




C A P Í T U L O 8

PLANEJAMENTO OTIMIZADO DE PROJETOS RESIDENCIAIS: UMA ABORDAGEM COMPUTACIONAL BASEADA EM DADOS

 <https://doi.org/10.22533/at.ed.185112624028>

Wiley Brito Almondes Lima

Universidade de São Paulo. Mestre em Engenharia Elétrica.
Skjutbanegatan 3D; Västerås, Västmanland, Suécia

Solange Pereira Dos Santos Farah

FATEC Sertãozinho. Professora Mestre Associada
Sertãozinho, São Paulo, Brasil

RESUMO: A eficiência no planejamento é essencial para o sucesso de obras residenciais, que exigem equilíbrio entre custo, prazo e qualidade. Este trabalho propõe a aplicação de Algoritmos Genéticos (AGs) como ferramenta de otimização no planejamento de obras civis residenciais, com foco na redução de custos e prazos, além da melhoria na alocação de recursos. A metodologia foi validada por meio de um estudo de caso real, envolvendo uma residência térrea padrão R1-N (residência unifamiliar padrão normal) situada na região Norte do Brasil. O modelo computacional desenvolvido simulou 20 atividades da obra, considerando variáveis como produtividade, complexidade, condições climáticas e restrições operacionais. O AG foi implementado em linguagem Python, utilizando uma função de avaliação que integra critérios de custo, tempo e satisfação do cliente. Os resultados obtidos demonstraram a capacidade do algoritmo em encontrar soluções eficientes, respeitando as restrições estabelecidas e mantendo alto nível de desempenho. A abordagem proposta mostrou-se promissora como apoio à tomada de decisão no setor da construção civil, com potencial de adaptação para projetos mais complexos e integração com tecnologias emergentes.

PALAVRAS-CHAVE: algoritmos; otimização; construção; planejamento; eficiência.

Optimized Planning of Residential Projects: A Data-Driven Computational Approach

ABSTRACT: Efficiency in planning is essential for the success of residential construction projects, which require a balance between cost, time, and quality. This work proposes the application of Genetic Algorithms (GAs) as an optimization tool in the planning of residential civil works, focusing on the reduction of costs and deadlines, as well as the improvement in resource allocation. The methodology was validated through a real case study involving a standard single-story R1-N residence (standard normal single-family residence) located in the northern region of Brazil. The developed computational model simulated 20 construction activities, considering variables such as productivity, complexity, weather conditions, and operational constraints. The GA was implemented in Python language, using an evaluation function that integrates cost, time, and customer satisfaction criteria. The results obtained demonstrated the algorithm's ability to find efficient solutions, respecting the established constraints and maintaining a high level of performance. The proposed approach proved to be promising as support for decision-making in the civil construction sector, with the potential for adaptation to more complex projects and integration with emerging technologies.

KEYWORDS: algorithms; optimization; construction; planning; efficiency.

INTRODUÇÃO

No cenário da construção civil, o planejamento eficiente é crucial para o sucesso de qualquer empreendimento, especialmente quando se trata de obras residenciais. Essas obras, caracterizadas por exigências específicas de qualidade, design e prazos, demandam uma abordagem minuciosa para garantir a realização de projetos que atendam às expectativas dos clientes e estejam alinhados com as normas e regulamentações vigentes.

A falta de planejamento e de ferramentas computacionais tem gerado ineficiências significativas no setor. Segundo levantamento da Deloitte, encomendado pela FIESP, o setor pode perder até R\$ 59,1 bilhões entre 2023 e 2025 devido a atrasos. Esse cenário reforça a urgência da adoção de métodos mais avançados para otimizar cronogramas e apoiar a tomada de decisão no setor (Imobi Report, 2023).

Neste contexto, a utilização de técnicas de otimização torna-se uma estratégia valiosa para enfrentar os desafios complexos associados ao planejamento de obras civis residenciais. Os algoritmos genéticos despontam como uma ferramenta poderosa para resolver problemas de otimização complexos (Sivanandam; Deepa, 2008), tais como os enfrentados no planejamento de obras civis. Inspirados no processo de

seleção natural e na teoria da evolução, os AG são métodos de busca e otimização baseados em princípios biológicos, que imitam o processo de evolução natural para encontrar soluções eficientes para problemas complexos (Holland, 1992).

Esses algoritmos operam com uma população de soluções candidatas — os “indivíduos” — que evoluem por meio de operadores genéticos como seleção, cruzamento e mutação (Gen e Cheng, 2000; Chavali, Pahwa e Das, 2002). Através de ciclos iterativos, os AGs promovem a combinação e a variação das soluções, conduzindo progressivamente a melhores resultados, mesmo em contextos com alta variabilidade de dados e múltiplas restrições.

Nos últimos anos, diversos estudos vêm reforçando o potencial dos AGs em projetos de engenharia civil. Srimathi et al. (2023) propuseram um modelo automatizado de otimização de cronogramas, eliminando conflitos entre atividades e melhorando a produtividade Zu e Liu (2024), por sua vez, demonstraram reduções significativas de prazo e custo em empreendimentos habitacionais com o uso de algoritmos genéticos. Tais aplicações práticas comprovam a robustez e a atualidade da abordagem. Nazari e Yan (2025) evidenciaram a superioridade dos AGs em fases iniciais de projeto arquitetônico, ao compará-los com métodos de busca aleatória e em grade, destacando sua capacidade de gerar soluções mais precisas e com menor custo computacional.

Além de sua capacidade de lidar com a complexidade e a não linearidade das variáveis envolvidas, os AGs são altamente adaptativos, sendo capazes de ajustar-se dinamicamente a variações no mercado e nas condições operacionais. Essas características tornam os AGs particularmente adequados para o setor da construção civil, que frequentemente lida com ambientes incertos e decisões multidimensionais.

Dentro desse contexto, este trabalho teve como objetivo contribuir para o aumento da eficiência operacional e a tomada de decisão, baseada em dados, com a aplicação de AGs no planejamento de obras civis residenciais. Os objetivos específicos foram: (i) desenvolver um modelo computacional em Python para simular o planejamento de uma obra residencial, considerando dados reais da construção civil; (ii) aplicar um algoritmo genético para otimizar o custo total da obra, o tempo de execução e a alocação de recursos; (iii) validar a metodologia por meio de um estudo de caso real, utilizando uma residência térrea padrão R1-N, com 20 atividades simuladas; (iv) analisar os resultados da otimização com base em indicadores de desempenho – custo, tempo e satisfação do cliente – e por meio de representações gráficas.

MATERIAL E MÉTODOS

A pesquisa é classificada como aplicada, de natureza quantitativa, com caráter exploratório-descritivo, conduzida sob a forma de estudo de caso em uma residência térrea padrão R1-N.

Esta seção apresenta os dados utilizados e os procedimentos metodológicos adotados para o desenvolvimento da pesquisa. O objetivo principal foi avaliar a eficácia da aplicação de algoritmos genéticos como ferramenta de otimização no planejamento de obras civis residenciais, com foco na redução de custos, prazos e na melhor alocação de recursos.

ESTUDO DE CASO: RESIDÊNCIA TÉRREA PADRÃO R1-N

A validação do modelo proposto foi realizada por meio de um estudo de caso aplicado a um projeto real de uma incorporadora localizada no estado do Pará (Figura 1). O empreendimento analisado corresponde a uma residência unifamiliar térrea, classificada como padrão R1-N (normal), conforme os critérios do Sinduscon e da ABNT (2025).

As principais características do projeto são descritas a seguir:

- ▮ Tipo de edificação: residência térrea unifamiliar;
- ▮ Número de pavimentos: 1;
- ▮ **Área** construída total: 116,24 m²;
- ▮ **Área** do terreno: 199,81 m²;
- ▮ Técnica construtiva: estrutura de concreto com blocos de vedação;
- ▮ Cômodos: 2 quartos, 1 suíte, 1 sala, cozinha americana, 2 vagas de garagem cobertas.



Figura 1. Imagem ilustrativa da residência utilizada no estudo de caso

Fonte: Dados originais da pesquisa

Para a estimativa de custos, utilizou-se como base o Custo Unitário Básico (CUB) da construção civil na região Norte, divulgado em março de 2025, no valor de R\$ 2.350,00/m², conforme dados do Sindicato da Indústria da Construção Civil do Estado do Pará – Sinduscon-PA (2025). Considerando que esse valor não contempla despesas adicionais, como projetos complementares, fiscalização, ligações públicas e mobiliário, foi aplicado um coeficiente de ajuste de 1,15. Como resultado, chegou-se a um Custo Unitário da Construção (CUC) de R\$ 2.702,50/m². Com base na área construída (116,25 m²), o custo global estimado da obra foi utilizado como base comparativa frente aos resultados obtidos pela aplicação do AG.

