Bad cipher design: Chessography and Cascaded Spin Shuffle

Martin Stanek[1]

[1]*Department of Computer Science, Faculty of Mathematics, Physics and Informatics, Comenius University*

Abstract

This paper shows the weaknesses of two symmetric encryption schemes – Chessography and Cascaded Spin Shuffle. The security claims made by their authors are unsubstantiated. Despite being featured in peer-reviewed publications, their flaws are readily apparent and do not require any sophisticated cryptanalysis. Consequently, the paper proposes a set of speculative "red flag" indicators aimed at identifying encryption proposals of potentially questionable quality.

Keywords

Encryption, Symmetric ciphers, Cryptanalysis

1. Introduction

Symmetric encryption schemes are important cryptographic constructions and are extensively used in many applications to ensure confidentiality of data. Best practices and cybersecurity guidelines require using standardized and approved encryption schemes for sensitive data. Those schemes are assumed to be sufficiently analyzed and their implementation and application scenarios understood from a security perspective.

Nevertheless, there is a constant stream of new symmetric encryption scheme proposals. These proposals often lack clear motivation, as well as any meaningful security analysis. A partial explanation for this phenomenon is that it is easy to design a bijective function with an additional parameter (key), if you discard most of the modern cryptanalytic knowledge. Thus, one can obtain a *correct* encryption scheme, although with dubious security.

We believe the academic practice of proposing schemes without a proper motivation and analysis should be discouraged.

Our contribution. We discuss the security of two symmetric encryption schemes, Chessography and Cascaded Spin Shuffle. These schemes were chosen because of their poor design. We show that both schemes are insecure, despite security claims of their authors. Moreover, the analysis is basic and straightforward, not requiring any advanced cryptanalytic methods.

We propose "red flag" indicators that can help identify problematic proposals. Whether these indicators are sufficient or if other indicators might be more useful remains an open question. Even so, avoiding red flags can help improve proposals.

Related work. Quality of cryptographic algorithms is an important part of security controls, where *best practice* requires standardized and approved algorithms [1]. Analysis and prevention of cryptographic failures often focus on implementation and configuration problems, since these occur in practice [2, 3]. Weaknesses of obscure, low-profile algorithms might go unnoticed. We are not aware of a study that addresses this type of algorithms specifically.

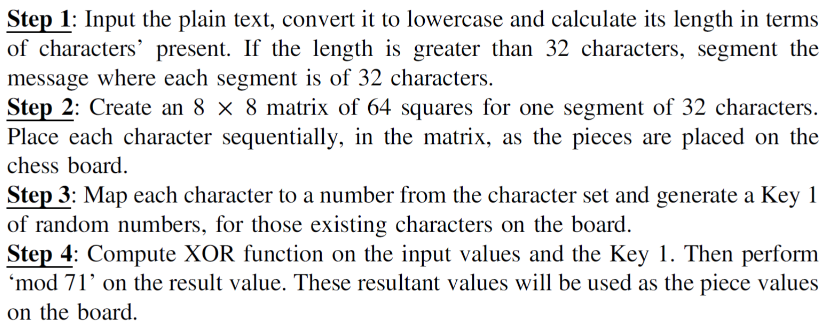
This paper is organized into four sections. After the introduction, we discuss Chessography in Section 2 and Cascaded Spin Shuffle in Section 3. Red flag indicators are presented in Section 4.

ITAT'25: Information Technologies – Applications and Theory, September 26–30, 2025, Telgárt, Slovakia

martin.stanek@fmph.uniba.sk (M. Stanek)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2025-11-13



Step 1: Input the plain text, convert it to lowercase and calculate its length in terms of characters' present. If the length is greater than 32 characters, segment the message where each segment is of 32 characters.
Step 2: Create an 8 × 8 matrix of 64 squares for one segment of 32 characters. Place each character sequentially, in the matrix, as the pieces are placed on the chess board.
Step 3: Map each character to a number from the character set and generate a Key 1 of random numbers, for those existing characters on the board.
Step 4: Compute XOR function on the input values and the Key 1. Then perform 'mod 71' on the result value. These resultant values will be used as the piece values on the board.

