Decipherment of Lost Ancient Scripts as Combinatorial Optimisation using Coupled Simulated Annealing

Fabio Tamburini
FICLIT, University of Bologna, Italy
fabio.tamburini@unibo.it

Abstract

This paper presents a new approach to the ancient scripts decipherment problem based on combinatorial optimisation and coupled simulated annealing, an advanced non-convex optimisation procedure. Solutions are encoded by using k-permutations allowing for null, one-to-many, and many-to-one mappings between signs. The proposed system is able to produce enhanced results in cognate identification when compared to the state-of-the-art systems on standard evaluation benchmarks used in literature.

1 Introduction

There are still a number of undeciphered scripts in the world and most of them date back thousands of years. The lack of an appropriate amount of inscriptions, the lack of known language descendants written using these scripts or even any certainty whether the symbols actually constitute a writing system made the decipherment of such scripts really challenging. In the Aegean area, for example, we can count at least three syllabic scripts that have not been deciphered yet, namely the Linear A script, Cretan Hieroglyphs and the Cypro-Minoan script. They are scripts strictly connected from a historical point of view, but no one has yet been able to solve these decipherment puzzles. In this work we deal with general decipherment problems, but we are mainly interested in investigating these undeciphered scripts from the Aegean area.

Deciphering an ancient script is, in general, a very complex task; the solution of this problem has been often split into different subproblems in order to obtain specific answers or to simplify the task by decomposing it into simpler problems. In the literature, we can find various contributions dealing with all these subproblems and propose computational methods for solving them in some way, often in relation to one specific script. In order, we have to (a) decide if a set of symbols actually represent a writing system, then (b) we have to devise appropriate procedures to isolate or segment the stream of symbols into a sequence of single signs and then (c) reduce the set of signs to the minimal set for the given writing system forming the alphabet (or syllabary, or whatever inventory of signs), identifying all the allographs. Once we have such a minimal, but complete, set of symbols, we can start (d) assigning to them phonetic/orthographic values and, finally, (e) trying to match phonetic/orthographic transcriptions to a specific language. Here we are interested in studying and discussing steps (d) and (e).

2 Related Works

Any modern attempt to decipher lost scripts using computational tools, a field that has been gaining more and more interest in NLP in the last years (Knight and Sproat, 2009), is based on the comparison of a lost script/language wordlist with words of a known deciphered script/language. These computational approaches have to address two main problems: the first regards the possibility that the two scripts do not correspond. In this case the phonological values of the lost symbols could also be unknown and the matching between the two wordlists must be preceded by some matching between scripts; then, the two wordlists must be matched in some way searching for "cognate" words, i.e. words in different languages that can share an etymological ancestor in a common parent language.

Some scholarly works focus only on cognate detection within the same script (Bouchard-Côté et al., 2009) or directly using the International Phonetic Alphabet sound representations (Hall and Klein, 2010). In both cases the tested languages were typologically very similar.

Conversely, the most advanced recent studies on the automatic decipherment of lost languages proposed systems producing both signs mappings between different scripts and mapping of words into their corresponding cognates (e.g. Snyder et al., 2010; Berg-Kirkpatrick and Klein, 2011; Luo et al., 2019, 2021). These studies share a common view on the computational approach: they structured the algorithm as a two-step procedure, taking inspiration from the Expectation–Maximization (EM) algorithm. The first step proposes a temporary matching between the two "alphabets"[1], and the second step, by relying on the script-matching, tries to match the two word lists proposing possible cognates. At the beginning of the process the scripts matching will be almost random, and so will the cognate matching, but, after several iterations the whole process should converge proposing both a script-matching and a list of possible cognates. The key point hinges on finding an appropriate function, to be optimised by this iterative process, representing in an optimal way the concept of matching between words, including also some linguistic constraints regarding scripts, words and possibly sounds. Let us review the most recent and relevant analyses, in our view, which tackle the decipherment problem in an automatic way, all following the general scheme just discussed.

Snyder et al. (2010) presented the first paper which adopts the modern approach to the computational decipherment problem: their method requires a non-parallel corpus in a known related language and produces both alphabetic mappings and translations of words into their corresponding cognates, employing a non-parametric Bayesian framework to simultaneously capture both low-level sign mappings and high-level morphemic correspondences. They tested this method on Ugaritic, an ancient Semitic language, comparing it with old Hebrew: the model correctly maps 29 of 30 signs to their old Hebrew counterparts, and deduces the correct Hebrew cognate for 60% of the Ugaritic words that have cognates in Old Hebrew.