COLETA DE DADOS

A coleta de dados englobou informações técnicas e operacionais necessárias para a construção do modelo computacional. Os principais elementos considerados foram:

Requisitos de construção: especificações técnicas, diretrizes de projeto e exigências normativas aplicáveis;

Cronogramas: fases da obra com prazos estimados, organizadas conforme o cronograma de execução;

Orçamentos: estimativas detalhadas de custos por atividade, incluindo materiais, mão de obra e equipamentos;

Restrições regulatórias: normas urbanísticas e legislações vigentes em âmbito municipal, estadual e federal.

DADOS HISTÓRICOS E OPERACIONAIS

Para assegurar maior realismo ao modelo, foram utilizados dados operacionais oriundos de obras anteriores realizadas pela incorporadora. Esses dados possibilitaram a calibração precisa do algoritmo, refletindo o comportamento real de variáveis importantes do planejamento. Os principais dados incorporados incluem:

Histórico de desempenho: registros de prazo e custo efetivo em obras passadas;

Custos por atividade: valores médios praticados em serviços de construção civil na região;

Produtividade da equipe: indicadores operacionais de rendimento por frente de trabalho;

Consumo e desperdício de materiais: padrões médios observados na execução de projetos residenciais similares.

ALGORITMO GENÉTICO

O Algoritmo Genético (AG) é uma técnica de otimização inspirada nos mecanismos da evolução natural, como seleção, reprodução e mutação (Goldberg, 1989). Desenvolvido por John Holland na década de 1960, o AG é utilizado para encontrar soluções aproximadas em problemas complexos, principalmente quando a solução exata é inviável devido à elevada quantidade de variáveis e restrições envolvidas (Mitchell, 1998; Yang, 2010).

Estudos recentes reforçam a aplicabilidade dos AGs no setor da construção civil. Al-Douri e Al-Zwainy (2024) propuseram um modelo de agendamento sustentável para projetos de construção, utilizando AGs com o objetivo de reduzir o tempo total de execução e, simultaneamente, manter os índices de emissão de poluentes dentro dos padrões estabelecidos por normas ambientais. A aplicação bem-sucedida da técnica evidencia sua eficácia em contextos reais e destaca seu potencial como ferramenta de apoio à tomada de decisão em ambientes complexos e multidimensionais.

ESTRUTURA GERAL DO ALGORITMO

O fluxograma do AG é apresentado na Figura 2. O processo inicia com a definição da função objetivo, variáveis de controle, representação das soluções e espaço de busca. Em seguida, gera-se uma população inicial aleatória, avaliada por função fitness. Se a solução atende aos critérios, o processo é encerrado; caso contrário, executa-se o ciclo de seleção, cruzamento e mutação até que o critério de parada seja alcançado (Radosavljevic, 2018).

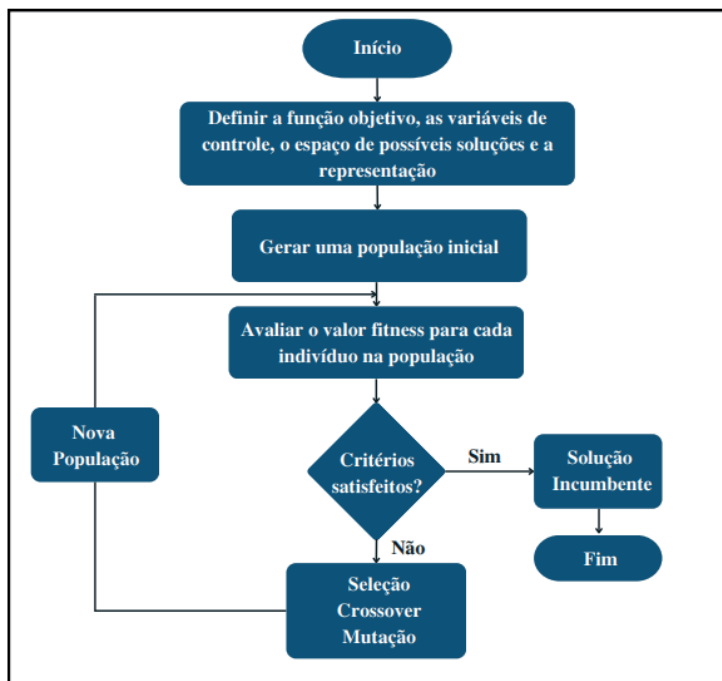


Figura 2. Fluxograma do algoritmo genético simples

Fonte: Dados originais da pesquisa

POPULAÇÃO INICIAL

A população inicial utilizada neste trabalho foi composta por 100 indivíduos, cada um representando uma solução candidata para o planejamento de uma obra residencial. Cada indivíduo foi codificado como um vetor com 20 genes, correspondentes às principais atividades da construção civil.

Os valores dos genes foram gerados de forma aleatória dentro do intervalo contínuo de 0,5 a 1,5. Essa faixa foi definida com base em variações plausíveis observadas em campo quanto ao tempo e custo de execução de cada atividade, refletindo cenários reais de produtividade, eficiência e restrições operacionais no canteiro de obras.

A diversidade genética dessa população inicial foi um elemento crítico para o sucesso do AG, pois permitiu explorar amplamente o espaço de busca, com o intuito de obter soluções mais robustas e inovadoras. Essa variabilidade contribuiu para evitar a convergência prematura a ótimos locais, ampliando as chances de encontrar resultados, ainda mais, otimizados em termos de custo, prazo e satisfação do cliente.

CODIFICAÇÃO

A codificação adotada é do tipo vetorial com números reais, apropriada para problemas com variáveis contínuas. Cada gene do vetor representa um fator de ajuste aplicado a uma atividade da obra, como alterações em produtividade, custo ou tempo de execução. Essa abordagem permite uma representação mais precisa das soluções, facilitando a aplicação dos operadores genéticos — como cruzamento e mutação — e oferecendo maior flexibilidade para simular diferentes cenários de alocação de recursos.

Além disso, a codificação contínua favorece o ajuste fino das soluções, sendo especialmente vantajosa em ambientes com alta variabilidade operacional. Isso é crucial quando se busca otimizar simultaneamente a utilização de mão de obra, materiais e prazos de execução das etapas da obra, contribuindo para um planejamento mais realista e eficiente (Chuang; Chen; Hwang, 2016).

FUNÇÃO FITNESS

A avaliação das soluções foi realizada por meio de uma função fitness composta por três componentes principais: penalização por violação de restrições, satisfação do cliente e tempo total de execução. A função fitness (F) foi calculada conforme a equação (1).

$$F = w_1 \cdot (1 - P') + w_2(T_{ref} - T) + w_3 \cdot S \quad (1)$$

onde, P valor da penalização; T : tempo total de execução da obra; T_{ref} : tempo de referência (limite superior); S : nível de satisfação do cliente (entre 0 e 100); w_1 , w_2 , w_3 : pesos atribuídos a cada critério (custo, tempo e satisfação).

PENALIZAÇÃO (P)

A penalização total é calculada com base na violação de limites máximos e mínimos para custo e tempo. A equação geral, apresentada de maneira simplificada, é expressa na eq. (2).

$$P' = \begin{cases} \alpha(C - C_{máx}), & \text{se } C > C_{máx} \\ \beta(C_{mín} - C), & \text{se } C < C_{mín} \\ \gamma(T - T_{máx}), & \text{se } T > T_{máx} \\ 0, & \text{caso contrário} \end{cases} \quad (2)$$

onde, C : custo total da obra; T : tempo total de execução; $C_{mín}$, $C_{máx}$, $T_{máx}$: limites definidos; α, β, γ coeficientes de penalização ajustados conforme a importância de cada critério.

SATISFAÇÃO DO CLIENTE (S)

A satisfação do cliente foi modelada como uma função que parte de um valor ideal (100%) e sofre reduções proporcionais sempre que o custo ou o tempo extrapolam os limites estabelecidos. Assim, quanto mais próxima a solução estiver dos parâmetros considerados ideais, maior será o valor de satisfação conforme a eq. (3):

$$S = 100 - f(C, T) \quad (3)$$

onde, $f(C,T)$: representa uma função decrescente que penaliza a satisfação à medida que os custos ou prazos se afastam dos limites estabelecidos, refletindo a percepção do cliente em relação ao desempenho da obra.

Essa formulação permitiu balancear os três objetivos principais do modelo: respeitar as restrições orçamentárias e temporais, maximizar a satisfação do cliente e promover a eficiência global do planejamento.

SELEÇÃO

A seleção dos indivíduos que participam da próxima geração foi realizada com base no método de torneio, estratégia amplamente validada em problemas de otimização combinatória (Vélez, 2015). A cada geração, os cinco indivíduos com melhor desempenho (maior valor de fitness) foram automaticamente preservados para compor a nova população - mecanismo conhecido como elitismo.

O restante dos indivíduos foi selecionado por meio de torneios aleatórios com três participantes, nos quais o mais apto era escolhido. Esse processo foi repetido até preencher o restante da população, promovendo uma competição controlada entre candidatos e incentivando a diversidade.