Figure 1: First four steps of encryption [5, Sect. 3.1]

2. Chessography

Chess is an interesting and complex game with a vast number of possible positions. This sometimes leads to ideas for combining chess with cryptography [4, 5, 6, 7]. Chessography [5] is a symmetric encryption scheme. The main idea of this scheme is to use a chess game to encrypt a plaintext block of 32 characters. Plaintext characters are placed on squares where white and black pieces are positioned at the start of the game. These characters are then transformed using the first key (Key 1), and the game is used to move them on the board. The second key is based on a particular chess game, forming so called "paired key". However, the details of its construction and use are irrelevant for our discussion. The ciphertext consists of the final position of the pieces on the board, together with additional information that allows to reconstruct the initial positions of the pieces, including those that were captured during the game. The main objections to the quality and strength of the Chessography scheme are summarized in the following section.

2.1. Failures of Chessography

Imprecise description of the scheme

The description of the encryption and decryption algorithms is rather vague. It lacks mathematical formulas, reference implementation, or pseudocode. It is unclear whether distinct chess games are used for subsequent plaintext blocks or if a single game is used for all blocks. The provided example also leaves numerous questions unanswered, for example, what is the exact procedure to produce the final ciphertext, since the text description does not correspond to the ciphertext presented in figures.

The scheme is incorrect

The scheme uses an alphabet with 71 characters, which are encoded as numbers ranging from 1 to 71. Figure 1 illustrates the initial four steps of the encryption algorithm. There are two notable issues with step 4. A minor issue is that after a modulo 71 operation, the range changes to $0, \dots, 70$.

A major issue is that XOR-ing with randomly chosen "Key 1", whose values can be quite large[1], and performing modulo 71 operation is not reversible. For instance, consider the plaintext characters encoded as numbers 9 and 64, and the key value 62 for both numbers:

$$(9 \text{ XOR } 62) \bmod 71 = 55 \bmod 71 = 55;$$
$$(64 \text{ XOR } 62) \bmod 71 = 126 \bmod 71 = 55.$$

It is impossible to tell what the original plaintext number was just from the result 55 and the key value 62. Hence, the step 4 is not reversible. It does not matter what the next transformations are, the decryption is unable to distinguish the correct plaintext.

[1] An example for Key 1 in the original paper contains values as low as 29 and as high as 943 [5, Figure 9].

Chess-related permutation is (sometimes) irrelevant

Let's assume that the "XOR-mod" step is correct, e.g., the scheme uses an alphabet with 64 characters, numbered from 0 to 63, and values in Key 1 are 6-bit integers. If only a single block is encrypted, this part of the encryption algorithm functions as a one-time pad cipher, achieving perfect secrecy. Any chess-related steps thereafter are irrelevant. If new Key 1 is chosen for each plaintext block separately, and the paper [5] can be interpreted both ways (yes and no), this observation extends to the entire ciphertext. The key is long, the first part of the encryption algorithm is a one-time pad, and other transformations are redundant for secrecy.

If Key 1 is the same for each block, which is likely the intended construction, a known plaintext attack becomes a problem. It is possible to reconstruct values of Key 1, at least for characters presented in the final position on the chess board, and depending on details of the encryption algorithm even for the entire block.

Chess part of the scheme is incomplete

The scheme encodes moves by creating pairs of squares where a piece was and where it moved to, respectively. There is no mention whether this encoding is able to correctly handle moves like castling, en passant, and pawn promotion.

Remark. As a curiosity, the example game used in [5] is the following one (annotation symbols were added by Stockfish):

```
1. b4 e6 2. c3 f5 3. g3 g6 4. Nf3 Bd6?! 5. h4 Nf6 6. Nd4?! a6 7. e3 Bf8
8. Qf3? Nd5? 9. Nc2 Nc6 10. e4 Ne5 11. Qe2 fxe4 12. d4?? c5??
13. dxe5 cxb4 14. cxb4?! Rb8? 15. Bg5?! Qc7 16. h5?? Ra8?? 17. hxg6 Be7?!
18. g7 Bxb4+?? 19. Nxb4 h6 20. Bxh6?! d6?! 21. Qh5+ Ke7 22. Qg5+ Kf7
23. gxh8=Q Nxb4 24. Qh7+ Ke8 25. Qh5+ Kd8 26. Bg5+ Qe7 27. Bxe7+ Kd7
28. Bxd6+ Kc6 29. Bxb4 Bd7 30. Qxe4+ Kb6 31. Qhh7 Bc6 32. Qd4#
```

The game is rather illogical, full of blunders, and white overlooks multiple *mate in 1* opportunities, the first one in move 20.

