	1.0	1.19
AGS	1000	0.01	0.81
AI	1000	0.01	0.51

Fonte: Elaborada pela autora.

Figura 19 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para relevos suaves ($K = 0$) com dimensionalidades $N = 9, 12$ e 15 como indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O número de agentes é $M = 10$.

Fonte: Elaborada pela autora.

Figura 20 – Custo computacional médio dos quatro algoritmos de busca em função do número de agentes M para a probabilidade de mutação u indicada. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O relevo com $N = 12$ e $K = 6$ tem 114 máximos.

Fonte: Elaborada pela autora.

4.2 Relevo rugoso

Nessa seção vamos considerar principalmente o desempenho dos algoritmos de busca na exploração do relevo rugoso de parâmetros $N = 12$ e $K = 6$, que tem 114 máximos, sendo que apenas um é máximo global. Como na seção anterior, iremos também considerar muito brevemente o desempenho dos algoritmos em relevos com dimensionalidades $N = 9$ e $N = 15$. A presença de 113 máximos locais resulta em uma probabilidade razoável de que a busca fique estagnada em regiões distantes do máximo global, resultando em um aumento do custo computacional. Nesse cenário, os mecanismos de geração de diversidade têm um papel fundamental na eficiência da busca, pois permitirão que os algoritmos escapem com rapidez dos máximos locais. Além disso, no relevo rugoso o *fitness* não é mais necessariamente um bom indicativo de proximidade ao máximo global e esperamos observar uma deterioração do desempenho dos algoritmos que se baseiam fortemente nessa característica. De fato, a busca gulosa, que é a estratégia ótima para o relevo suave, tem grandes chances de nunca encontrar o máximo global de nosso relevo rugoso.

A figura 20 compara o desempenho dos algoritmos em função do tamanho da população M para alguns valores da probabilidade de mutação. Notamos, que ao contrário do observado no caso do relevo suave, agora o AI apresenta um desempenho inferior ao dos

Tabela 4 – Valores ótimos do tamanho da população para os três algoritmos cooperativos para as probabilidade de mutação u utilizadas na figura 20.

Algoritmo	u	$M_{\text{ótimo}}$	$\langle C \rangle_{\min}$
AGA	0.3	3310	2.26
AGS	0.3	490	1.03
AI	0.3	4	2.26
AGA	0.5	58	1.68
AGS	0.5	144	0.98
AI	0.5	6	0.67
AGA	0.7	82	1.40
AGS	0.7	230	1.02
AI	0.7	34	0.67
AGA	0.9	12	1.20
AGS	0.9	57	1.08
AI	0.9	120	0.94

Fonte: Elaborada pela autora.

outros algoritmos, incluindo a busca cega, para valores baixos da probabilidade de mutação u e valores intermediários de M . A enorme influência exercida pelo agente modelo (ou seja, o agente com maior *fitness*) nos outros agentes passa a ser desvantajosa, pois nesse caso a imitação pode direcionar a busca para um máximo local. Adicionando-se a forte redução da diversidade do grupo para valores baixos de u , obtemos o desempenho catastrófico do AI observado para $u = 0.3$, por exemplo. Como já mencionado, esse desempenho catastrófico é similar ao fenômeno social de ‘pensamento de grupo’, no qual todos os membros de um grupo passam a pensar de forma muito parecida, devido a fatores como a influência de um líder.

No entanto, o desempenho ruim para probabilidades de mutação baixas não se limita ao AI. Tanto o AGS quanto o AGA têm desempenho inferior ou equivalente ao da busca cega para boa parte dos valores de M analisados. Além disso, quando analisamos o valor ótimo do parâmetro M , mantendo os valores de u fixos, como mostrado na tabela 4, notamos que o AI supera os demais algoritmos em todos os casos, exceto para $u \leq 0.3$, embora apresente desempenho muito inferior aos demais algoritmos quando o número de agentes não é escolhido de maneira ótima.

Assim como ocorre com o parâmetro M , a figura 21 mostra que também há uma probabilidade de mutação que otimiza o desempenho do AI para um dado tamanho de população. Já para os algoritmos evolucionários AGA e AGS, a probabilidade de mutação ótima é bem próxima de $u = 1$, implicando então que para problemas mais complexos esses algoritmos não apresentam ganho substancial em relação à busca cega. A tabela 5

Figura 21 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para o número de agentes M indicado. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O relevo com $N = 12$ e $K = 6$ tem 114 máximos.

