CriptoAdC: algoritmo para resolução de cifras de substituição monoalfabéticas

Walter Bolitto Carvalho
Universidade Federal do ABC
walter.carvalho@ufabc.edu.br

Resumo — ***O presente trabalho discute o uso dos principais instrumentos para resolução de cifras de substituição monoalfabéticas. Baseando-se na diversidade de métodos encontradas na literatura, foi elaborado um algoritmo para resolução de cifras de substituição simples, capaz de resolver as cifras apresentadas na disciplina de TCCM20120253 / Arquiteturas de Computadores da UFABC, bem como mensagens médias com mais de 250 letras com uma qualidade de 95% das palavras obtidas, e mensagens longas com uma qualidade de 90% de palavras conhecidas, originada de trechos obtidos de livros e filmes.***

Palavras Chave — ***Criptografia, Cifra de Substituição***

I. Introdução

Nos mais diferentes campos da pesquisa científica, é possível identificar a relação do desenvolvimento da ciência com a guerra, como no desenvolvimento de armas de destruição químicas, biológicas e nucleares [1]. No contexto da computação, destaca-se o desenvolvimento da máquina Bombe por Alan Turing, que possibilitou o surgimento dos computadores modernos, e do projeto Arpanet, que possibilitou o surgimento da internet [1][2]. Na área de estudos de criptografia, um incentivo ao desenvolvimento do campo de pesquisa foi seu potencial de guerra, que pode ser identificado há aproximadamente 4.000 anos, sendo popular a Cifra de César, utilizado pelo ditador romano Júlio César em suas missões militares [2].

Enquanto a Cifra de César é baseada no deslocamento do alfabeto original em relação ao criptografado, o Método da Substituição Monoalfabética, identificado inicialmente no século IX, substitui as 26 letras do alfabeto original por quaisquer outras letra, possibilitando $26! - 1$ combinações [3]. Com o avanço do conhecimento da área, outras formas de criptografia mais difíceis de serem resolvidas foram desenvolvidas, como a criptografia de chave simétrica e assimétrica, assim como encontra usos diversos no cotidiano para além do contexto militar, como dados do mercado financeiro, arquivos governamentais e de comunicação pessoal [4].

Este artigo visa apresentar um algoritmo capaz de solucionar cifras por substituição, integrando diferentes aspectos desse tipo de abordagem.

II. Metodologia

A natureza da pesquisa, apresentada no presente trabalho, é caracterizada como uma pesquisa aplicada, por propor contribuições a partir do conhecimento acumulado de cifra de substituição. Quanto aos seus objetivos, pode ser classificada como exploratória, por apresentar o desenvolvimento do algoritmo aprofundando conhecimentos estabelecidos na literatura.

A metodologia de pesquisa e desenvolvimento baseia-se nos seguintes momentos: (1) Revisão de Literatura, visa compreender os aspectos centrais para aplicações com cifra de substituição, identificando a diversidade de soluções possíveis; (2) Elaboração do algoritmo, visa a construção de um código capaz de decifrar duas mensagens propostas em inglês, sendo organizado também um modelo para maior compreensão do funcionamento da ferramenta, bem como para compreensão de sua eficiência; e (3) Estudo de Caso e Discussão, visa aplicação do algoritmo com as mensagens propostas e de mensagens geradas, a fim de compreender os potenciais e limitações da ferramenta, por meio de uma discussão dos resultados obtidos.

III. Revisão Bibliográfica

Schneier [5] define a cifra de substituição como a substituição de cada carácter de um texto simples por outro carácter com uma cifra, sendo possível quatro tipos de cifra: simples, homofônica, poligramática e polialfabética.

No contexto de resolução de uma cifra de solução monoalfabética, Friedman [6] identifica formas potenciais diversas para solucionamento, como análise de frequência de letras mais comuns no idioma inglês, presença de pares de letras (dígrafos) e trios de letras (trígrafos) mais comuns em palavras, destacando o dígrafo 'th', além da distinção de vogal de consoante observando a posição em que mais ocorrem entre as classificações das duas letras. Sobre a análise de frequência de letras, o autor destaca as letras 'e', 't', 'a', 'o', 'n' como as que mais aparecem em textos na língua inglesa.