Weak chess games

Some chess games are only a few moves long, fore example, Scholar's Mate, Fool's Mate, and Legal's Mate. These and other games leave many pieces on their original squares, thus weakening the resulting permutation. The proposal [5] does not address the possibility of weak chess games for the Chessography, nor does it explain how to select suitable chess games for encryption.

Chess game permutation is weak

Let's assume the chess game is generated by a chess engine to be human-like, or selected from a huge pool games played by humans. The final composition and placement of pieces is far from statistically random, let alone cryptographically strong. A simple analysis presented in Section 2.2 demonstrates this convincingly. Using the final position of a chess game and intermediate moves as a permutation component in a cipher is a bad idea.

The claim "*The strength of this algorithm is based upon the complexity of the chess game.*" [5] by the author of Chessography, and then using the estimate for the number of possible chess games to argue the scheme's security, is simply deceiving: "*...any intruder wishes to orderly break the cipher text, through the knowledge of chess game ...will take a long time with an estimate as given* 10^{50} *board positions and* 10^{123} *sequences available to try out.*" [5].

Table 1
The most common piece placement in the final position

piece	white		black	
	max [%]	square	max [%]	square
king	22.28	g1	22.46	g8
queen	2.52	d1	3.31	d8
bishop	3.83	g2	4.75	g7
knight	3.88	f3	4.03	f6
rook	10.45	a1	12.15	a8
pawn	26.22	f2	27.89	f7

2.2. Analysis of chess games final positions

The dataset consists of 100,000 games played on the Lichess server by users with an average rating of 2558. It is a subset of games played in October 2024 [8]. The dataset contains mostly blitz and rapid games and excludes bullet time controls. White win rate is 47%, black win rate is 42%, and only 11% of the games are draws. The average length of the game is 43 moves (86 plies).

Figures 3 and 4 in Appendix A show heatmaps for the location of different pieces on the chess board for the final position of the game. The heatmaps illustrate the limited randomness of the final positions and subsequently weak (partial) permutations that chess games provide. Colors are scaled individually for each heatmap, therefore the same shade can represents different percentage in distinct heatmaps. Table 1 summarizes the maximal percentages and placement on the board for each piece type. In conclusion, the rules of chess limit the range of possible moves, and the placement of pieces in final positions is not sufficiently random for cryptographic applications.

3. Cascaded Spin Shuffle

Cascaded Spin Shuffle (CSS) is a transposition cipher proposed in [9]. The cipher uses a very simple permutation – it arranges plaintext in a $n \times n$ grid (representing a torus) in a circular, spiral motion. The starting position and the spiral direction is determined by a key. The ciphertext is formed by reading permuted characters row by row from the grid. An example for 7×7 grid with the initial position in the first row and fourth column, and the spiral motion starting *left* and *clockwise* (see also Figure 2):

$$\begin{bmatrix} 28 & 11 & 2 & 1 & 6 & 19 & 40 \\ 27 & 10 & 9 & 8 & 7 & 20 & 41 \\ 26 & 25 & 24 & 23 & 22 & 21 & 42 \\ 49 & 48 & 47 & 46 & 45 & 44 & 43 \\ 31 & 32 & 33 & 34 & 35 & 36 & 37 \\ 30 & 13 & 14 & 15 & 16 & 17 & 38 \\ 29 & 12 & 3 & 4 & 5 & 18 & 39 \end{bmatrix}$$

The corresponding output permutation starts with 28, 11, 2 ...and ends with 5, 18, 39.

3.1. Failures of CSS

Small key space

The CSS scales the grid size according to the plaintext length. It doubles n until the whole plaintext fits into the grid. The plaintext is padded with random characters to fill the grid. Therefore, the plaintext length is proportional to n^2 regardless of padding length.

A	E	A	B	P	S	P
Y	D	R	E	H	S	I
H	P	A	R	G	O	N
E	L	F	F	U	H	S
C	A	S	C	A	D	E
D	I	G	N	C	H	D
N	S	D	C	I	E	S

Plaintext: BADCIPHERDESIGNCHESSOGRAPHYANDCASCADEDSPINSHUFFLE
Ciphertext: AEABPSPYDREHSIHPARGONELFFUHSCASCADEDIGNCHDNSDCIES

Figure 2: The CSS example, see also [9, Figure 8]

The CSS uses an unnecessarily complicated procedure to derive the key. At the end, the key contains a starting position in the grid and initial directions for the spiral movement: $\{up, down, left, right\} \times \{clockwise, anticlockwise\}$, i.e., overall $8n^2$ keys. Hence, the brute-force attack has linear complexity with respect to the plaintext length, assuming the wrong keys can be recognized and discarded with only a few initial characters decrypted. If the attacker decrypts the entire ciphertext with all possible keys, there is only polynomial (quadratic) overhead in this approach.