Berg-Kirkpatrick and Klein (2011) took a different approach: they devised an objective function that, when optimised, yields accurate solutions to both decipherment and cognate pair identification problems. Their system requires only a list of words in both languages as input. The proposed solution is both simple and elegant: binary variables govern both the matching between signs in the two scripts and the matching between the two lexica. By applying an integer combinatorial optimisation procedure, their system was able to obtain good results on the same problem introduced by Snyder et al. (2010) and on a new matching task on Romance languages.

Luo et al. (2019) present a novel neural approach that defines the state-of-the-art for the automatic decipherment of lost languages producing the highest matching performance. To compensate for the lack of strong supervision information, their model is designed to include known patterns in language change documented by historical linguistics. The mapping between signs is carried out by a bidirectional recurrent neural network while the procedure for matching cognates is formalised as a minimum-cost flow problem. They applied this method to the same problem presented in Snyder et al. (2010), a sort of benchmark in this field, and on a brand new dataset that included Linear B and ancient Mycenaean Greek lexica obtaining very good mapping results.

In a subsequent paper by Luo et al. (2021), the authors faced a more difficult hurdle considering scripts that are not fully segmented into words and contexts in which the closest known language is not determined. By building on rich linguistic constraints reflecting consistent patterns in historical sound change, they were able to capture natural phonetic geometry by learning character embeddings based on the International Phonetic Alphabet. The resulting generative framework jointly models word segmentation and cognate alignment, informed by phonetic/phonological constraints. They tested their method on both deciphered languages, namely Gothic and Ugaritic, and on an undeciphered one, Iberian, showing that incorporating phonetic geometry leads to consistent gains.

The two studies from Berg-Kirkpatrick and Klein (2011) and Luo et al. (2019) are the main works with which to compare our proposal.

Berg-Kirkpatrick and Klein (2011) proposed an approach that inspires our work, namely the possibility of tackling the decipherment problem as a pure function optimisation problem, but their results do not represent the state-of-the-art because they have been superseded by subsequent works.

The work from Luo et al. (2019), on the contrary, presents a system able to obtain very good results, but it is not as flexible as we need. With this respect, the mapping between lost and known signs is realised by a recurrent neural network (NN)

[1]With the term "alphabet", we indicate a generic notion of inventory of signs, glyphs, etc. used as a writing system.

and, despite the positive facts that context could be taken into account, it does not allow any further flexibility. In practical decipherment situations we have to face two problems that cannot be easily solved by the approach from Luo et al. (2019): the first regards the fact that very often paleographers have a partial knowledge about the mapping of some signs and this information must be injected into the system and taken into account; the second problem concerns the fact that very often real inscriptions are broken or damaged and some signs cannot be read, thus some kind of uncertainty must be included into the system, for example by using wildcards or other special symbols. These special treatments are very hard to implement into a recurrent NN. Moreover, deep NNs typically require a lot of data in order to be properly trained and this is often not the case in real situations.

As we said before, we are interested in studying some undeciphered scripts from Aegean and we definitely need a more flexible system that allows partial readings and fixed knowledge to be included as well as being able to work on limited amounts of data.

3 Reference Benchmarks and other Datasets

Various datasets have been used in past studies and became standard benchmarks for evaluating the performance of any computational tool aiming at helping scholars in the decipherment process.

3.1 Ugaritic/Old Hebrew - U/OH

Ugaritic is an ancient Semitic language closely related to Old Hebrew. This dataset has been introduced by Snyder et al. (2010) for testing their system and became a common benchmark in the field. Following Berg-Kirkpatrick and Klein (2011), we evaluate our system on 2214 cognates pairs in the two lexica.

3.2 Linear B/Mycenaean Greek - LB/MG

Linear B is a syllabic writing system used to write Mycenaean Greek dating back to around 1450BC. Luo et al. (2019) introduced a new dataset extracting pairs of Linear B and Greek words from a compiled lexicon and removing some uncertain translations obtaining 919 pairs of cognates. This is a very interesting benchmark for us as we are primarily interested in working on syllabic scripts from the Aegean area. On the LB side, we defined the signs inventory as the original set of signs defined in LB while for Greek, given the syllabic nature of the mapping, we included complex signs formed by all open syllables excluding those marking vowel quantity (syllables ending in η or ω) to reduce the signs inventory dimension on the Greek side.