Hart [7] aponta o potencial da análise de frequência de letras, e também aborda a frequência de palavras mais conhecidas, pois de acordo com o autor, cada palavra de uma frase tem 50% de chance de que a palavra faça parte das 135 palavras mais frequentes no contexto do idioma inglês, mesmo considerando o idioma inglês apresenta mais de cem mil palavras. O autor propõe um algoritmo para solucionar a mensagem cifrada ao comparar cada palavra de um dicionário com as palavras do dicionário, estando atento ao tamanho da palavra e posições das letras repetidas, com a proposta de construção do algoritmo usando busca em profundidade [7].

Pesquisadores de outros países investigam quebras de criptografias em outros idiomas, Quaresma e Pinho [8] analisam as principais letras em textos em português, destacando-se as letras 'a', 'e', 'o', 's' e 'r', observando também como destaque os bigramas 'de', 'ra', 'os', 'es' e 'as' e os trigramas 'que', 'ent', 'nte', 'com' e 'ado'. Os autores também notam que algumas palavras destacam determinadas letras finais, como as letras 'a', 'o', 's', 'e' e 'm', com menor diversidade de letras finais que na língua inglesa. Assim, outros idiomas apresentam metodologias parecidas de resolução de cifras, utilizando-se de dicionários adequados e do enfoque em outras letras mais utilizadas.

IV. ALGORITMO

Foi projetado um algoritmo, intitulado CRIPTOAdC, para a disciplina TCCM20120253 / Arquiteturas de Computadores, realizada no 3° Quadrimestre de 2025 da UFABC, capaz de descriptografar uma mensagem em inglês, cujo texto está representado na sua forma binária. A fim de ser possível discutir sobre aspectos do CRIPTOAdC, foi proposto um modelo em 6 partes, sendo elas: (1) Transformação do binário em letras; (2) Letra mais comum; (3) Dicionário de palavras mais comuns; (4) Dicionário completo de palavras; (5) Pontuação; e (6) Avaliação do resultado. O modelo, identificando as ações em laranja e os artefatos presentes em azul é ilustrado na Figura 1.

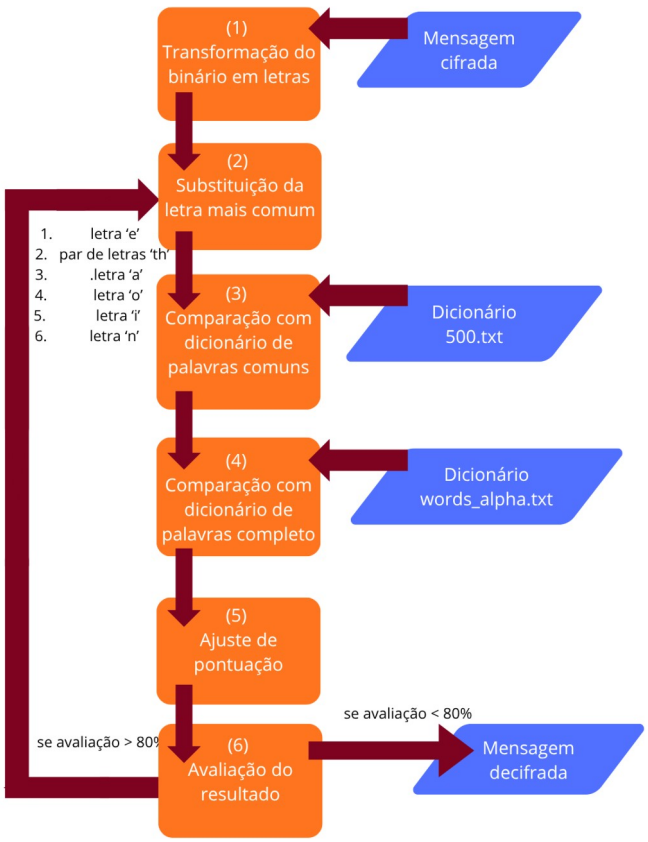

Fig. 1. Modelo do algoritmo CRIPTOAdC.