Fonte: Elaborada pela autora.

Tabela 5 – Valores ótimos da probabilidade de mutação para os três algoritmos cooperativos para os tamanhos da população M utilizados na figura 21.

Algoritmo	M	$u_{\text{ótimo}}$	$\langle C \rangle_{\min}$
AGA	10	1.0	1.11
AGS	10	1.0	1.11
AI	10	0.6	0.65
AGA	100	0.95	1.12
AGS	100	0.95	1.10
AI	100	0.8	0.81
AGA	1000	1.0	1.19
AGS	1000	0.9	1.17
AI	1000	0.95	1.15

Fonte: Elaborada pela autora.

mostra que, escolhendo-se convenientemente a probabilidade de mutação como $u = u_{\text{ótimo}}$, o AI supera os demais algoritmos.

Para finalizarmos, vamos considerar agora o efeito da dimensionalidade N dos relevos rugosos no desempenho dos algoritmos de busca. A fim de mantermos a correlação entre o *fitness* de configurações vizinhas (veja (2.10)) invariante ao modificarmos N , devemos também variar K tal que $(K + 1)/N$ seja constante. A figura 22 compara o desempenho dos algoritmos em relevos rugosos com dimensionalidades $N = 9$ (11 máximos), $N = 12$ (39 máximos) e $N = 15$ (119 máximos) mas com o mesmo valor ($2/3$) da correlação local entre o *fitness* de configurações vizinhas. Surpreendentemente, observa-

Figura 22 – Custo computacional médio dos quatro algoritmos de busca em função da probabilidade de mutação u para relevos rugosos com parâmetros $(N = 9, K = 2)$, $(N = 12, K = 3)$ e $(N = 15, K = 4)$ como indicado e com número de máximos iguais a 11, 39 e 119, respectivamente. Cada símbolo representa a média sobre o custo computacional de pelo menos 5×10^4 buscas no mesmo relevo. O número de agentes é $M = 10$.

Fonte: Elaborada pela autora.

se uma melhora no desempenho de todos os algoritmos cooperativos. Isso é explicado pela diminuição da densidade de máximos locais nos relevos rugosos que ocorre com o aumento da dimensionalidade do relevo. De fato, mesmo para o relevo de máxima rugosidade $K = N - 1$, a densidade de máximos locais tende a 0 com $1/(N + 1)$ para N grande (veja (2.14)). Assim, à medida que aumentamos N o relevo rugoso torna-se mais parecido com relevo suave, pois os máximos locais ficam mais raros, o que explica a similaridade qualitativa entre as figuras 22 e 19.

5 CONCLUSÃO

Ao longo dessa dissertação estudamos algoritmos de busca estocásticos, cuja tarefa é encontrar o máximo global de relevos NK¹⁵ com diferentes níveis de complexidade. O principal ingrediente desses algoritmos são os agentes que representam as possíveis configurações dos relevos NK. O desempenho da busca é medido pelo seu custo computacional que é, essencialmente, o número total de atualizações realizadas nos agentes até que o máximo global seja encontrado.

Iniciamos nosso estudo com a formulação matemática do modelo NK e a derivação das principais características dos relevos de *fitness* ou relevos adaptativos gerados por esse modelo (Capítulo 2). Em particular, analisamos como as interações epistáticas, que são controladas pelo parâmetro K , afetam a rugosidade do relevo, determinada pelo número de máximos locais e pela correlação dos *fitness* de configurações vizinhas.