A combinação de elitismo e torneio proporciona um equilíbrio eficiente entre exploração (busca por novas soluções no espaço de busca) e intensificação (aperfeiçoamento das soluções mais promissoras). Esse equilíbrio é fundamental para evitar a estagnação em ótimos locais e garantir que o algoritmo continue evoluindo de forma eficaz ao longo das gerações (Vélez, 2015).

CROSSOVER

A Figura 3 ilustra o funcionamento do crossover uniforme, técnica adotada neste trabalho para gerar novos indivíduos a partir da combinação de dois pais. Esse operador atua gene a gene, utilizando uma sequência de decisão aleatória que define, para cada posição do vetor, de qual dos pais o gene será herdado (Kato, 2021).

Se o valor da sequência for 1, o gene correspondente foi herdado do Pai 1; se for 0, o gene foi herdado do Pai 2. Esse processo é repetido ao longo de todo o vetor, gerando dois filhos com combinações genéticas distintas.

Essa abordagem garante uma recombinação equilibrada dos genes dos pais, promovendo maior diversidade na população e permitindo que o algoritmo explore múltiplas regiões do espaço de busca.

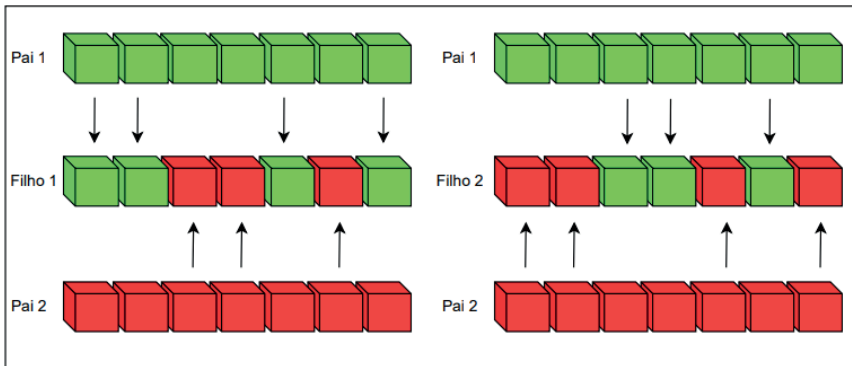


Figura 3. Ilustração do processo de crossover uniforme

Fonte: Dados originais da pesquisa

MUTAÇÃO

A mutação é um operador fundamental nos algoritmos genéticos, responsável por preservar a diversidade genética e explorar novas regiões do espaço de busca (Sivanandam; Deepa, 2008). Trata-se de um processo no qual valores aleatórios são aplicados aos genes de determinados indivíduos, modificando-os dentro de uma faixa predefinida (Kato, 2021). Essa variação pode representar, na prática, flutuações em produtividade, mudanças climáticas ou imprevistos na execução da obra.

Neste estudo, a mutação foi aplicada com uma taxa de 10% sobre os indivíduos da população. Para cada indivíduo selecionado, um de seus genes foi modificado por um fator aleatório entre 0,9 e 1,1, simulando pequenas variações na execução de cada atividade. Essa abordagem foi essencial para evitar a estagnação do algoritmo em soluções locais, permitindo que novas combinações fossem exploradas mesmo nas últimas gerações.

A Figura 4 ilustra o processo de mutação, no qual um gene específico sofre alteração, resultando em uma nova configuração genética.

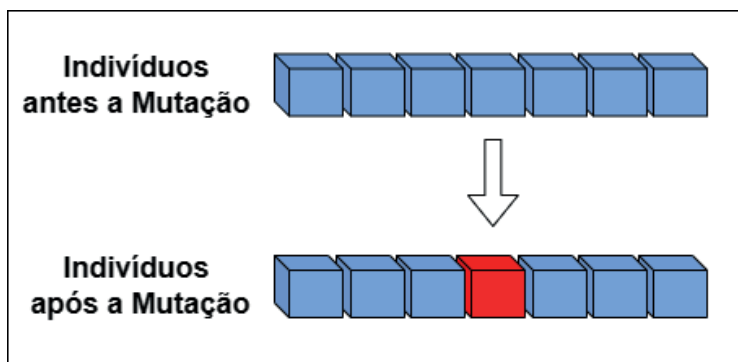


Figura 4. Representação esquemática da operação de mutação.

Fonte: Dados originais da pesquisa

Essa estratégia mostrou-se eficiente na prática, mantendo a diversidade genética necessária para a convergência do algoritmo e garantindo maior flexibilidade na identificação de soluções com melhor desempenho.

ELITISMO

Como comentado na seção de seleção, o elitismo foi incorporado ao algoritmo para garantir que os melhores indivíduos — aqueles com maior valor de fitness — fossem preservados a cada geração. Essa técnica impede a perda de boas soluções durante as operações de cruzamento ou mutação, assegurando que características vantajosas sejam mantidas ao longo do processo evolutivo (Radosavljevic, 2018).

Além de funcionar como um mecanismo de proteção das melhores soluções, geralmente, a aplicação do elitismo melhora o desempenho do AG (Chidanandappa; Ananthapadmanabha; Ranjith, 2016). Essa abordagem mostrou-se eficaz no contexto do planejamento de obras civis, onde múltiplos objetivos e restrições exigem soluções robustas e de alta qualidade.

CRITÉRIO DE PARADA

O critério de parada define quando o algoritmo deve encerrar seu ciclo evolutivo (Kato, 2021). Nesta pesquisa, foi adotado o critério de número máximo de gerações, com o valor de 100 iterações. Essa abordagem permite controlar o tempo de processamento e garante que o algoritmo execute ciclos suficientes para alcançar soluções satisfatórias sem desperdício de recursos computacionais.

VANTAGENS DOS ALGORITMOS GENÉTICOS

Os AGs oferecem várias vantagens que os tornam uma ferramenta eficiente em otimização:

Capacidade de encontrar soluções globais: os AGs são projetados para explorar amplamente o espaço de busca, evitando a armadilha de soluções locais e aumentando as chances de identificar a solução global ótima (Sivanandam; Deepa, 2008).

Aptidão para problemas multiobjetivo: Os AGs podem lidar simultaneamente com múltiplos objetivos e restrições, como minimizar custo, tempo e maximizar a satisfação, sem exigir linearidade ou continuidade nas funções envolvidas (Gen; Cheng, 2000).

Alta adaptabilidade: a estrutura flexível dos AGs permite sua adaptação a diferentes tipos de problemas e contextos (Sivanandam; Deepa, 2008). São especialmente úteis em ambientes dinâmicos e incertos, como o da construção civil, onde condições e requisitos mudam constantemente.

DESVANTAGENS DOS ALGORITMOS GENÉTICOS

Apesar das vantagens, os AGs também apresentam desafios que precisam ser considerados (Yang, 2010):

Sensibilidade a parâmetros de controle: o desempenho dos AGs depende fortemente da configuração adequada de parâmetros como tamanho da população, taxa de cruzamento, taxa de mutação e critério de parada. Parâmetros mal ajustados podem levar à convergência prematura ou a execuções ineficientes.

Custo computacional elevado: em problemas de alta complexidade, o tempo de execução e os recursos necessários para rodar múltiplas gerações podem ser significativos, exigindo maior capacidade computacional.

PARÂMETROS DE CONTROLE

Para alcançar bons resultados, é essencial o ajuste cuidadoso dos seguintes parâmetros (Kato, 2021):

- Tamanho da população: um número maior de indivíduos aumenta a diversidade e a chance de encontrar boas soluções, mas eleva o custo computacional.
- Taxa de cruzamento: define quantos indivíduos serão recombinados a cada geração.

- I Taxa de mutação: introduz variabilidade e ajuda a evitar estagnação. Esta taxa deve ser baixa para preservar boas soluções.
- I Número de gerações: define quantas vezes os operadores genéticos serão aplicados. Um valor adequado garante equilíbrio entre tempo de execução e qualidade da solução.

O ajuste desses parâmetros foi feito com base em simulações preliminares e referências da literatura (Radosavljevic, 2018), De Jong apud (Barcellos, 2000), (Susteras, 2006) garantindo uma aplicação robusta do algoritmo ao problema específico de planejamento de obras civis residenciais.

ALGORITMO 1: OTIMIZAÇÃO DO PLANEJAMENTO COM AG

A seguir, apresenta-se o pseudocódigo desenvolvido para a aplicação de um algoritmo genético voltado à otimização do planejamento de obras civis residenciais.

Dados de entrada: custos e tempos base das atividades, limites máximos de custo e tempo, e parâmetros de configuração do algoritmo.