Sendo identificado que o padrão de mensagens a ser analisada apresenta códigos binários com até sete dígitos e caracteres diversos com até seis dígitos, foi feito uma transformação rápida de binário para letra (quando sete dígitos) e para número (quando seis dígitos) na **Etapa (1)** para facilitar a visualização do operador do algoritmo em outras etapas, assim como o binário '100000' foi traduzido como barra de espaço.

Na **Etapa (2)**, considerando o potencial observado na literatura sobre o uso de análise de frequência, é realizada a identificação da letra mais repetida, e em seguida, a mesma é substituída por 'e', caso ao final do processo, o algoritmo não seja capaz de decifrar, essa etapa irá se repetir usando outras sugestões de análise de frequência, como o dígrafo 'th', e caso seja necessário, novamente rodará utilizando as seguintes letras como mais comuns: 'a', 'o', 'i', e 'n'.

Nas **Etapas (3) e (4)**, foram realizadas uma comparação de cada palavra cifrada com cada palavra apresentada no dicionário, de forma que quando uma cifra apresenta uma única possibilidade de palavra no dicionário de mesmo tamanho e compartilhando as letras já decifradas, ela era substituída e suas letras são adicionadas ao mapa de palavras. Considerando que esta análise ocorre em lista, o algoritmo realiza isso em *loop* em rodadas posteriores para que, com novas versões do mapa, seja possível solucionar as outras palavras. A diferença entre as duas etapas é que, a **Etapa (3)** usa apenas as 500 palavras mais comuns do idioma inglês[1], como sugerido na literatura [7]; e a **Etapa (4)** é aplicada em seguida com um dicionário de inglês com 479 mil palavras[2], incluindo algumas siglas e termos utilizados em ambientes digitais.

Na **Etapa (5)**, no caso da mensagem apresentar pontuações diversas como vírgulas, pontos e hifens, é identificado quais binários de seis dígitos representam cada caractere e é realizada a substituição necessária.

É realizada uma avaliação da mensagem decifrada na **Etapa (6)**, identificando a quantidade de palavras conhecidas, ou seja, que são encontradas no dicionário de 479 mil palavras, com a totalidade de palavras na mensagem e caso não seja obtida uma porcentagem adequada, o algoritmo retorna à Etapa (2). No momento que o código é interrompido ou o último *loop* é realizado, sendo capaz ou não de solucionar a cifra, é medido o tempo de funcionamento do mesmo.

Em relação a eficiência do algoritmo, no contexto da análise da complexidade de tempo do algoritmo proposto, a Figura 2 ilustra em pseudocódigo a complexidade de tempo do pior caso, sendo podemos identificar uma complexidade à quarta potência.

No decorrer dos testes, com base nas mensagens solicitadas na disciplina para que fossem decifradas, foi observado que não seria necessário o *loop*[3] proposto para teste com uma nova letra ou dígrafo após uma avaliação

[1] Dicionário foi obtido em: https://gist.github.com/theRemix/48181ee5d45c9f01033a. Acesso em 20/10/2025.

[2] Dicionário foi obtido em: https://github.com/dwyl/english-words. Acesso em 20/10/2025.

[3] (while score <0.8 and ponteiro <7)

inadequada, já que foram obtidos resultados adequados já na primeira iteração, de forma que a complexidade de tempo poderia ser cúbica. Apesar disso, foi tomada a decisão de manter a complexidade à quarta potência para eventuais mensagens cifradas que não apresentassem a letra 'e' como letra mais comum, aspecto que será aprofundado na próxima seção.