No Capítulo 3, introduzimos o conceito de buscas baseadas em agentes, representados em nossos estudos por sequências binárias de comprimento N , e descrevemos a implementação de três algoritmos ditos cooperativos, a saber, dois algoritmos evolucionários⁴ – os algoritmos genéticos assexuado (AGA) e sexuado (AGS) – e de um algoritmo cultural¹³ – o aprendizado por imitação (AI). Como base de comparação, utilizamos a busca cega onde as configurações dos agentes são atualizadas de maneira completamente aleatória. Buscamos uniformizar a execução das buscas de forma que a única diferença entre os algoritmos seja os operadores que os caracterizam (seleção, recombinação ou imitação). Assim, os algoritmos são implementados de forma que o processo de mutação seja independente do processo de seleção e recombinação (ou imitação, no caso do AI). Esse esquema correspondente ao modelo Crow-Kimura da genética de populações,³⁴ que supõe também que não haja superposição de gerações. Uma vez estabelecido o procedimento geral das buscas, caracterizamos individualmente cada um dos algoritmos, apresentando os esquemas de implementação e seus desempenhos para diferentes relevos, tamanho de população M e probabilidade de mutação u . Além disso, as buscas são realizadas em relevos fixados, de forma que os resultados dos diferentes algoritmos não sejam enviesados por diferenças entre os relevos. Iniciamos nossa descrição pela busca cega, abordando o conceito de movimento elementar (mutação) fizemos uma descrição estatística do algoritmo obtendo analiticamente o tempo médio das buscas em função do tamanho da população M e do comprimento das sequências binárias N . Lembramos que a busca cega não é influenciada pelo parâmetro de epistasia K . A seguir, apresentamos um breve histórico descrevendo a origem e importância dos algoritmos evolucionários e estabelecemos as características das duas variantes (AGA e AGS) utilizadas nesse trabalho. Também introduzimos o AI, algoritmo com base no processo de cooperação social, como uma alternativa

aos tradicionais algoritmos evolucionários. Já nessa etapa discutimos alguns pontos fortes e fracos de cada um dos algoritmos cooperativos em comparação com a busca cega.

Tendo em mãos as informações dos desempenhos dos quatro algoritmos de busca, consolidamos no Capítulo 4 a comparação entre eles com o objetivo de entender quantitativamente em quais situações os algoritmos cooperativos superam significativamente o desempenho da busca cega e encontrar os parâmetros u e M que otimizam o *trade off* entre a diversidade e assertividade para cada um dos algoritmos cooperativos. Abordamos também a influência da dimensionalidade do relevo (ou comprimento das sequências) N nos nossos resultados. Inicialmente comparamos os algoritmos utilizando um relevo suave, ou seja, um relevo sem máximos locais. Nesse cenário era esperado que o desempenho do algoritmos cooperativos fosse muito superior ao da busca cega mas, surpreendentemente, não foi esse o resultado obtido. De fato, os algoritmos evolucionários AGA e AGS apresentam desempenho bastante próximo ao da busca cega, superando-a em algumas faixas dos parâmetros M e u mas mantendo sempre a mesma ordem de grandeza do custo computacional. Nesse cenário destaca-se o bom desempenho do AI, que supera os demais algoritmos de forma consistente para qualquer valor dos parâmetros utilizados. Isso ocorre devido a maior eficiência do operador imitação em comparação ao operador seleção na identificação das configurações de maior *fitness*, além do fato de que nos relevos suaves o *fitness* ser uma informação confiável sobre a proximidade da configuração ao máximo global. Por essa razão, a influência social não traz prejuízo ao algoritmo cultural na exploração de relevos suaves. Finalizamos nossa análise comparativa explorando relevos rugosos, ou seja, relevos que exibem muitos máximos locais. Nesse cenário é importante que o algoritmo de busca seja capaz de equilibrar a habilidade de trocar informações entre os agentes para obter configurações com maior *fitness*, o que chamamos de assertividade, com a habilidade de manter a diversidade das configurações dos agentes, para que a população não fique aprisionada nos máximos locais. Verificamos que, para a maior parte dos parâmetros utilizados, o custo computacional dos algoritmos cooperativos é equivalente ou inferior ao custo computacional da busca cega, destacando-se a baixa eficiência do AI que exhibe resultados catastróficos para u pequeno e tamanho intermediários da população M , de maneira similar ao que ocorre no fenômeno social de ‘pensamento de grupo’.⁵²

Após os resultados apresentados nos aproximamos de responder a pergunta que motivou esse estudo – algoritmos sociais são menos eficientes que os genéticos? Com base nos dados apresentado é possível afirmar que, enquanto os algoritmos genéticos apresentam em geral desempenhos mais regulares e próximos da busca cega, o aprendizado por imitação tem resultados mais extremos, em alguns casos destacando-se pelo desempenho excelente e em outros por resultados catastróficos. No entanto, vimos que em quase todos os cenários, o AI é capaz de superar os algoritmos evolucionários quando feita uma escolha cuidadosa dos parâmetros u e M adotados para a busca. O baixo ganho dos algoritmos em relação à busca cega está de acordo com o fato que encontrar o máximo global de um relevo do

tipo NK com $K > 0$ é um problema NP completo²⁹ e, portanto, esperamos que nenhum algoritmo seja capaz de reduzir consideravelmente o tempo necessário para encontrar a solução ótima.