Resultados esperados: vetor de ajuste ótimo, histórico de evolução da função fitness e avaliação final do projeto com base nos critérios de custo, tempo e satisfação do cliente.

início

Inicializa os parâmetros do AG: tamanho da população, número de gerações, taxa de mutação e número de elitistas;

Gera a população inicial com indivíduos contendo fatores aleatórios;

Para cada geração, faça:

Para cada indivíduo da população, faça:

Aplica os fatores de ajuste aos custos e tempos base;

Calcula custo_total e tempo_total;

Calcula a penalidade associada à solução:

Se custo_total > limite_custo, **então**

Adiciona penalidade proporcional ao excesso de custo;

Se tempo_total > limite_tempo, **então**

Adiciona penalidade proporcional ao excesso de tempo;

Se custo_total < limite_mínimo, **então**

Adiciona penalidade por economia excessiva;

Calcula satisfacao_cliente com base no custo e tempo;

Calcula fitness combinando penalidade, tempo e satisfação;

Fim para

Seleciona os melhores indivíduos por elitismo;

Realiza torneio entre indivíduos para seleção dos pais;

Aplica cruzamento entre os pais para gerar filhos;

Para cada filho, faça:

Se ocorrer mutação (com probabilidade definida), **então**

Modifica aleatoriamente o valor de um gene;

Fim para

Atualiza a população com elitistas e filhos;

Registra o melhor valor de fitness da geração;

Fim para

Seleciona o melhor indivíduo da população final (solução ótima);

Aplica os fatores ótimos às atividades:

- Calcula os custos finais por atividade;
- Calcula os tempos finais por atividade;
- Avalia a satisfação final do cliente;
- Gera gráficos e exporta os resultados em planilhas e figuras;

fim do algoritmo

O código-fonte que implementa o pseudocódigo foi desenvolvido pelo autor em Python, com base em dados reais da obra, permitindo simular cenários residenciais e ajustar dinamicamente os parâmetros do algoritmo para análise de custo, tempo e satisfação do cliente, conforme apresentado no Apêndice A.

RESULTADOS

Para validar a aplicação dos algoritmos genéticos no planejamento de obras civis residenciais, realizou-se uma simulação computacional com base no estudo de caso da residência unifamiliar térrea padrão R1-N, situada no estado do Pará. O modelo foi desenvolvido em Python e incorporou dados técnicos e operacionais da construção civil, além de parâmetros compatíveis com a realidade do setor, conforme descrito na Tabela 1.

Tabela 1. Parâmetros de Entrada do Modelo de Otimização

Parâmetro	Valor
Área construída	116,24 m ²
CUB médio regional	R\$ 2.350,00/m ²
Coefficiente de ajuste	1,15
Custo unitário (CUC)	R\$ 2.702,50/m ²
Custo máximo permitido	R\$ 355.000,00
Tempo máximo permitido	240 dias (8 meses)
Tamanho da população	100 indivíduos
Número de gerações	100
Taxa de cruzamento	50%
Taxa de mutação	10%

Fonte: Resultados originais da pesquisa

A simulação envolveu 20 atividades-chave da obra, desde a preparação do canteiro até os acabamentos finais. O AG ajustou, individualmente, os fatores de escala de cada atividade, simulando diferentes cenários de produtividade, alocação de recursos e condições operacionais.

RESULTADOS OTIMIZADOS

A execução do algoritmo genético resultou em uma solução otimizada com melhorias expressivas nos critérios de custo e prazo, mantendo um nível aceitável de satisfação do cliente. A solução final, após 100 gerações, foi avaliada com base nos parâmetros estabelecidos na Tabela 1.

A Tabela 2 apresenta os principais indicadores da solução obtida pelo algoritmo em comparação com o planejamento tradicional. Os resultados demonstram que a utilização do algoritmo genético no planejamento da obra foi eficaz, proporcionando uma redução significativa de 35,1% no custo total, reduzindo o valor estimado de R\$ 314.138,60 para R\$ 203.939,54. Apesar do aumento no prazo de execução de 180 dias para 238,49 dias, o novo cronograma permaneceu dentro do limite máximo previamente estipulado de 240 dias, mantendo-se viável do ponto de vista gerencial.

Além do desempenho financeiro, verificou-se um avanço expressivo no nível de satisfação do cliente, que passou de uma estimativa inicial de 80% para 92,12%. Esse aumento reflete a efetividade do modelo proposto em equilibrar múltiplos objetivos simultâneos, como custo, prazo e qualidade percebida. A alocação mais eficiente dos recursos e o ajuste dinâmico das variáveis críticas da obra contribuíram diretamente para esse desempenho superior.

Tabela 2. Comparativo com Planejamento Tradicional

Critério	Planejamento Tradicional	Solução Otimizada (AG)
Tempo total estimado	180 dias	238,49 dias
Custo total estimado	R\$ 314.138,60	R\$ 203.939,54
Satisfação do cliente	80%	92,12%

Fonte: Resultados originais da pesquisa

Na sequência, a Tabela 3 apresenta os resultados detalhados por atividade, permitindo uma análise específica dos impactos da otimização em cada etapa da construção.

Tabela 3. Resultados da Otimização por Atividade: Custos, Prazos e Fatores de Ajuste Aplicados

Atividade	Custo Base (R\$)	Tempo Base (dias)	Fator de Ajuste	Custo Final (R\$)	Tempo Final (dias)
Barracão e Projetos	14000	18,72	0,47	8218,71	12,03
Infraestrutura	26000	22,88	0,41	13702,06	15,21
Supraestrutura	52000	20,07	0,62	33015,21	10,84
Paredes e Painéis	28000	18,62	0,42	12715,75	11,80
Esquadrias	25000	15,59	0,41	11485,94	7,70
Vidros e Plásticos	5000	21,15	0,34	2059,85	11,41
Coberturas	25000	15,94	0,63	16445,63	11,04
Impermeabilizações	7500	27,29	0,29	2251,05	10,50
Revestimentos Internos	28000	29,09	0,39	13692,85	17,81
Forros	5000	14,59	0,46	2569,29	8,72
Revestimentos Externos	15000	24,79	0,34	6423,08	13,61
Pinturas	15000	18,22	0,52	9899,37	15,83
Pisos	32000	19,20	0,60	21380,49	15,22
Acabamentos	4000	28,14	0,25	1219,56	11,72
Elétrica e Telefonia	15000	6,78	0,75	14591,93	7,08
Hidráulica	15000	7,18	0,73	10230,32	6,19
Esgoto e Águas Pluviais	16000	5,51	0,77	14665,10	5,36
Louças e Metais	7500	25,82	0,49	4488,83	21,98
Complementos	5000	24,45	0,49	2639,39	13,44
Outros Serviços	7500	26,75	0,29	2245,12	10,97

Fonte: Resultados originais da pesquisa

ANÁLISE GRÁFICA DOS RESULTADOS

A fim de complementar a análise quantitativa, esta seção apresenta os principais resultados gráficos da simulação, permitindo visualizar o comportamento do algoritmo genético ao longo das gerações, bem como os impactos da otimização em cada atividade da obra. Os gráficos foram construídos a partir dos dados finais obtidos no modelo, refletindo diretamente os efeitos da variação de parâmetros como custo, tempo, clima, complexidade e eficiência operacional.

EVOLUÇÃO DA FUNÇÃO FITNESS AO LONGO DAS GERAÇÕES

Conforme ilustrado na Figura 5, observa-se a evolução da melhor função fitness ao longo das gerações do algoritmo genético, evidenciando o comportamento típico de processos evolutivos computacionais. Nas gerações iniciais, verifica-se um crescimento acentuado, resultado da identificação e eliminação de soluções inviáveis, fortemente penalizadas por descumprirem restrições do problema. A partir da metade da execução, a curva passa a apresentar comportamento estável, sinalizando a convergência do algoritmo para soluções otimizadas, capazes de equilibrar simultaneamente os critérios de custo, prazo e satisfação do cliente.

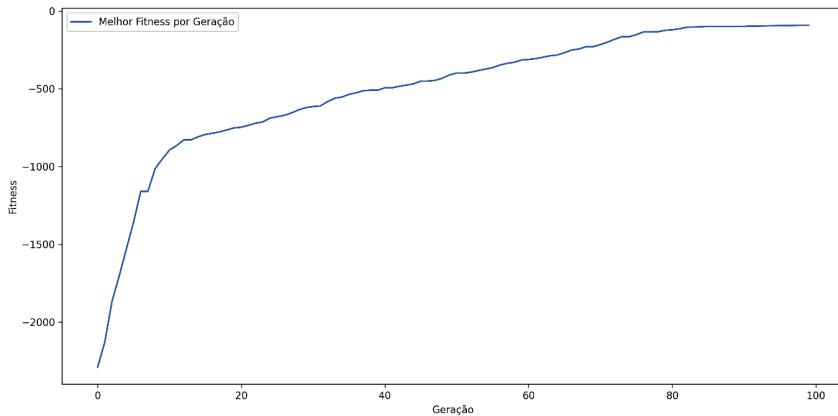


Figura 5. Evolução da Função Fitness ao Longo das Gerações

Fonte: Resultados originais da pesquisa

COMPARATIVO DE CUSTOS POR ATIVIDADE – ANTES E DEPOIS DA OTIMIZAÇÃO

A Figura 6 apresenta a comparação entre os custos base e os custos finais otimizados por atividade, evidenciando os impactos da aplicação do algoritmo genético na alocação orçamentária. Atividades como supraestrutura, pisos e coberturas mantiveram valores elevados, indicando a preservação da qualidade em fases estruturais essenciais.