O algoritmo CRIPTOAdC, os dicionários utilizados e um detalhamento dos testes discutidos no Estudo de Caso podem ser consultados por meio do link: https://github.com/walterbolitto/adcufabc.

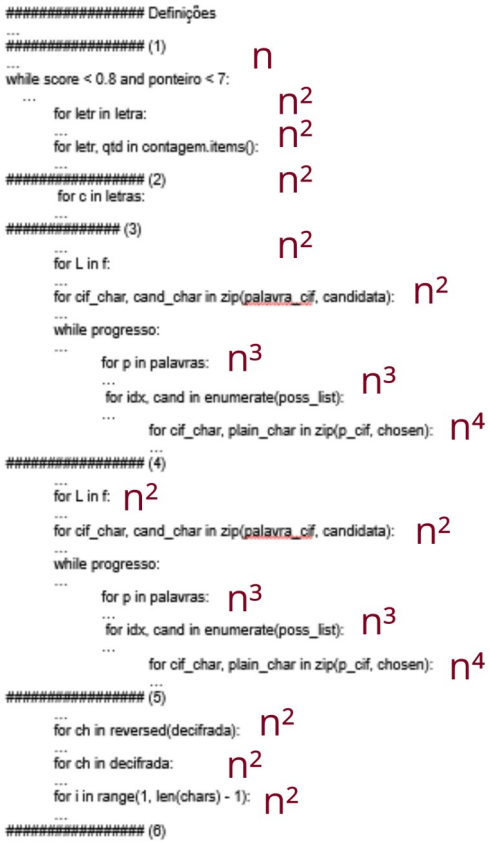

Fig 2. Modelo do algoritmo CRIPTOAdC.

V. Estudo de caso e Discussão

Além das duas mensagens estabelecidas para serem decifradas (A1 e A2), foram definidas outras mensagens para compreensão do potencial e limitações do algoritmo proposto, originadas de livros e filmes. As outras mensagens foram elaboradas seguindo os moldes das duas mensagens da disciplina, de forma que as letras seriam substituídas por um código binário de sete dígitos, e os espaços no texto foram substituídos por '100000'. As mensagens inventadas não apresentam pontuação com caracteres especiais, pelo enfoque na discussão sobre aspectos observados ao se analisar as mensagens, como tamanho do texto e escolha de palavras. A Tabela I apresenta informações resumidas das mensagens analisadas. Além disso, os testes foram realizados na ferramenta Jupyter Notebook[4].

Em relação as duas mensagens definidas inicialmente na disciplina (A1 e A2), o algoritmo foi capaz de resolver tanto a cifra quanto os caracteres especiais em até 15 segundos e com mais de 99% palavras conhecidas nas duas mensagens, tempo observado em outras mensagens de tamanho médio para A1 e grande para A2.

O algoritmo apresentou dificuldades de solucionar mensagens mais curtas, em especial às com menos de 250 letras (I1, I2, I3, I4 e I5), havendo exceções como A8 e A11. Os dois casos solucionados neste contexto levaram menos de quatro segundos, um deles teve uma qualidade de 95% (A11), e A8 de 80% devido à presença de nomes próprios na mensagem.

TABELA I
MENSAGENS DECIFRADAS

Cifra	Total de Letras	Qualidade (%)	Tempo (s)	Especificidade
A1	370	100	10.23200	Cifra do Moodle
A2	1767	99.19	14.14300	Cifra do Moodle / Caracteres
A3	602	98.68	8.31000	
A4	884	99.50	12.96400	
A5	398	98.88	6.26000	
A6	302	100.00	4.65800	
A7	311	95.00	9.97500	
A8	125	80.77	1.81000	Curta
A9	451	98.08	14.79100	Par de letras 'th' / letra 'o'
A10	380	98.68	17.80600	Par de letras 'th'
A11	154	94.87	3.97000	Curta / Par de letras 'th'
A12	3494	97.10	122.2580	Longa
A13	3669	99.54	103.0630	Longa
A14	6498	90.44	80.78200	Longa
A15	373	100.00	20.95800	Par de letras 'th'
I1	195	52.27	27.06000	Curta / Resposta Inadequada
I2	183	78.43	18.28200	Curta / Resposta Inadequada
I3	243	68.25	20.13700	Curta / Resposta Inadequada
I4	131	68.97	17.48200	Curta / Resposta Inadequada
I5	146	54.29	19.75800	Curta / Resposta Inadequada