Uma crítica importante as nossas conclusões é que elas são baseadas na análise de apenas um relevo para cada par dos parâmetros N e K utilizados. Como visto no Capítulo 2, exceto para $K = 0$, há uma grande variabilidade das características dos relevos gerados pelo modelo NK e, conseqüentemente, dos resultados das buscas realizadas neles. Outra crítica menos importante é que a variante dos algoritmos evolucionários utilizada na nossa análise, onde o processo de mutação é independente dos processos de seleção e recombinação, não é a comumente implementada nos algoritmos evolucionários. Finalmente, uma crítica interessante é que em problemas do mundo real o desempenho dos algoritmos não é medido pelo custo computacional, uma vez que não conhecemos a solução ótima (o máximo global, no caso) e o tempo de execução computacionalmente factível da busca é tipicamente ordens de magnitude menor que o tempo necessário para encontrar a solução. Desse modo, o critério de desempenho para problemas do mundo real é a maior *fitness* encontrada para um tempo de execução fixo. Essa crítica é curiosa pois implicitamente admite que o algoritmo que encontra a solução mais rapidamente quando o tempo de busca é ilimitado, pode não ser o mesmo que encontra a melhor solução quando o tempo de busca é limitado. Esses pontos foram cuidadosamente abordados, demonstrando que as conclusões dessa dissertação permanecem válidas quando considera-se diversas realizações dos relevos NK, quando os algoritmos evolucionários são implementados da forma usual, bem como quando o critério de desempenho é modificado para levar em conta tempo de busca fixado a priori.⁵⁷

Considerando a natureza quase que iconoclástica de nossos resultados, que desveste o mito da eficiência dos algoritmos evolucionários, torna-se urgente repetir esse estudo para outros problemas de busca, a fim de delimitarmos a generalidade de nossas conclusões.

REFERÊNCIAS

- 1 KAR, A.K. Bio inspired computing—a review of algorithms and scope of applications. *Expert Systems with Applications*, v. 59, p. 20–32, 2016. DOI: 10.1016/j.eswa.2016.04.018.
- 2 KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. *Science*, v. 220, n. 4598, p. 671–680, 1983.
- 3 HOLLAND, J.H. *Adaptation in natural and artificial systems*. Cambridge, MA: MIT Press, 1992.
- 4 GOLDBERG, D.E. *Genetic algorithms in search, optimization, and machine learning*. Reading: Addison-Wesley, 1989.
- 5 BÄCK, T. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford: Oxford University Press, 1996.
- 6 NEUMANN, J. von. *Theory of self-reproducing automata*. Urbana: University of Illinois Press, 1966.
- 7 ULAM, S. M. *A collection of mathematical problems*. New York: Interscience, 1960.
- 8 MINSKY, M. Steps toward artificial intelligence. *Proceedings of the IRE*, v. 49, n. 1, p. 8–30, 1961.
- 9 BURROWS, W. E. *This new ocean: the story of the first space age*. New York: Modern Library, 1999.
- 10 GROVES, L. *Now it can be told: the story of the Manhattan project*. New York: Harper & Row, 1962.
- 11 KENNEDY, J. Thinking is social: experiments with the adaptive culture model. *Journal of Conflict Resolution*, v. 42, n. 1, p. 56–76, 1998.
- 12 KENNEDY, J. Minds and cultures: particle swarm implications for beings in sociocognitive space. *Adaptive Behavior*, v. 7, n. 3-4, p. 269–288, 1999.
- 13 FONTANARI, J. F. Imitative learning as a connector of collective brains. *PLoS ONE*, v. 9, n. 10, p. e110517, 2014.
- 14 TOLENTINO, J. *Trick mirror: reflections on self-delusion*. New York: Random House, 2019.
- 15 KAUFFMAN, S.A.; LEVIN, S. Towards a general theory of adaptive walks on rugged landscapes. *Journal of Theoretical Biology*, v. 128, n. 1, p. 11–45, 1987.
- 16 WRIGHT, S. The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In: INTERNATIONAL CONGRESS ON GENETICS, 6., 1932, Ithaca. *Proceedings [...]*. New York: Brooklyn Botanic Garden, 1932. v. 1, p. 356–366.
- 17 KAUFFMAN, S.A.; WEINBERGER, E.D. The NK model of rugged fitness landscapes and its application to maturation of the immune response. *Journal of Theoretical Biology*, v. 141, n. 2, p. 211–245, 1989.