The same authors introduced also a more challenging benchmark, more realistic from the paleographic point of view, considering the same LB lexicon and compare it with a reduced Greek lexicon containing only proper nouns (LB/MG-names).

3.3 Cypriot Syllabary/Arcadocypriot Greek - CS/AG

Given our primary interests, it seemed reasonable to introduce a new dataset to be used as reference for the decipherment of syllabic scripts from the same area. The Cypriot Syllabary is a right-to-left syllabic script used in Iron Age Cyprus. It is descended from the Cypro-Minoan syllabary, in turn, a derivative of Linear A. Most texts using this script are in the Arcadocypriot dialect of Greek.

Relying on the alphabetic-syllabic index in Hintze (1993), we compiled a new dataset consisting of 693 pairs of cognates, the first written using the CS and the second the Greek alphabet from which we removed any diacritic following the same procedure applied in Luo et al. (2019) for creating the LB/MG dataset. With regard to Greek, we considered only the open syllables as for the previous dataset.

4 The Proposed Method

The proposed approach to the decipherment problem is configured as a global optimisation procedure taking inspiration from the work proposed in Berg-Kirkpatrick and Klein (2011). We will introduce a flexible encoding of possible solutions and an ‘energy function’ able to capture the goodness of a single solution, both from the point of view of signs matching and lexica matchings; by minimising the energy function, we will search for suitable solutions to a decipherment problem.

Let us introduce some notation useful in the next sections: L_s and K_s are two linearly ordered sets[2] containing respectively the signs in the lost and known languages (with $|L_s|$ and $|K_s|$ their cardinality and l^i, k^j respectively the i-th and j-th element in the ordered sets), while L_{lex} and K_{lex} are

[2]A linearly ordered set is a set with a total order on it. Here, it is useful only for indexing the set elements.

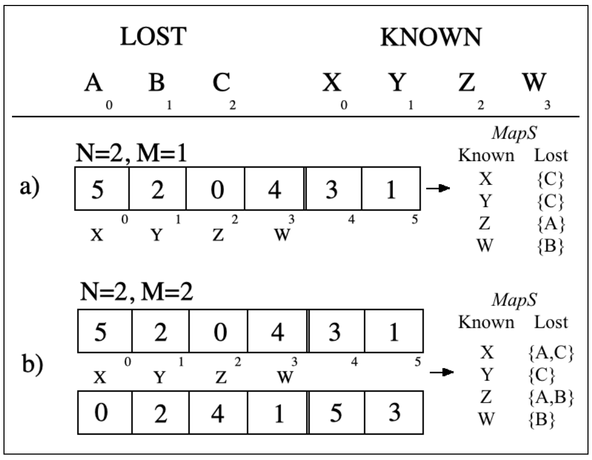

Figure 1: Two simple examples of solution coding. a) $M = 1$ then the first $|K_s|$ cells contain the mapping $MapS$ (shown on the right) for the known signs to the lost signs. Note that using k-permutations of size $N \cdot |L_s|$ allows for one-to-many mappings from lost to known signs (see also the definition $MapS$ in 4.1). b) $M = 2$, then we have two k-permutations allowing for one-to-many assignments from known to lost signs. In both cases it could happen that a lost sign does not receive any assignment (not shown in the picture).

the two lexica and $|L_{lex}|$ and $|K_{lex}|$ their respective number of words.

4.1 Solution Coding

The basic tool for encoding a problem solution is the k-permutation without repetition. Let $p_1, ..., p_n$ be n objects. Let s_1,...,s_k be k (where $k \leq n$) slots to which k of the n objects can be assigned. A k-permutation of n objects is one of the possible ways to choose k objects and place them into the k slots respecting the order. Each object can be chosen only once. The number of possible k-permutations is $P_{n,k} = n!/(n-k)!$. Here we consider the k-permutation of the first n integer numbers.

In order to find a suitable sign assignment between lost language and known language, a generic solution σ must have the possibility to express multiple assignments in both directions but paying attention to the combinatorial explosion problem.