It is straightforward to conclude that the key space size is insufficient.

Weak permutation

The resulting permutation will always contain significant parts of consecutive characters. In the previous example with a 7×7 grid the sequence of 31, 32, ..., 37 will appear in the output. Similarly, the row above contains similar consecutive sequence, 43, ...49, just reversed. Nontrivial substrings of plaintext appearing in the ciphertext indicate a vulnerability of the scheme. This problem becomes worse with increasing grid size.

Moreover, the attacker can use these substrings to narrow the starting position of the spin, since these appear in approximately $n/2$ distance from the starting position. Similarly, the spin direction (*clockwise* or *anticlockwise*) can be derived from the correct continuation of such substring in the grid.

Unsubstantiated security claims

The original paper contains some ideas for strengthening the CSS. Unfortunately, none of them really work and address the small key space and weak permutation sufficiently. For example, using multi-byte plaintext characters does not make the scheme more secure. It might make cryptanalysis easier, since multi-byte characters introduce additional redundancy that can be exploited in cryptanalysis.

Another dubious idea is to expand the key by dividing a long key and encrypting one part of the key with another part, possibly multiple times with other parts of the key. Obviously, this does not solve any problems mentioned in the previous subsections.

The authors claim: "*The proposed algorithms systematically used complicated text scrambling to secure the message against guessing and brute force.*" [9]. This statement and similar security statements in the original paper are obviously false.

4. Red flags

The red flags are a set of indicators for fast detection of questionable symmetric ciphers proposals. Even thougth some of them are subjective, the goal is to find indicators with the following qualities:

- Easy to evaluate – it must be straightforward to decide whether the indicator is true or false, and in some cases to what degree. No thorough cryptanalytic assessment is expected.

- Related to the quality of the proposal – an indicator must directly show a deficiency or omission in the proposal.

Obviously, the red flags cannot replace a detailed and focused analysis, but they help to highlight potential negligence of proposals' authors. Whether they are a reliable tool for detection of dubious schemes remains an open question for future research. In the rest of the section some indicators are proposed and illustrated on Chessography and the CSS. A summary is provided in Table 2.

Missing reference implementation

Reference implementation clarifies the cipher's inner workings and helps resolve ambiguities in the proposal. It also facilitates easier experimentation with the cipher and its analysis.

There is no implementation provided for Chessography. It is more of an idea than an actual algorithm. The CSS is accompanied by a preliminary implementation, which is provided on GitHub[2] in the form of an IPython notebook.

Unrelated concepts discussed in the proposal

If the goal of a proposal is to introduce a novel symmetric encryption scheme, there is no need to discuss other types of cryptographic schemes in any significant detail. For example, details of public-key cryptography do not meaningfully contribute to the new scheme.

The Chessography proposal introduces only the necessary concepts, encryption and the game of chess. On the other hand, the CSS presents unrelated facts, such as details of the RSA scheme, key lengths of RC2, DES, Blowfish, and RC6 ciphers, inner operations of AES, etc.

Superfluous bibliography

An extensive bibliography is often correlated with the previous indicator when unrelated concepts are referenced. On the other hand, a context, similar schemes, and cryptanalytic assessments require appropriate bibliographic records. In the case of our two schemes Chessography [5] has 7 references, and the CSS [9] has 34 references.

Dubious arguments of cryptographic strength

It should raise a reader's suspicion if the proposal makes security claims without argument or with flawed reasoning. A usual problem is to select a particular feature of the cipher and use it as a token of cryptographic strength. In case of Chessography, it is the chess game tree complexity (the number of possible chess games) that have almost nothing to do with the cryptographic strength of resulting permutations, as we discussed in Section 2. The authors of the CSS use various statistical tests to argue the strength of their proposal. They also informally discuss brute-force resistance, concluding (incorrectly) that large grid size and key size make such an attack ineffective. Surprisingly, both proposals omit an explicit statement of bit-security of the ciphers.