Em contrapartida, atividades como vidros e plásticos, impermeabilizações, louças e metais, complementos e outros serviços apresentaram reduções significativas, revelando a eficácia do modelo em promover economia em itens de menor impacto técnico. Essa redistribuição estratégica permitiu concentrar investimentos em etapas mais críticas, contribuindo para um planejamento mais equilibrado e eficiente.

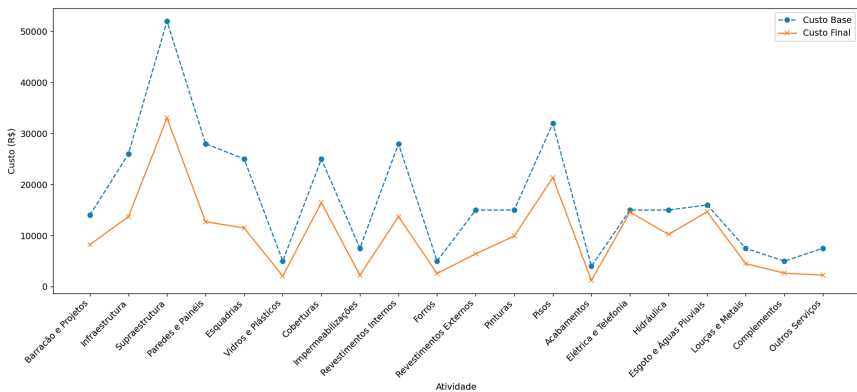


Figura 6. Comparativo de Custos por Atividade (Antes e Depois da Otimização)

Fonte: Resultados originais da pesquisa

EVOLUÇÃO ACUMULADA PLANEJADA DA OBRA

A Figura 7 apresenta a curva de avanço físico acumulado com base no planejamento tradicional da obra. Nota-se uma concentração significativa de atividades nas fases iniciais (1 a 3), evidenciada pela forte inclinação da curva nesse trecho. Esse comportamento reforça a necessidade de controle rigoroso no início do cronograma, período crítico para o desempenho geral do projeto.

Essa distribuição é coerente com os dados da Tabela 3, que demonstram maior tempo alocado em atividades como barracão e projetos, infraestrutura e supraestrutura. A comparação entre a curva e a tabela confirma que a otimização respeitou a lógica construtiva tradicional, mantendo a concentração inicial de esforços, mas com ajustes que tornaram o cronograma mais eficiente e tecnicamente equilibrado.

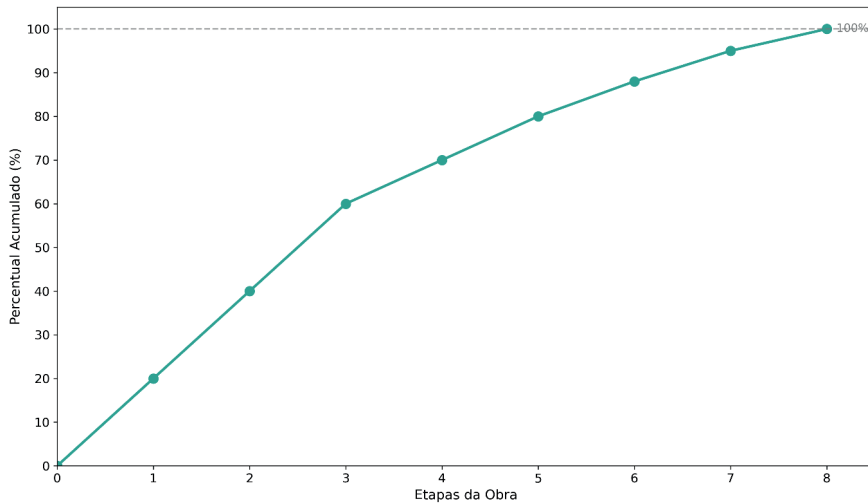


Figura 7. Evolução Acumulada Planejada da Obra

Fonte: Resultados originais da pesquisa

DISPERSÃO ENTRE CUSTO E TEMPO COM INDICAÇÃO DE SATISFAÇÃO

A Figura 8 apresenta a relação entre o custo final e o tempo de execução de cada atividade, com a coloração indicando o nível de satisfação individual. Observa-se que a maioria das atividades está concentrada na faixa entre R\$ 5.000 e R\$ 20.000 e 8 a 15 dias, sugerindo uma alocação eficiente dos recursos no planejamento.

Destacam-se as atividades de supraestrutura e pisos, que apresentaram custos mais elevados, mas mantiveram altos níveis de satisfação. Em contrapartida, tarefas como vidros e plásticos e impermeabilizações registraram baixo custo e tempo reduzido, mantendo, ainda assim, avaliações satisfatórias.

Os índices de satisfação individual variam entre 72% e 75%, enquanto a satisfação global atingiu 92,12%, como resultado da análise integrada do desempenho total do projeto. Mesmo com variações pontuais, o modelo demonstrou capacidade de equilibrar custo, tempo e qualidade, promovendo um planejamento final eficiente e bem avaliado pelo cliente.

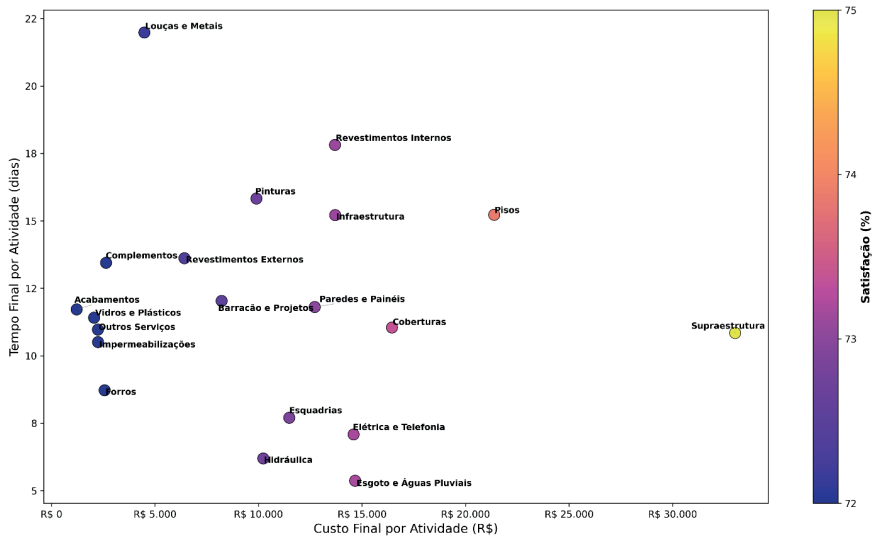


Figura 8. Dispersão entre Custo e Tempo com Indicação de Satisfação por Atividade

Fonte: Resultados originais da pesquisa

FATORES DE INFLUÊNCIA: CLIMA, COMPLEXIDADE E EFICIÊNCIA

Uma representação gráfica em escala de cores dos fatores de influência (clima, complexidade e eficiência) sobre cada atividade do cronograma é apresentada na Figura 9. Observa-se que infraestrutura, pinturas, pisos, revestimentos internos e louças e metais atingiram os maiores valores de complexidade (1,20), indicando elevado grau técnico e maior variabilidade operacional. No fator clima, destacam-se barracão e projetos (1,15), impermeabilizações (1,14) e infraestrutura (1,11), o que revela sensibilidade a condições ambientais. Em relação à eficiência, os menores índices foram observados em paredes e painéis, louças e metais, outros serviços, acabamentos e pinturas (todos com $\leq 0,84$), indicando riscos de retrabalho ou baixa produtividade. Essa análise permite identificar gargalos críticos e ajustar o planejamento com maior precisão.

