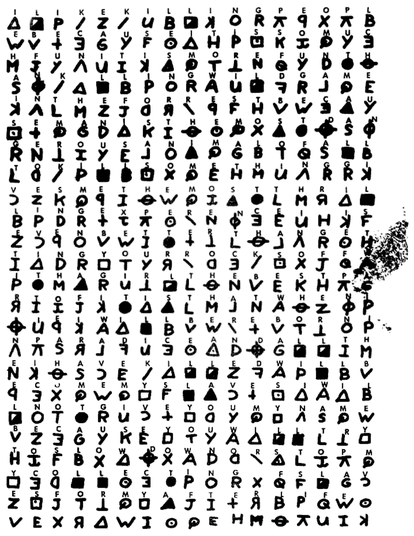
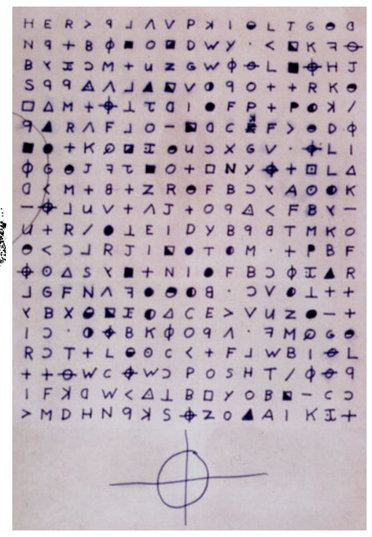
Evolutionary Algorithm for Decryption of Monoalphabetic Homophonic Substitution Ciphers Encoded as Constraint Satisfaction Problems

David Oranchak, doranchak@gmail.com http://oranchak.com

In 1969, the **Zodiac serial killer** sent two ciphers to San Francisco newspapers. The first cipher was solved very quickly by a high school teacher and his wife. **Forty years later, the second cipher remains unsolved.**



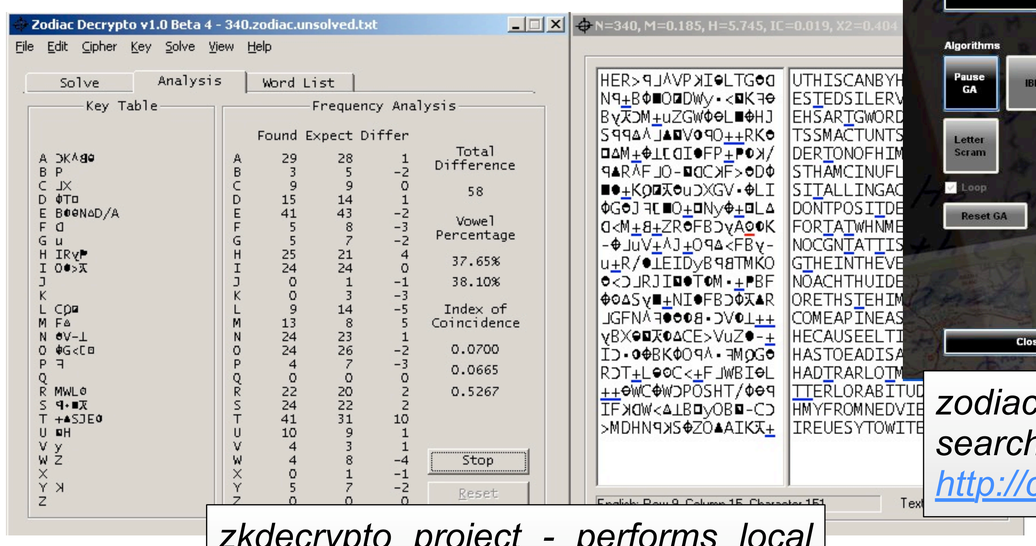
408-character cipher, with solution key.



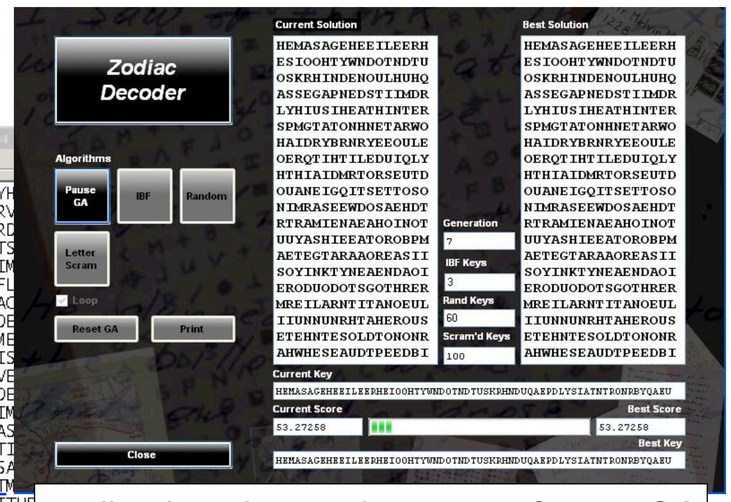
Unsolved 340-character cipher

Some have attempted to decipher the 340-character cipher using **n-gram frequency analysis**. Problems:

- Ciphertext must be sufficiently large
- n-gram frequencies vary depending on source text

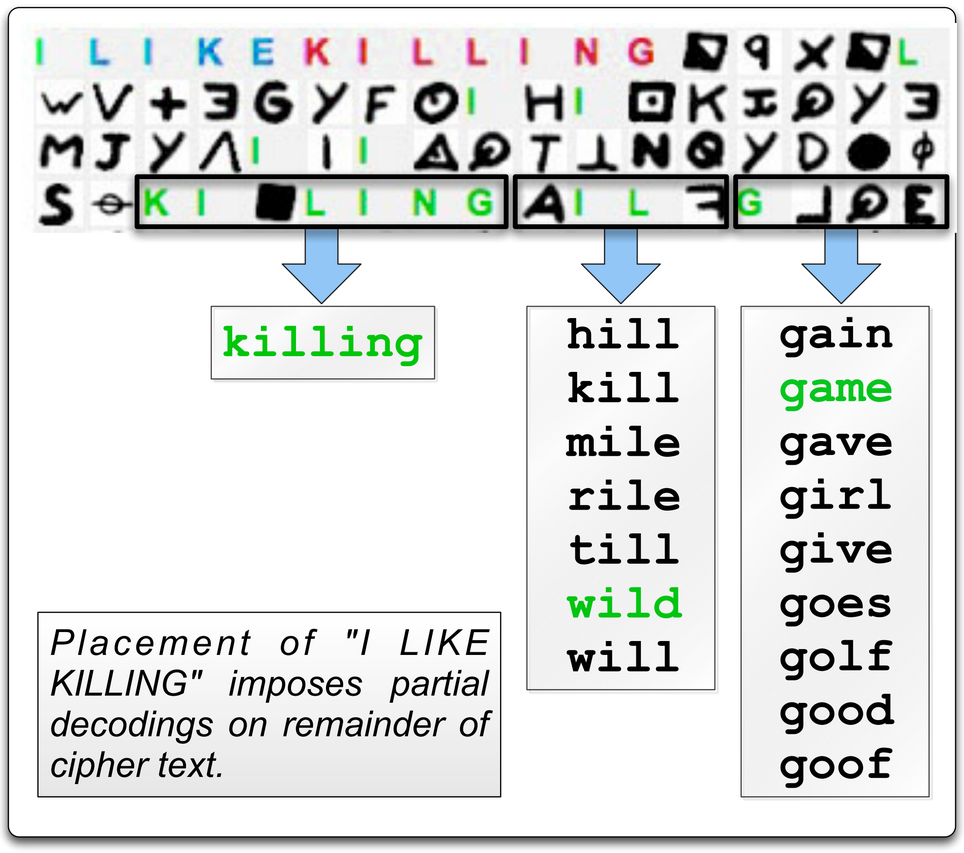


zkdecrypto project - performs local search using frequency analysis
http://code.google.com/p/zkdecrypto/