Nos testes que não chegaram a resultados adequados, o algoritmo retornava índices diferentes de qualidade em alguns casos, especialmente com palavras que não correspondiam às esperadas. Analisando os resultados obtidos caso a caso, foi definida que era necessário 80% de palavras conhecidas para identificar uma mensagem decifrada como adequada. O tempo necessário para todos os *loops* serem realizados nas mensagens curtas que ofereceram respostas inadequadas foram de até 20 segundos.

O código detalha o comportamento em cada etapa, em especial às possibilidades e conflitos presentes na comparação de palavras nas Etapas (3) e (4), colaborando com melhorias pontuais e com a compreensão das limitações do algoritmo. O primeiro aspecto observado foi a dificuldade do CRIPTOAdC de solucionar mensagens curtas, pois dois motivos: (a) ao comparar com o dicionário grande, mais de uma possibilidade de solução era apresentada, sem haver um avanço em letras mapeadas, o que interrompia o funcionamento do algoritmo; e (b) considerando a qualidade de palavras totais no dicionário maior, muitas palavras eram

[4] JupyterLite. Disponível em: https://jupyter.org/try-jupyter/lab/. Acesso em 28/10/2025.

pouco usuais para livros e filmes, apresentando em alguns casos soluções erradas que não geravam conflitos com outros termos.

Em mensagens de tamanho médio (A1, A2, A3, A4, A5, A6, A7, A9 e A10), entre 250 e 1000 letras, o algoritmo apresentou um funcionamento adequado, potencializado pela escolha de substituir a letra mais comum pela letra ‘e’, não gerando *loops* com outras letras iniciais ou par de letras.

Também foi observado no estudo de caso para mensagens de tamanho médio que, apesar de que o dicionário maior ser capaz de oferecer soluções com palavras pouco adequadas como siglas e jargões, surgiam conflitos ao final de rodadas de análise das etapas (3) e (4) com outras palavras mais adequadas, fazendo com que o algoritmo deixasse de gerar certas traduções desejadas, levando a mapeamentos errados. Para essas mensagens, o tempo necessário foi entre 5 segundos (A6) e 18 segundos (A10)

Sobre mensagens com mais de 1000 letras (A12, A13 e A14), apesar da expectativa de serem mais fáceis de resolução, pela grande quantidade de palavras, o dicionário maior traduzia algumas palavras com expressões pouco comuns, assim como palavras particulares do texto, como nome próprio (Alice traduzido como alive) levavam a quebra da cifra para outros lugares, gerando acúmulo de erros. As mensagens grandes levaram entre 80 e 120 segundos para serem resolvidas, e apresentaram um índice de qualidade acima de 90%.

Apesar de A14 apresentar 6498 letras, quase o dobro de A12 e A13, seu tempo de execução foi menor (80 segundos em comparação a 100 e 120 segundos), isso decorre do fato de que a letra ‘e’ não colaborou com a resolução de A12 e A13, sendo também observado o potencial do par de letras ‘th’ se destacar no contexto de mais palavras na mensagem cifrada.

Além dos dois textos longos (A12 e A13), o par de letras ‘th’ foi importante para a resolução de outras cifras menores (A9, A10, A11 e A15). A cifra A10 foi obtida intencionalmente de um texto para ensinar dígrafos para crianças[5]; a cifra A11 era muito curta e cada palavra apresentava muitas possibilidades, ao mesmo tempo que o par de letras ‘th’ ajudou que uma cifra de 154 palavras fossem concluídas; por fim, a cifra A15 apresentou um resultado relativamente adequado no primeiro *loop*, alcançando o valor de qualidade estabelecido para o algoritmo com o par de letras ‘th’.