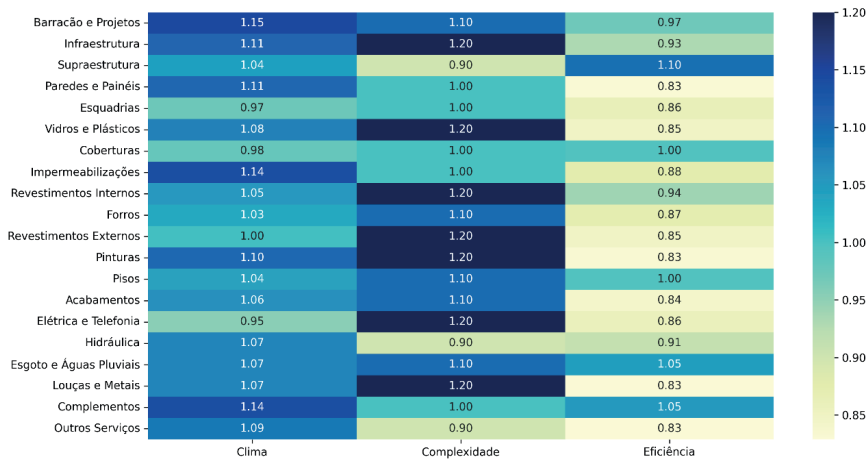


Figura 9. Fatores de Influência (Clima, Complexidade e Eficiência)

Fonte: Resultados originais da pesquisa

DISTRIBUIÇÃO PERCENTUAL DOS CUSTOS FINAIS POR ATIVIDADE

A Figura 10 apresenta a composição percentual dos custos finais por atividade, obtida após a aplicação do algoritmo genético ao planejamento de uma residência térrea padrão R1-N.

Os maiores percentuais de investimento concentram-se em supraestrutura (16,2%), pisos (10,5%), coberturas (8,1%), elétrica e telefonia (7,2%) e esgoto e águas pluviais (7,2%), refletindo a priorização de etapas com elevado impacto técnico e funcional. A supraestrutura absorve parcela significativa do orçamento devido à utilização de concreto armado e à necessidade de mão de obra qualificada. Os pisos e coberturas, por sua vez, abrangem áreas extensas da edificação, contribuindo para sua expressiva representatividade no custo total.

Na faixa intermediária de custo, destacam-se revestimentos internos (6,7%), infraestrutura (6,7%), paredes e painéis (6,2%), esquadrias (5,6%), hidráulica (5,0%), pinturas (4,9%) e barracão e projetos (4,0%), compondo etapas fundamentais para a execução e acabamento da obra.

As atividades de menor impacto orçamentário incluem revestimentos externos (3,1%), louças e metais (2,2%), complementos (1,3%), impermeabilizações (1,1%), outros serviços (1,1%), vidros e plásticos (1,0%) e acabamentos (0,6%). Esses percentuais reduzidos estão associados a itens de acabamento e serviços complementares, característicos de um padrão econômico de construção.

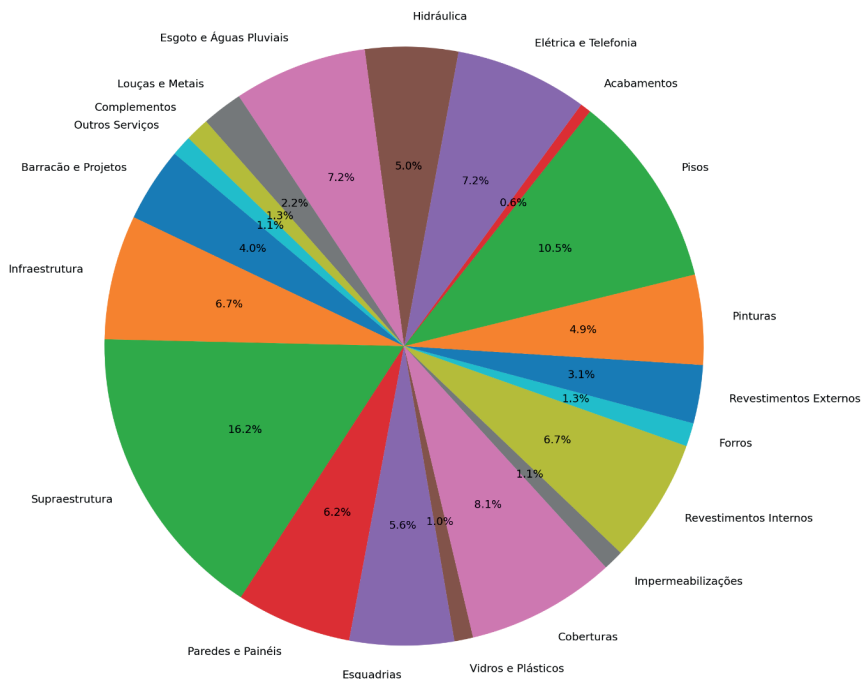


Figura 10. Distribuição Percentual dos Custos Finais por Atividade

Fonte: Resultados originais da pesquisa

TEMPOS FINAIS POR ATIVIDADE APÓS OTIMIZAÇÃO

O gráfico de barras (Figura 11) exibe o tempo final estimado para a execução de cada atividade da obra, após a aplicação do algoritmo genético. Os valores obtidos refletem a distribuição realista dos esforços no canteiro, considerando fatores externos e operacionais.

Destacam-se atividades como pinturas, revestimentos internos e louças e metais, que exigiram prazos mais longos devido ao detalhamento e à finalização dos ambientes. Por outro lado, elétrica, hidráulica e esgoto apresentaram tempos reduzidos, indicando maior eficiência operacional. A distribuição dos tempos reforça a coerência do modelo proposto com a realidade da construção civil regional.

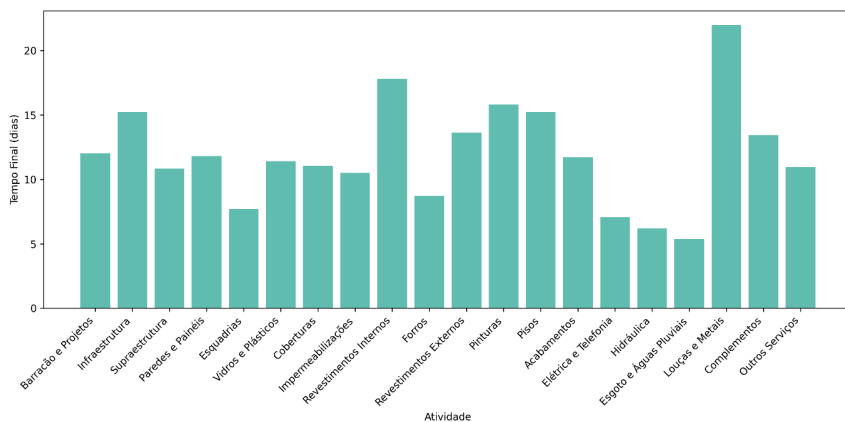


Figura 11. Tempos Finais por Atividade

Fonte: Resultados originais da pesquisa

Os resultados obtidos evidenciam o desempenho consistente do modelo baseado em algoritmos genéticos na otimização do planejamento de obras civis residenciais. A solução gerada apresentou uma significativa redução de 35,1% no custo total da obra, mantendo o prazo final dentro do limite estipulado de 240 dias. A satisfação do cliente também foi elevada, alcançando 92,12%.

A estrutura modular do modelo permitiu a incorporação de fatores externos — como clima, complexidade e eficiência das equipes —, ampliando o realismo das simulações. A análise gráfica reforçou tais resultados ao demonstrar a evolução da função *fitness*, a redistribuição de custos e o alinhamento com padrões técnicos do setor. Tais evidências comprovam o potencial técnico da abordagem adotada para cenários reais e restritivos.

CONSIDERAÇÕES FINAIS

Este trabalho demonstrou a viabilidade e a eficácia da aplicação de algoritmos genéticos no planejamento de obras civis residenciais, especialmente em cenários onde múltiplos objetivos — como redução de custos, controle de prazos e manutenção da satisfação do cliente — devem ser considerados de forma integrada.

A abordagem proposta resultou em uma solução otimizada e realista, capaz de atender aos critérios estratégicos de forma equilibrada. O modelo, implementado em Python, mostrou-se flexível e modular, permitindo personalização de parâmetros e integração com bases de dados específicas do setor da construção civil.

Além de seu potencial como ferramenta de apoio à decisão, o AG possibilitou uma análise gráfica e de sensibilidade aprofundada, contribuindo para a identificação de gargalos e oportunidades de melhoria nas diversas etapas da obra.

Como desdobramentos futuros, recomenda-se:

- Incorporar restrições adicionais de execução, como janelas climáticas, disponibilidade de equipes e condicionantes regionais;
- Incluir múltiplos critérios ponderados de satisfação;
- Integrar o modelo a plataformas BIM e bancos de dados reais da construção civil.

Em suma, os algoritmos genéticos se consolidam como uma alternativa eficiente e inovadora para melhorar a previsibilidade, a sustentabilidade e a racionalização de recursos na construção civil.

AGRADECIMENTO

Agradeço a Deus, à minha família, à orientadora, aos professores da Poli-USP e aos colegas que contribuíram para esta jornada. Registro, também, minha gratidão à Pós-Graduação da USP (POLI USP PRO), pelo suporte institucional e pela bolsa concedida.

REFERÊNCIAS

AL-DOURI, F. A.; AL-ZWAINY, F. M. **Eco-friendly scheduling model for construction projects utilizing genetic algorithms**. *Sustainability*, v. 16, n. 24, p. 11164, 2024. DOI: 10.3390/su162411164.