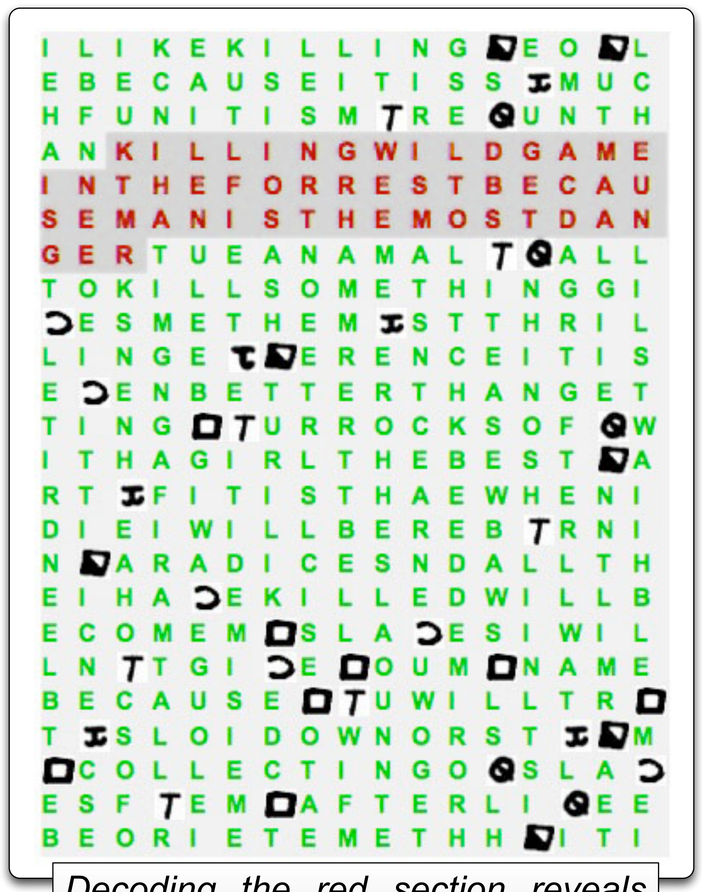
zodiacdecoder project - performs GA search using frequency analysis
http://code.google.com/p/zodiacdecoder/

We use a **dictionary-based attack:**
Place words in one section of cipher to impose constraints on other sections of cipher. Correct placements produce partial word decodings in other sections of cipher text. Dictionary is indexed by constraints for fast lookup.



Placement of "I LIKE KILLING" imposes partial decodings on remainder of cipher text.

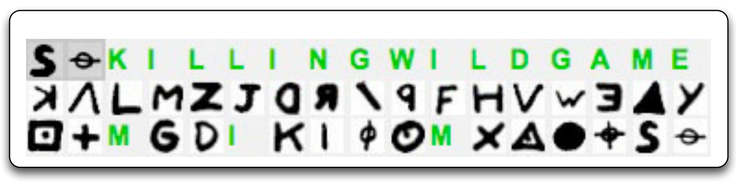
Reduce the search space by concentrating attack on small section of ciphertext.



Decoding the red section reveals 90% of the plaintext.

Evolutionary approach:

- Genome encodes attacks as (word,position) tuples. Words are drawn from a fixed dictionary.
- Infeasible encodings are immediately rejected.
- Each individual starts with a single tuple. Subsequent generations add more tuples via crossover and mutation.
- Fitness function measures counts and coverage of potential words that appear when the genome's words are plugged into the cipher text.

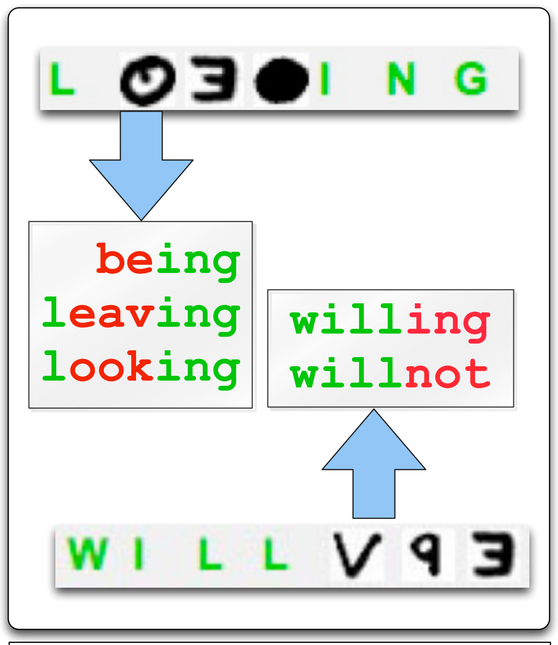


Example genome: {(0,0), (1,7), (2,11)}

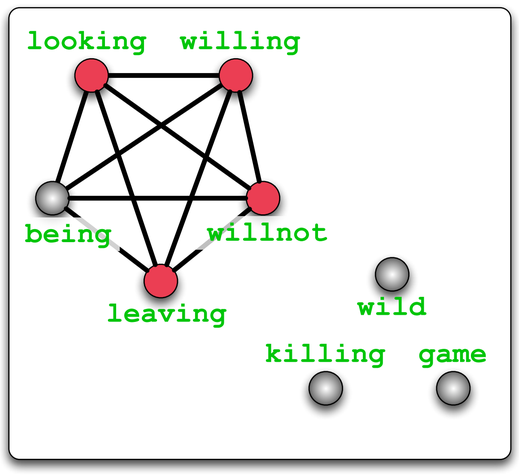
- W_0 ("killing") at position 0
- W_1 ("wild") at position 7
- W_2 ("game") at position 11

Conflicting decodings arise when we look up potential word matches. We seek a maximal set of words that produces no conflicts. This is a vertex cover problem. To find a factor-2 approximation of vertex cover quickly, we find maximal matchings in the conflict graphs.