Let us start considering the case $|L_s| \leq |K_s|$: in this situation some lost signs must be mapped to more than one known sign and we can easily encode this fact with a single k-permutation σ with $n = N \cdot |L_s|$ and $k = |K_s|$, $N = 2, 3, ...$ Each known sign k_j being in position j, with $j \leq k$, of the k-permutation $\sigma = \langle \sigma_1, ..., \sigma_k, ..., \sigma_n \rangle$ is then mapped to a set of lost signs by the function

$$MapS^{\sigma}\colon K_s \rightarrow \mathcal{P}(L_s),$$

$$MapS^{\sigma}(k_j) = l_{\sigma_j \bmod |L_s|}$$

where $\mathcal{P}(L_s)$ is the power set of L_s.

In the other case with $|L_s| > |K_s|$ we can define a solution σ formed by M k-permutations, $M = 2, 3, ...$, concatenated one after the others, each managed exactly in the same way as before, but now N can also be equal to 1.

By defining the structure of possible solutions σ in this way, each sign in the lost language can receive from 0 to a maximum of $N \times M$ possible assignments of known signs, allowing a very high level of flexibility in signs matching. Given that in the definition of k-permutations $k \leq n$, N controls the well-formedness of the basic structure supporting solution definition ensuring that every known sign will be assigned to at least one lost sign and managing the different situations that occur when $|L_s| \leq |K_s|$. Moreover, M controls the number of times a known sign will be assigned to a lost sign. N and M are not completely independent parameters as they interact in a complex way for governing the number of multiple assignments in both directions.

Figure 1 shows two small examples of the proposed schema for encoding solutions.

As an added value, k-permutations exhibit an interesting property: we can build an isomorphism between k-permutations and the natural numbers (Patel, 2022), thus each solution encoded by using our schema can be mapped into M integers and, for reasonable problems with $M \leq 2$, fragments of the search space can be visualised and inspected using a 2D/3D graph.

4.2 Energy function

The second fundamental ingredient used in the proposed method regards the design of an appropriate energy function able to measure the goodness of a given solution for a decipherment problem.

As we said before, Luo et al. (2019) broke the optimisation process into two separate steps repeated iteratively: the first computes the best match between signs given a lexicon match and, after having fixed the signs match, the second computes the best match between lexica. We adopted a different approach taking inspiration from Berg-Kirkpatrick and Klein (2011) and designed an energy function that measures the goodness of both aspects together.

4.2.1 Lost Words Expansion and Transliteration

In order to transliterate the lost lexicon we need to define the inverse function of $MapS$, $invMapS^\sigma\colon L_s \to \mathcal{P}(K_s)$, that associates each lost sign to the set of known signs mapped to it, as

$$invMapS^\sigma(l_i) = \{k_j | l_i \in MapS^\sigma(k_j)\}\ .$$

By building on this definition, we can introduce the transliteration and expansion function $TrExp^\sigma$ for a given lost word $lW = \langle lW_1, ..., lW_n\rangle$, with $lW_1, ..., lW_n$ the sequence of signs forming lW, as

$$TrExp^\sigma(lW) = \{\ tW \mid \quad tW = \langle q_1, ..., q_n\rangle, \\ q_j \in invMapS^\sigma(lW_j)\ \}.$$

$TrExp$ transliterates each lost word into the known alphabet and associates to it a set of transliterated words formed by any combination of known signs allowed by the mapping $invMapS$. This way of proceeding could potentially produce a combinatorial explosion, but, given that N and M are typically very small integers (almost always ≤ 3), this problem will not be particularly severe. Table 1 shows an example of this process.

l_i	$invMapS^\sigma(l_i)$	lW	$TrExp^\sigma(lW)$
A	{Z,X}	AA	{ZZ,ZX,XZ,XX}
B	{W,Z}	BC	{WX,WY,ZX,ZY}
C	{X,Y}	ABC	{ZWX,ZWY,ZZX, ZZY,XWX,XWY, XZX,XZY}

Table 1: Transliteration and expansion example over the same sets of signs used in Figure 1 (b).

4.2.2 Word Matching

A standard way to compare strings makes use of the so called *edit distance* - ED (a.k.a. Levenshtein distance). We used this measure to compare the expanded transliterations of lost words to known words. The standard definition of the ED involves counting the number of sign insertions, deletions and substitutions to transform the first string into the second. We modified the standard definition, following the ideas in Wang et al. (2021), for adding two wildcards that could be very useful in real settings. Very often real inscriptions are broken and/or some signs cannot be reliably distinguished; in these situation it might be preferable to process these data maintaining the reading problems. For these reasons, we included the special sign '?' to indicate a single unreadable sign and '*' to indicate multiple unreadable signs both allowed only in lost words. Let $X = \langle x_1, ...x_n\rangle$ and $Y = \langle y_1, ...y_m\rangle$ two words to be compared, with n and m their respective lengths, then the ED with wildcards used in this study, $EDW_{X,Y}(n,m)$, has been defined as in Figure 2.