BARCELLOS, J. C. H. **Algoritmos Genéticos Adaptativos: Um estudo comparativo**. 143 p. Tese (Mestrado) — Escola Politécnica da Universidade de São Paulo, 2000. Citado 4 vezes nas páginas 47, 51, 53 e 56.

CHAVALI, S.; PAHWA, A.; DAS, S. **A genetic algorithm approach for optimal distribution feeder restoration during cold load pickup**. In: Proceedings of the 2002 Congress on Evolutionary Computation.

CHIDANANDAPPA, R.; ANANTHAPADMANABHA, T.; RANJITH, H. C. **Genetic algorithm based service restoration in distribution systems with multiple dgs for time varying loads**. In: 2016 Biennial International Conference on Power and Energy Systems: Towards Sustainable Energy (PESTSE).

CHUANG, Yao-Chen; CHEN, Chyi-Tsong; HWANG, Chyi. **A simple and efficient real-coded genetic algorithm for constrained optimization.** *Applied Soft Computing*, v. 38, p. 87–105, 2016.

GEN, M.; CHENG, R. 2000. **Genetic Algorithms and Engineering Optimization.** John Wiley & Sons, New York.

GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization, and Machine Learning.** United States of America: Jaddison-Wesley Publishing Company, 1989.

HOLLAND, J. H. 1992. **Adaptation in natural and artificial systems: an introductory biology, control, and artificial intelligence.** MIT Press/Bradford Books edition, London, England.

IMOBIREPORT. **Atrasos na construção civil brasileira implicam perdas de R\$ 59 bilhões.** 2023. Disponível em: <https://imobireport.com.br/incorporacao/atrasos-na-construcao-civil-brasileira-implicam-perdas-de-r-59-bilhoes>.

KATO, R. B. **Algoritmos Genéticos.** 2021. <https://bioinfo.com.br/algoritmos-geneticos/>.

MENDES, B. G. **Otimização da Localização de Poços de Petróleo com Completação Seca Utilizando Algoritmos Genéticos.** 106 p. Tese (Mestrado) – PUC-Rio, 2013.

MITCHELL, M. **An introduction to genetic algorithms.** London, England: Massachusetts Institute of Technology, 1998.

NAZARI, F.; YAN, W. **A case study on evaluating genetic algorithms for early building design optimization: comparison with random and grid searches,** 2025.

RADOSAVLJEVIC, J. **Metaheuristic Optimization in Power Engineering.** London, United Kingdom: The Institution of Engineering and Technology, 2018.

SINDICATO DA INDÚSTRIA DA CONSTRUÇÃO DO ESTADO DO PARÁ (SINDUSCON-PA). **Custo Unitário Básico da Construção – CUB.** Belém: Sinduscon-PA, 2025. Disponível em: <https://www.sindusconpa.org.br/cub>.

SIVANANDAM, S.; DEEPA, S. 2008. **Introduction to Genetic Algorithms.** Springer, Berlin Heidelberg.

SRIMATHI, K. R.; PADMAREKHA, A.; ANANDH, K. S. **Automated construction schedule optimization using genetic algorithm.** *Research Square*, 2023.

SUSTERAS, G. L. **Aplicação de algoritmos genéticos para previsão do comportamento das distribuidoras como apoio à estratégia de comercialização de energia de agentes geradores.** 101 p. Tese (Mestrado) – Escola Politécnica da Universidade de São Paulo, 2006.

VÉLEZ, Digna Isabel Arteaga. **Otimização de estruturas reticuladas utilizando algoritmos genéticos**. 2015. 97 f. Dissertação (Mestrado em Estruturas e Construção Civil) – Universidade de Brasília, Brasília, 2015.

YANG, X.-S. **Engineering Optimization: An Introduction with Metaheuristic Applications**. Hoboken, New Jersey: John Wiley & Sons, 2010.

ZU, B.; LIU, X. **Construction schedule optimization based on genetic algorithm**. In: INTERNATIONAL CONFERENCE ON ENGINEERING MANAGEMENT AND INDUSTRIAL SYSTEMS, 2024. Atlantis Press, 2024. p. 123–130.

APÊNDICE A - CÓDIGO-FONTE EM PYTHON

Neste apêndice apresenta-se o código em Python do Algoritmo Genético aplicado ao planejamento de obras residenciais, simulando vinte atividades de uma residência padrão R1-N.

```
import numpy as np
import pandas as pd
from random import seed
import matplotlib.pyplot as plt
import matplotlib.ticker as mticker
from adjustText import adjust_text
import seaborn as sns
from matplotlib.colors import Normalize
seed(0)
np.random.seed(0)
config = {
    "area_construida": 116.24,
    "cub": 2350.00,
    "fator_ajuste": 1.15,
    "tempo_maximo": 240,
    "custo_maximo": 355000.00,
    "custo_minimo": 314138.60 * 0.80}
```

```

nomes_atividades = [
    "Barracão e Projetos", "Infraestrutura", "Supraestrutura", "Paredes e Painéis",
    "Esquadrias", "Vidros e Plásticos", "Coberturas", "Impermeabilizações",
    "Revestimentos Internos", "Forros", "Revestimentos Externos", "Pinturas",
    "Pisos", "Acabamentos", "Elétrica e Telefonia", "Hidráulica",
    "Esgoto e Águas Pluviais", "Louças e Metais", "Complementos", "Outros Serviços"]

custos_base = np.array([ 14000.00, 26000.00, 52000.00, 28000.00, 25000.00,
5000.00,
    25000.00, 7500.00, 28000.00, 5000.00, 15000.00, 15000.00,
    32000.00, 4000.00, 15000.00, 15000.00, 16000.00, 7500.00,
    5000.00, 7500.00])

tempos_base = np.random.uniform(5, 30, size=20)
clima_fator = np.random.uniform(0.95, 1.15, size=20)
complexidade_atividade = np.random.choice([0.9, 1.0, 1.1, 1.2], size=20)
eficiência_equipes = np.random.uniform(0.8, 1.1, size=20)
custos_base_original = custos_base.copy()
tempos_base_original = tempos_base.copy()
custos_otimizados = custos_base * complexidade_atividade
riscos_custo = np.random.uniform(1.00, 1.15, size=len(custos_otimizados))
custos_otimizados *= riscos_custo

tempos_otimizados = tempos_base * clima_fator * complexidade_atividade /
eficiência_equipes
riscos_temporais = np.random.uniform(1.00, 1.20, size=len(tempos_otimizados))
tempos_otimizados *= riscos_temporais

```

```
def penalizacao(custo_total, tempo_total, limite_custo=config["custo_maximo"],
limite_tempo=config["tempo_maximo"]):
```

```
    penalidade = 0
```

```
    if custo_total > limite_custo:
```

```
        penalidade += (custo_total - limite_custo) * 0.01
```

```
    if tempo_total > limite_tempo:
```

```
        penalidade += (tempo_total - limite_tempo) * 100
```

```
    if custo_total < config["custo_minimo"]:
```

```
        penalidade += (config["custo_minimo"] - custo_total) * 0.02
```

```
    return penalidade
```

```
def satisfacao_cliente(custo_total, tempo_total):
```

```
    satisfacao = 100
```

```
    if custo_total > config["custo_maximo"]:
```

```
        satisfacao -= (custo_total - config["custo_maximo"]) * 0.00005
```

```
    if tempo_total > config["tempo_maximo"]:
```

```
        satisfacao -= (tempo_total - config["tempo_maximo"]) * 0.25
```

```
    if custo_total < config["custo_minimo"]:
```

```
        satisfacao -= (config["custo_minimo"] - custo_total) * 0.0001
```

```
    return max(0, min(100, satisfacao))
```

```
def criar_populacao(tamanho_populacao, num_variaveis):
```

```
    return np.random.uniform(low=0.5, high=1.5, size=(tamanho_populacao,
num_variaveis))
```

```
def calcular_fitness_multi(populacao, base_custos, base_tempos):
```

```
    fitness_scores = []
```

```
    for individuo in populacao:
```

```
        custos_estimados = base_custos * individuo
```

```

    tempos_estimados = base_tempos * individuo
    custo_total = np.sum(custos_estimados)
    tempo_total = np.sum(tempos_estimados)
    penal = penalizacao(custo_total, tempo_total)
    satisfacao = satisfacao_cliente(custo_total, tempo_total)
    fitness = (
        0.1 * (100 - penal) + # prioridade menor no custo
        0.7 * (config["tempo_maximo"] - tempo_total) + # tempo como fator
        0.6 * satisfacao)
    fitness_scores.append(fitness)
return np.array(fitness_scores)

def selecionar_individuos(populacao, fitness_scores, num_elitismo=5):
    indices_elitistas = np.argsort(fitness_scores)[-num_elitismo:]
    elitistas = populacao[indices_elitistas]
    pais = []
    torneio_k = 3
    for _ in range(len(populacao) - num_elitismo):
        participantes = np.random.choice(len(populacao), size=torneio_k,
replace=False)
        melhor_idx = participantes[np.argmax(fitness_scores[participantes])]
        pais.append(populacao[melhor_idx])
    return elitistas, np.array(pais)

def cruzamento(pais, num_filhos):
    filhos = []
    for _ in range(num_filhos // 2):
        pai1, pai2 = pais[np.random.randint(len(pais))], pais[np.random.
randint(len(pais))]