Results: For dictionary sizes up to 1600 words, algorithm was able to find correct decodings for the 408-character cipher. 340-character cipher remains unsolved. We are working on making our technique more robust for future attacks.



Partial decoding imposed by genome "KILLING(0) WILD(7) GAME(11)." Conflicted decodings are shown in red.



Conflict graph for found words. Removal of red nodes results in maximum non-conflicting words. Remaining words contribute to coverage score in the fitness function.

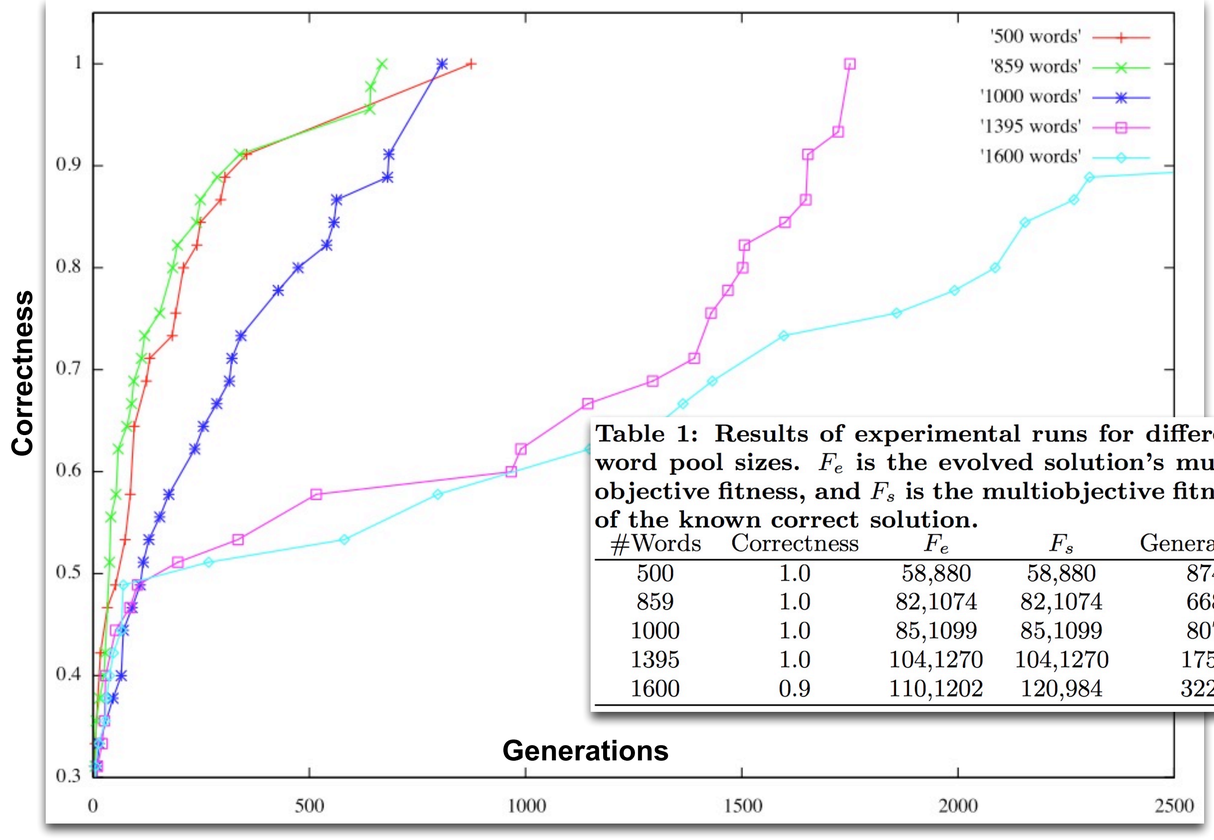


Table 1: Results of experimental runs for different word pool sizes. F_e is the evolved solution's multi-objective fitness, and F_s is the multiobjective fitness of the known correct solution.

#Words	Correctness	F_e	F_s	Generations
500	1.0	58,880	58,880	874
859	1.0	82,1074	82,1074	668
1000	1.0	85,1099	85,1099	807
1395	1.0	104,1270	104,1270	1750
1600	0.9	110,1202	120,984	3222