- 18 LEVITAN, B.; KAUFFMAN, S.A. Adaptive walks with noisy fitness measurements. *Molecular Diversity*, v. 1, p. 53–68, 1995. DOI: 10.1007/BF01715809.
- 19 WELCH, J.J.; WAXMAN, D. The NK model and population genetics. *Journal of Theoretical Biology*, v. 234, n. 3, p. 329–340, 2005.
- 20 WILKE, C.O.; CAMPOS, P.R.A.; FONTANARI, J.F. Genealogical process on a correlated fitness landscape. *Journal of Experimental Zoology Part B: molecular and developmental evolution*, v. 294, n. 3, p. 274–284, 2002.
- 21 MACKEN, C.A.; PERELSON, A.S. Protein evolution on rugged landscapes. *Proceedings of the National Academy of Sciences*, v. 86, n. 16, p. 6191–6195, 1989.
- 22 CARNEIRO, M.; HARTL, D.L. Adaptive landscapes and protein evolution. *Proceedings of the National Academy of Sciences*, v. 107, Suppl. 1, p. 1747–1751, 2010.
- 23 REIA, S. M.; CAMPOS, P. R. A. Analysis of statistical correlations between properties of adaptive walks in fitness landscapes. *Royal Society Open Science*, v. 7, n. 1, p. 192118, 2020.
- 24 LEVINTHAL, D.A. Adaptation on rugged landscapes. *Management Science*, v. 43, n. 7, p. 934–950, 1997.
- 25 LAZER, D.; FRIEDMAN, A. The network structure of exploration and exploitation. *Administrative Science Quarterly*, v. 52, n. 4, p. 667–694, 2007.
- 26 BILLINGER, S.; STIEGLITZ, N.; SCHUMACHER, T.R. Search on rugged landscapes: an experimental study. *Organization Science*, v. 25, n. 1, p. 93–108, 2013.
- 27 REIA, S.M.; HERRMANN, S.; FONTANARI, J.F. Impact of centrality on cooperative processes. *Physical Review E*, v. 95, n. 2, p. 022305, 2017.
- 28 FONTANARI, J. F. Exploring NK fitness landscapes using imitative learning. *European Physical Journal B*, v. 4, p. 251, 2015. DOI: 10.1140/epjb/e2015-60608-1.
- 29 HWANG, S.; SCHMIEGELT, B.; FERRETTI, L.; KRUG, J. Universality classes of interaction structures for NK fitness landscapes. *Journal of Statistical Physics*, v. 172, p. 226–278, 2018. DOI: 10.1007/s10955-018-1979-z.
- 30 WRIGHT, A.H.; THOMPSON, R.K.; ZHANG, J. The computational complexity of NK fitness functions. *IEEE Transactions on Evolutionary Computation*, v. 4, n. 4, p. 373–379, 2000.
- 31 WASSERMAN, L.A. *All of statistics: a concise course in statistical inference*. New York: Springer, 2004.
- 32 DERRIDA, B. Random-energy model: an exactly solvable model of disordered systems. *Physical Review B*, v. 24, n. 5, p. 2613–2626, 1981.
- 33 BOYD, R.; RICHERSON, P.J. *The origin and evolution of cultures*. Oxford: Oxford University Press, 2005.
- 34 CROW, J. F.; KIMURA, M. *An introduction to population genetics theory*. New York: Harper and Row, 1970.