The edit distance in general, and also our variation including wildcards, does not take into account word lengths and it is not suitable for comparing the distance between sets of words. For this reason, most studies introduced a kind of normalisation for ED values. Given the interesting properties (Fisman et al., 2022) of the *Generalised Edit Distance* proposed by Li and Liu (2007)[3], we normalised EDW as

$$\overline{EDW}_{X,Y} = \frac{2 \cdot EDW_{X,Y}}{|X| + |Y| + EDW_{X,Y}}$$

where $|\cdot|$ represents the word length.

We used $\overline{EDW}$ to compare the transliterated and expanded lost lexicon, created by applying the $TrExp$ function to every word in L_{lex}, with the known words in K_{lex} (see next Section).

We implemented the $\overline{EDW}$ function in a fast code that also works on GPUs[4].

4.2.3 Lexica Matching

The datasets presented in Section 3, as produced by the cited works, associate each lost word with one or more cognates in the known language. In order to adhere to this view and to perform a correct evaluation, we introduced a specific variant of the standard Linear Sum Assignment - LSA - problem (a.k.a. Hungarian algorithm) for matching lexica: instead of matching single words, we match groups of words both on the lost side and on the known side. On the lost language side, this accounts for different transliteration of the same lost word due to multiple assignments to the same lost sign (see the definitions of functions $invMapS$ and $TrExp$), while, on the known language side, this accounts for the sets of cognates considered in the cited benchmarks.

In order to introduce our modified version of the LSA algorithm, let us define a partition

[3]Generalised Edit Distance is a metric, its upper bound is 1 and it does not escalate repetitions remaining simple and quick to calculate.

[4]https://github.com/ftamburin/EditDistanceWild

$$EDW_{X,Y}(i,j) = \begin{cases} \max(i,j), & min(i,j) = 0 \\ \min \begin{cases} EDW_{X,Y}(i-1,j) + wD \\ EDW_{X,Y}(i,j-1) + wI \\ EDW_{X,Y}(i-1,j-1) + S(x_i,y_j) \cdot wS \end{cases} & min(i,j) \neq 0, x_i \neq \text{`*'} \\ \min \begin{cases} EDW_{X,Y}(i-1,j) \\ EDW_{X,Y}(i,j-1) \\ EDW_{X,Y}(i-1,j-1) \end{cases} & min(i,j) \neq 0, x_i = \text{`*'} \end{cases}$$

$$S(x,y) = \begin{cases} 0 & x = y \text{ or } x = \text{`?'} \\ 1 & \text{otherwise} \end{cases}$$

Figure 2: Edit Distance with Wildcards definition. wD, wI and wS represents the weight penalisations respectively for sign deletion, insertion and substitution and, for this study, they have been all fixed to 1.

$K_{lexG} = K^1_{lexG}, ..., K^G_{lexG}$ of K_{lex} where K^j_{lexG} represents a set of known cognates in the dataset; we can then introduce the variables $A_{i,j} \in \{0,1\}$ representing the lexica alignment obtained by the LSA algorithm (with $A_{i,j} = 1$ iff lW^i is assigned to K^j_{lexG}), configure the LSA problem to be solved as

$$\min \sum_{i=1}^{|L_{lex}|} \sum_{j=1}^{|K_{lexG}|} A_{i,j} \cdot \left[\min_{\substack{X \in TrExp^{\sigma}(lW^i) \\ Y \in K^j_{lexG}}} \overline{EDW}_{X,Y} \right]$$

$$\text{s.t.} \quad \sum_i A_{i,j} = 1, \quad j = 1,2,...,|K_{lexG}|$$
$$\sum_j A_{i,j} = 1, \quad i = 1,2,...,|L_{lex}|$$

and, once solved the LSA and fixed the values for the As matching variables, we can define the Energy function E for a given problem solution σ as

$$E(\sigma) = \sum_{i=1}^{|L_{lex}|} \sum_{j=1}^{|K_{lexG}|} A_{i,j} \cdot \left[\min_{\substack{X \in TrExp^{\sigma}(lW^i) \\ Y \in K^j_{lexG}}} \overline{EDW}_{X,Y} \right] \tag{1}$$

See Figure 3 for an example of the lexica matching process.