A mensagem A9 é originada de um texto que não apresenta a letra ‘e’[6], apesar disso, no primeiro *loop* apresentou uma qualidade insuficiente de quase 50%, sendo resolvida no par de letras ‘th’, mas caso fosse removido, também seria solucionado de forma adequada com a substituição da letra mais comum com a letra ‘o’ em determinada iteração. Desta forma, o conjunto de seis mensagens analisadas no decorrer do Estudo de Caso (A9, A10, A11, A12, A13 e A15) justifica a manutenção do *loop* inicial do algoritmo, mesmo com o impacto no tempo computacional para sua operação pelo complexidade à quarta potência.

[5] Digraph Reading 10 - Ancient Throne Worksheet. Disponível em https://readingduck.com/worksheet/th-digraph-reading-10/. Acesso em 26/10/2025.

[6] Wright, Ernest Vincent. *Gadsby: Enriched edition. A Story of Over 50,000 Words Without Using the Letter" E"*. Good Press, 2019. Acesso em 26/10/2025.

VI. Conclusão

O algoritmo proposto atingiu o objetivo proposto de ‘apresentar um algoritmo capaz de solucionar cifras por substituição, integrando diferentes aspectos desse tipo de abordagem’, solucionando com facilidade as mensagens médias criptografadas com mais de 300 letras com um critério de 95% de palavras conhecidas, enquanto mensagens longas apresentavam mais de 90% de palavras conhecidas, no qual foi possível a compreensão total da mensagem em todos os arquivos decifrados.

O algoritmo também possibilitou uma compreensão de como as diferentes metodologias da literatura colaboram individualmente e coletivamente para solucionar cifras de substituição monoalfabéticas diversas, colaborando também com a solução de casos específicos, como cifras com escolhas textuais incomuns.

Referências

[1] C. Bueno. “Ciência para a guerra e para a paz: uso militar ajudou a ciência a avançar, mas o papel da ciência na busca pela paz é fundamental”. *Ciência e Cultura*, v. 74, n. 4, p. 01-06, 2022.

[2] W. W. M. Silva. “A evolução da criptografia e suas técnicas ao longo da história”. 2019. 29 f. TCC (Graduação) - Curso de Sistemas de Informação, Instituto Federal Goiano, Goiás, 2019. Disponível em: https://repositorio.ifgoiano.edu.br/bitstream/prefix/795/1/tcc_Willian_Wallace_de_Matteus_S ilva.pdf. Acesso em: 25 out. 2025.

[3] F. O. Loureiro. Tópicos de Criptografia para o Ensino Médio. Dissertação de Mestrado em Matemática, Universidade Estadual no Norte Fluminense Darcy Ribeiro, Rio de Janeiro. Disponível em: http://uenf.br/posgraduacao/matematica/wp-content/uploads/sites/14/2017/09/29082014Flavio-Ornellas- -Loureiro.pdf. Acesso em: 25 out. 2025.

[4] R. R. Oliveira. "Criptografia simétrica e assimétrica-os principais algoritmos de cifragem." *Segurança Digital [Revista online]*. 31 (2012): 11-15.

[5] Schneier B. “Applied cryptography: protocols, algorithms, and source code in C”. John wiley & sons; 2007.

[6] W.F. Friedman, “Elements of Cryptanalysis, Training Pamphlet No. 3”. Washington, DC: Government Printing Office, 1923.

[7] G. W. Hart. “To decode short cryptograms”. Communications of the ACM, v. 37, n. 9, p. 102-108, 1994.

[8] P. Quaresma and A. Pinho. “Análise de frequências da língua portuguesa”. *Livro de Actas da Conferência Ibero-Americana InterTIC*. p. 267-272, 2007.