Missing or informal only discussion of modern cryptanalytic attacks

This can be viewed as a continuation of the previous indicator. Any new proposal should contain a justified complexity estimates for state of the art attacks. For stream ciphers, we expect an analysis of algebraic attacks, correlation attacks etc. For block ciphers, variants of differential, linear, integral and other attacks should be considered. In case of iterated ciphers, showing the best attacks on round-reduced versions of the cipher is a plus.

The Chessography proposal lacks any discussion in this regard. The CSS proposal informally and vaguely addresses algebraic attacks and linear cryptanalysis resistance. However, no supporting evidence is provided.

[2] https://github.com/AhmadAbu-Shareha/CSS-Transposition-Cipher

Table 2
Summary of red flags for Chessography and CSS

Red flag	Chessography	CSS
Missing reference implementation	🟥	OK
Unrelated concepts discussed in the proposal	OK	🟥
Superfluous bibliography	OK	🟥
Dubious arguments of cryptographic strength	🟥	🟥
Missing or informal only discussion of modern cryptanalytic attacks	🟥	🟥
Performance comparison with substantially stronger cipher	missing	🟥

Performance comparison with substantially stronger cipher

There are many proposals targeted at IoT, sensors, and similar applications. Constrained environments lead to lightweight schemes. There is nothing wrong with comparing new ciphers to “full” ciphers like AES to see the performance or resource consumption difference. However, it is unfair to not acknowledge the trade-off between security and performance that was made.

Since there is no Chessography implementation, no performance comparison was done. The CSS is extensively compared with “modern encryption algorithms, such as AES”, see [9, Table 9]. Unsurprisingly, results are very favorable for the CSS.

5. Conclusion

This paper shows that Chessography and Cascaded Spin Shuffle are not good proposals, and they seems insecure beyond repair. We hypothesize that these and similar encryption schemes can be recognized by simple indicators (red flags). Certainly, these indicators cannot replace a real cryptanalysis and security assessment, but they raise a bar for acceptable proposals. In future work we would like to apply our indicators to a broader set of symmetric encryption scheme proposals.

Declaration on Generative AI

The author has not employed any Generative AI tools.

References

[1] National Institute of Standards and Technology, NIST SP 800-53, Revision 5: Security and Privacy Controls for Information Systems and Organizations, 2020. URL: https://csrc.nist.gov/pubs/sp/800/53/r5/final.

[2] OWASP Foundation, OWASP Top 10: 2025, 2025. URL: https://owasp.org/www-project-top-ten/, accessed: August 2025.

[3] S. Garfinkel, A Field Guide to Spotting Bad Cryptography, CSO, 2005. URL: https://www.csoonline.com/article/516768/data-protection-a-field-guide-to-spotting-bad-cryptography.html.

[4] A. Manimaran, V. M. Chandrasekaran, A. Gupta, R. Porwal, Encryption and decryption using algebraic chess notations, International Journal of Pharmacy and Technology 8 (2016) 22098–22105.

[5] V. K. Kamat, Chessography: A cryptosystem based on the game of chess, in: H. S. Behera, D. P. Mohapatra (Eds.), Computational Intelligence in Data Mining, Springer Singapore, Singapore, 2017, pp. 309–324. doi:10.1007/978-981-10-3874-7_29.

[6] M. S. Ahmed, P. MaryAnkitha, P. Anitha, M. R. Raju, B. P. Kumar, Chess games as a method for file encryption and storage (2024). doi:10.21203/rs.3.rs-5088828/v1.

[7] M. Singh, A. Kakkar, M. Singh, Image encryption scheme based on knight's tour problem, Procedia Computer Science 70 (2015) 245–250. URL: https://www.sciencedirect.com/science/article/pii/S1877050915032457. doi:10.1016/j.procs.2015.10.081, Proceedings of the 4th International Conference on Eco-friendly Computing and Communication Systems.

[8] Lichess Elite Database, 2024. URL: https://database.nikonoel.fr/.

[9] A. A. Abu-Shareha1, M. Al-Zyoud, Q. Y. Shambour, Cascaded spin shuffle: A transposition cipher using spin motion and grid cascading, International Journal of Intelligent Engineering and Systems 17 (2024) 824–838. doi:10.22266/ijies2024.1231.63.