It seems important to note that the computation of the energy function E for a given solution σ strictly derives from the solution itself, first by converting the solution coding into signs assignments by using the function $TrExp$ and then matching the two lexica by the LSA procedure described above.

4.2.4 Penalty factors

In order to regularise the entire process and help the optimisation procedure to find reliable solutions, we introduced some regularisation factors into the energy function E. Given that our method relies on a flexible assignment schema allowing no assignments to lost signs and multiple assignments of known signs, we have to guarantee that the optimisation procedure does not abuse of these instruments. In general, no assignments to lost signs rarely produces a good solution as well as exaggerating in including multiple assignments of known signs. In order to discourage solutions with these characteristics, we introduced two penalisation factors: if we define $\#UA(\sigma)$ the number of lost signs without any assignment and $\#MA(\sigma)$ the number of known signs with multiple assignments for a given solution σ, then the final energy function to be minimised is

$$E'(\sigma) = E(\sigma) + \lambda \cdot (\#UA(\sigma) + \#MA(\sigma)) \,. \tag{2}$$

To strongly discourage these potentially degenerate solutions we set $\lambda = 4$.

4.3 Energy Optimisation using Coupled Simulated Annealing

Having configured our problem as a general global optimisation procedure led us to minimise the energy function E' defined before by using any metaheuristic proposed in the literature, e.g. tabu-search, genetic and evolutionary methods, ant colony optimisation, simulated annealing, etc.

Coupled Simulated Annealing - CSA (de Souza et al., 2010) is a method for global optimisation

L_{lex}		K_{lex}				
lW	$TrExp$	WX	WW	XX	XWY	XWZ
AA	ZZ	2	2	2	3	2
	ZX	$\boxed{1}$	2	1	3	3
	XZ	2	2	1	2	$\boxed{1}$
	XX	1	2	$\boxed{0}$	2	2
BC	WX	$\boxed{0}$	1	$\boxed{1}$	2	2
	WY	1	1	2	$\boxed{1}$	2
	ZX	1	2	1	3	3
	ZY	2	2	2	2	3
ABC	ZWX	$\boxed{1}$	2	2	2	2
	ZWY	2	2	3	1	2
	ZZX	2	3	2	3	3
	ZZY	3	3	3	2	3
	XWX	1	2	$\boxed{1}$	1	1
	XWY	2	2	2	$\boxed{0}$	1
	XZX	2	3	1	2	2
	XZY	3	3	2	1	2

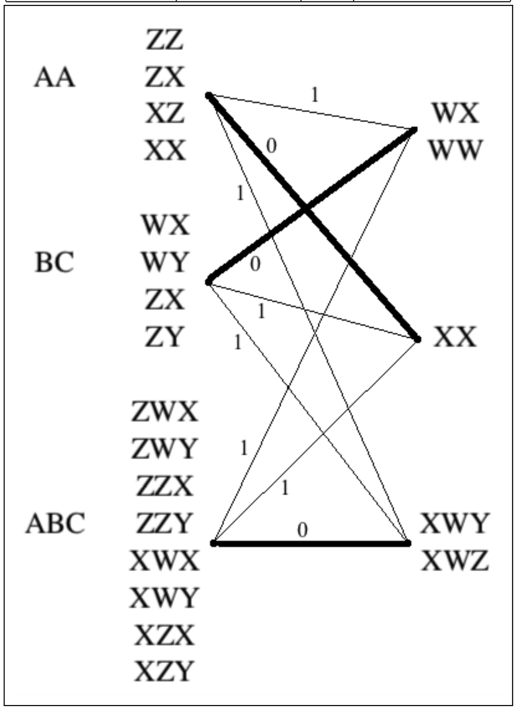

Figure 3: A simple example of the lexica matching process. The lost lexicon is identical to that in Table 1, while the known lexicon is formed by five words grouped into three set of cognates. On the top, we have the cost matrix computed by using the edit distance (we did not use the normalised version for readability) and the values surrounded by a box indicate the minimum considering two respective groups. At the bottom, these minimal values represent the costs for a LSA problem that finds the min-cost matching (thick lines) between the two lexica respecting the groupings in the lost and known lexica.

based on Simulated Annealing (SA). CSA is characterised by a set of parallel standard SA processes (with $\#Anns$ defining the number of annealers) coupled by their acceptance probabilities. The coupling is performed by a term in the acceptance probability function that is a function of the energies of the current states of all SA processes creating a cooperative behaviour via information exchange between the parallel annealing processes. Coupling can also provide information that can be used to drive the overall optimisation process towards the global optimum. The authors of the original work present a system able to use the acceptance temperature to control the variance of the acceptance probabilities with a simple control scheme (called 'CSA-MwVC' in the original paper). This leads to a much better optimisation efficiency because it reduces the sensitivity of the algorithm to initialisation parameters while guiding the optimisation process towards quasi-optimal states.