```

```

mask = np.random.rand(len(pai1)) < 0.5
filho1 = np.where(mask, pai1, pai2)
filho2 = np.where(mask, pai2, pai1)
filhos.extend([filho1, filho2])
return np.array(filhos)

```

```

def mutacao(populacao, taxa_mutacao=0.1):
    for individuo in populacao:
        if np.random.rand() < taxa_mutacao:
            idx = np.random.randint(len(individuo))
            individuo[idx] *= np.random.uniform(0.9, 1.1)
    return populacao

```

```

def algoritmo_genetico(base_custos, base_tempos, num_geracoes=100,
tamanho_populacao=100, num_elitismo=5):
    num_variaveis = len(base_custos)
    populacao = criar_populacao(tamanho_populacao, num_variaveis)
    historico = []
    for _ in range(num_geracoes):
        fitness = calcular_fitness_multi(populacao, base_custos, base_tempos)
        melhor_fitness = np.max(fitness)
        historico.append(melhor_fitness)
        elitistas, pais = selecionar_individuos(populacao, fitness, num_elitismo)
        filhos = cruzamento(pais, tamanho_populacao - num_elitismo)
        populacao = np.vstack((elitistas, filhos))
        populacao = mutacao(populacao)
    melhor_indice = np.argmax(fitness)
    melhor_solucao = populacao[melhor_indice]

```

```

return melhor_solucao, historico

melhor_solucao, historico = algoritmo_genetico(custos_otimizados, tempos_
otimizados)

custos_finais = custos_otimizados * melhor_solucao
tempos_finais = tempos_otimizados * melhor_solucao
custo_total = np.sum(custos_finais)
tempo_total = np.sum(tempos_finais)
satisfacao_final = satisfacao_cliente(custo_total, tempo_total)

tabela_resultado = pd.DataFrame({
    'Atividade': nomes_atividades,
    'Custo Base (R$)': custos_base,
    'Tempo Base (dias)': tempos_base,
    'Fator de Ajuste': melhor_solucao,
    'Custo Final (R$)': custos_finais,
    'Tempo Final (dias)': tempos_finais,
})

custo_total_com_bdi = 355000.00
print(tabela_resultado)
print("\nCusto Total Final: R$", round(custo_total, 2))
print("Tempo Total Final:", round(tempo_total, 2), "dias")
print("Satisfação Estimada do Cliente:", round(satisfacao_final, 2), "%")
print("\n--- Comparativo com Planejamento Convencional ---")
print(f"Custo estimado tradicional (com BDI): R$ {custo_total_com_bdi}")
print("Tempo estimado tradicional: 180 dias")
print("Custo estimado tradicional: R$ 314.138,60")

```

```

print("Satisfação estimada tradicional: 80%")
print(f"Tempo otimizado: {round(tempo_total, 2)} dias")
print(f"Custo otimizado: R$ {round(custo_total, 2)}")
print(f"Satisfação otimizada: {round(satisfacao_final, 2)}%")

```

```

plt.figure(figsize=(14, 6))

plt.plot(nomes_atividades, custos_base, label='Custo Base', linestyle='--',
marker='o')

plt.plot(nomes_atividades, custos_finais, label='Custo Final', linestyle='-',
marker='x')

plt.xlabel('Atividade', fontsize=11)
plt.ylabel('Custo (R$)', fontsize=11)
plt.xticks(rotation=45, ha='right', fontsize=11)
plt.yticks(fontsize=11)
plt.legend(fontsize=11)
plt.grid(False)
plt.tight_layout()

plt.savefig('comparativo_custos_atividades.png', dpi=300)
plt.show()

```

```

plt.figure(figsize=(12, 6))
sns.heatmap(
    pd.DataFrame({
        'Clima': clima_fator,
        'Complexidade': complexidade_atividade,
        'Eficiência': eficiencia_equipes
    }, index=nomes_atividades),
    annot=True, cmap='YlGnBu', fmt='.2f')
plt.tight_layout()

```

```
plt.savefig('fatores_atividade_heatmap.png', dpi=300)
plt.show()
```

```
plt.figure(figsize=(12, 6))
plt.plot(historico, label='Melhor Fitness por Geração', color='blue')
plt.xlabel('Geração')
plt.ylabel('Fitness')
plt.grid(False)
plt.legend()
plt.tight_layout()
plt.savefig('evolucao_fitness.png', dpi=300)
plt.show()
```

```
plt.figure(figsize=(12, 6))
plt.bar(
    tabela_resultado['Atividade'],
    tabela_resultado['Tempo Final (dias)'],
    color='#2A9D8F',
    alpha=0.7
)
plt.xlabel('Atividade')
plt.ylabel('Tempo Final (dias)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.savefig('tempos_finais_atividades_hd_semgrade.png', dpi=300)
plt.show()
try:
    tabela_resultado.to_excel("resultados_planejamento_ag.xlsx", index=False)
```

```
except ImportError:
```

```
    print("[AVISO] Biblioteca 'openpyxl' não instalada. A exportação para Excel  
foi ignorada.")
```

```
plt.figure(figsize=(8,8))
```

```
plt.pie(
```

```
    tabela_resultado['Custo Final (R$)],
```

```
    labels=nomes_atividades,
```

```
    autopct='%1.1f%%',
```

```
    startangle=140,
```

```
    textprops={'fontsize': 7}
```

```
)
```

```
plt.tight_layout()
```

```
plt.savefig('distribuicao_custos_pizza.png', dpi=300)
```

```
plt.show()
```

```
etapas = [0, 1, 2, 3, 4, 5, 6, 7, 8]
```

```
porcentagens_etapa = [0, 20, 20, 20, 10, 10, 8, 7, 5]
```

```
porcentagens_acumuladas = np.cumsum(porcentagens_etapa)
```

```
plt.figure(figsize=(12, 7))
```

```
plt.plot(etapas, porcentagens_acumuladas, marker='o', linestyle='-',  
color='#2A9D8F', linewidth=2.5, markersize=9)
```

```
plt.axhline(100, color='gray', linestyle='--', alpha=0.7)
```

```
plt.text(etapas[-1] + 0.1, 100, '100%', fontsize=11, color='gray', va='center')
```

```
plt.xlabel('Etapas da Obra', fontsize=13)
```

```
plt.ylabel('Percentual Acumulado (%)', fontsize=13)
```

```
plt.xticks(etapas, fontsize=12)
```

```
plt.yticks(range(0, 110, 10), fontsize=12)
```

```

plt.ylim(0, 105)
plt.xlim(0, len(etapas) - 0.5)
plt.tight_layout()
plt.savefig('evolucao_acumulada_planejada_semgrade.png', dpi=300)
plt.show()

satisfacoes_atividade = [
    satisfacao_cliente(c, t)
    for c, t in zip(tabela_resultado['Custo Final (R$)'], tabela_resultado['Tempo
Final (dias)'])
]
norm = Normalize(vmin=72, vmax=75)
plt.figure(figsize=(14, 8))
scatter = plt.scatter(
    tabela_resultado['Custo Final (R$)'],
    tabela_resultado['Tempo Final (dias)'],
    c=satisfacoes_atividade,
    cmap='plasma',
    norm=norm,
    s=140,
    alpha=0.9,
    edgecolors='black',
    linewidths=0.6
)
texts = []
for i, nome in enumerate(tabela_resultado['Atividade']):
    x = tabela_resultado['Custo Final (R$)'][i]
    y = tabela_resultado['Tempo Final (dias)'][i]

```

```

texts.append(plt.text(x, y, nome, fontsize=9, weight='bold', color='black'))

adjust_text(
    texts,
    only_move={'points': 'y', 'texts': 'y'},
    arrowprops=dict(arrowstyle='-', color='gray', lw=0.5, alpha=0.6))
cbar = plt.colorbar(scatter)
cbar.set_label('Satisfação (%)', fontsize=12, weight='bold')
cbar.set_ticks([72, 73, 74, 75])
plt.xlabel('Custo Final por Atividade (R$)', fontsize=11)
plt.ylabel('Tempo Final por Atividade (dias)', fontsize=11)
plt.gca().xaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f'R$
{x:.0f}'.replace(",", "")))
plt.gca().yaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f'{x:.0f}'))
plt.tight_layout()
plt.savefig('grafico_satisfacao_cliente_escala_fixa.png', dpi=300)
plt.show()

```