A. Final positions of chess pieces

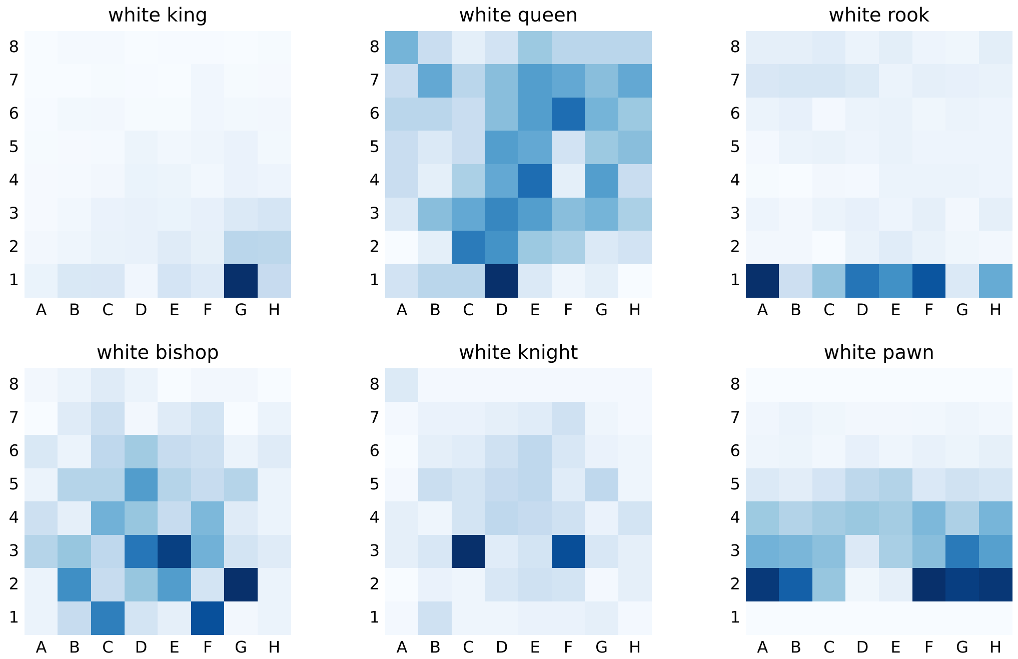


Figure 3: Final positions of white pieces

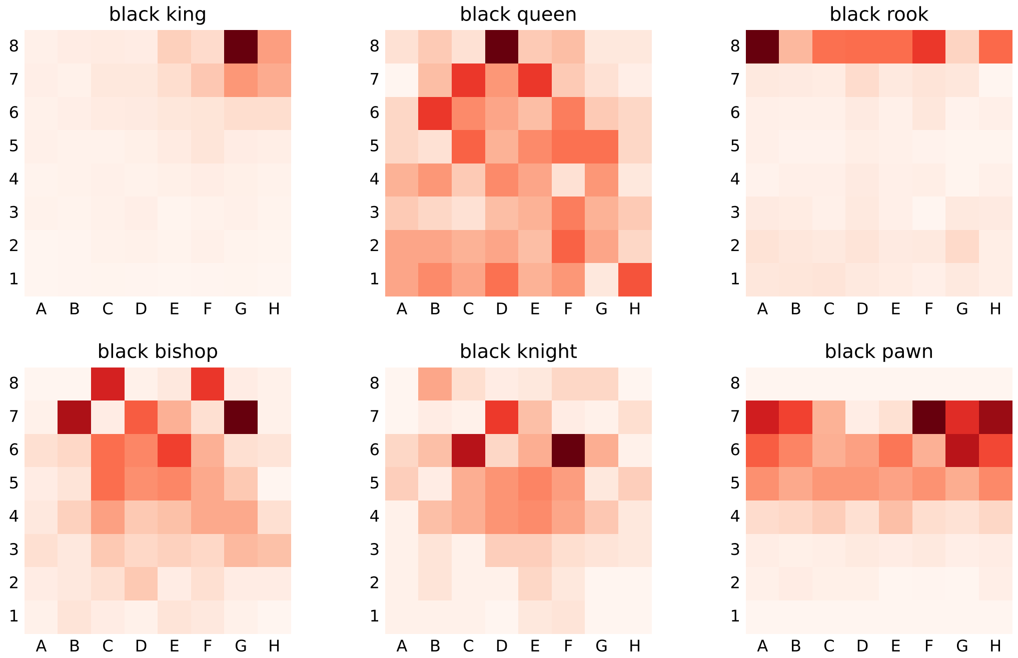


Figure 4: Final positions of black pieces