-
- 35 BAAKE, E.; BAAKE, M.; WAGNER, H. Ising quantum chain is equivalent to a model for biological evolution. *Physical Review Letters*, v. 78, n. 3, p. 559–562, 1997.
  - 36 BAAKE, E.; WAGNER, H. Mutation-selection models solved exactly with methods of statistical mechanics. *Genetics Research*, v. 78, n. 1, p. 93–117, 2001.
  - 37 SAAKIAN, D.B.; FONTANARI, J.F. Evolutionary dynamics on rugged fitness landscapes: exact dynamics and information theoretical aspects. *Physical Review E*, v. 80, n. 4, p. 041903, 2009.
  - 38 FELLER, W. *An introduction to probability theory and its applications*. 3rd ed. New York: Wiley, 1968.
  - 39 REICHL, L.E. *A modern course in statistical physics*. London: Arnold, 1980.
  - 40 RIDLEY, M. *Evolution*. Oxford: Oxford University Press, 2003.
  - 41 SANTOS, M.; SZATHMÁRY, E.; FONTANARI, J.F. Phenotypic plasticity, the baldwin effect, and the speeding up of evolution: the computational roots of an illusion. *Journal of Theoretical Biology*, v. 371, p. 127–136, 2015. DOI: 10.1016/j.jtbi.2015.02.012.
  - 42 FONTANARI, J.F. Social interaction as a heuristic for combinatorial optimization problems. *Physical Review E*, v. 82, n. 5 Pt 2, p. 056118, 2010.
  - 43 CASTELLANO, C.; FORTUNATO, S.; LORETO, V. Statistical physics of social dynamics. *Review of Modern Physics*, v. 81, n. 2, p. 591, 2009.
  - 44 GALAM, S. *Sociophysics*. New York: Springer, 2012.
  - 45 HUBERMAN, B.A. The performance of cooperative processes. *Physica D: nonlinear phenomena*, v. 42, n. 1–3, p. 38–47, 1990.
  - 46 CLEARWATER, S.H.; HUBERMAN, B.A.; HOGG, T. Cooperative solution of constraint satisfaction problems. *Science*, v. 254, n. 5035, p. 1181–1183, 1991.
  - 47 RENDELL, L.; BOYD, R.; COWNDEN, D.; ENQUIST, M.; ERIKSSON, K.; FELDMAN, M.W.; FOGARTY, L.; GHIRLANDA, S.; LILLICRAP, T.; LALAND, K.N. Why copy others? insights from the social learning strategies tournament. *Science*, v. 328, n. 5975, p. 208–213, 2010.
  - 48 BLACKMORE, S. *The meme machine*. Oxford: Oxford University Press, 2000.
  - 49 SHIBANAI, Y.; YASUNO, S.; ISHIGURO, I. Effects of global information feedback on diversity. *Journal of Conflict Resolution*, v. 45, n. 1, p. 80–96, 2001.
  - 50 PERES, L.R.; FONTANARI, J.F. The media effect in axelrod’s model explained. *Europhysics Letters*, v. 96, n. 3, p. 38004, 2011.
  - 51 AXELROD, R. The dissemination of culture: a model with local convergence and global polarization. *Journal of Conflict Resolution*, v. 41, n. 2, p. 203–226, 1997.
  - 52 JANIS, I.L. *Groupthink: psychological studies of policy decisions and fiascoes*. Boston: Houghton Mifflin, 1982.

- 53 FONTANARI, J.F. When more of the same is better. *Europhysics Letters*, v. 113, n. 2, p. 28009, 2016.
- 54 FONTANARI, J.F.; RODRIGUES, F.A. Influence of network topology on cooperative problem-solving systems. *Theory in Biosciences*, v. 135, n. 3, p. 101–110, 2016.
- 55 GOMES, P.F.; REIA, S.M.; RODRIGUES, F.A.; FONTANARI, J.F. Mobility helps problem-solving systems to avoid groupthink. *Physical Review E*, v. 99, n. 3, p. 032301, 2019.
- 56 LALAND, K. N.; WILLIAMS, K. Social transmission of maladaptive information in the guppy. *Behavioral Ecology*, v. 9, n. 5, p. 493–499, 1998.
- 57 REIA, S. M.; AQUINO, L. F.; FONTANARI, J. F. The surprising little effectiveness of cooperative algorithms in parallel problem solving. *European Physical Journal B*, v. 93, n. 7, p. 140, 2020.