After some attempt with other techniques, we decided to adopt CSA mainly for two reasons: (a) it is a method that can be easily parallelised on a multicore CPU allowing for heavily parallel computations with a minimal exchange of information and (b) the control mechanism over the variance of the acceptance probabilities automatically governs the annealing process avoiding the introduction of complex annealing schemas that often have to be tuned for a specific dataset.

For the implementation of CSA we relied on a code specifically developed for problems based on permutations[5] configuring it to employ 16 parallel annealers.

The generic SA algorithm is quite simple: given a solution, we have to perturb it obtaining a new solution in its neighbourhood that is accepted, or not, depending on a stochastic decision based on the new solution energy and the global current system temperature. Selecting a neighbouring solution perturbing the current is a delicate step as we have to ensure an appropriate sampling of the solution space. Luckily, Tian et al. (1999) made an in depth study regarding the most promising 'moves' for solutions based on permutations and the swapping of two items in the permutation is considered the best move for assignment problems. In order to help the system to avoid getting stuck in a local minimum, we further introduced a random k-swap perturbation with probability 0.1 with k decaying

[5]https://github.com/structurely/csa.

Algorithm 1 *CSA_OptMatcher*

```
Data: $L_s, K_s, L_{lex}, K_{lex}, N, M, \#Anns$
Result: the optimised solution $best\_\sigma$
• Init one solution $\sigma_j$ for each annealer $j$
• Init annealing and generation temperatures
  $T_a$ and $T_g$
for $iter = 1, \ldots$ do
    • Generate $\#Anns$ perturbed solutions $\sigma'_j$
      by swapping two indices in each of them
    • Compute $E'(\sigma'_j), j = 1, ..., \#Anns$ using
      equations (1) and (2)
    for $j = 1, ..., \#Anns$ do
        if $E'(\sigma'_j) \leq E'(\sigma_j)$ then
            • $\sigma_j = \sigma'_j$
        else
            • Accept $\sigma'_j$ following the
              CSA-MwVC algorithm
        end if
    end for
    • Decrease $T_a$ and $T_g$ according to
      CSA-MwVC temperature schedules
    • $best\_\sigma = min_j\ E'(\sigma_j)$
end for
```

with the generation temperature governed by the CSA schedule.

See Algorithm 1 for a general picture of the entire optimisation process.

4.4 Evaluation

With regard to evaluation, we stick to the same procedure introduced in previous literature and, in particular, in Luo et al. (2019) and measured the system Accuracy in finding pairs of lost and known cognates as listed in the considered dataset.

The influential paper from Reimers and Gurevych (2017) makes clear to the community that reporting a single score for each session could be heavily affected by the system random initialisation and we should instead report the mean and standard deviation of various runs, with the same setting, in order to get a more accurate picture of the real systems performance and make more reliable comparisons between them. For these reasons, any new result proposed in this paper is presented as the mean and standard deviation of system Accuracy over 4 runs with different random initialisations. In this way, we should give a real picture of our system performances.

5 Results

The two parameters N and M for solution shaping described in Section 4.1 could be considered hyperparameters for the proposed method as they can give more power to the possible solutions at the price of more parameters to be fixed and thus slower convergence. We decided to avoid any optimisation of these parameters in our experiments and fix them using a very simple rule: $N = 1$, $M = 2$ if $|L_s| > |K_s|$ and $N = 2$, $M = 1$ otherwise.

Table 2 shows the results of our experiments compared with the reference literature. Our system is able to produce better Accuracy than any other work on all considered benchmark datasets with a large margin. If we consider the fact that our results are presented as the mean and std. deviation of more runs and not as the maximum Accuracy achieved by the system, the results are even more relevant.

6 Discussion and Conclusions

We presented a new approach to the ancient scripts decipherment able to produce very good results in cognate identification w.r.t. the state-of-the-art. All the hyperparameters were not optimised at all and it seems reasonable that increasing N and/or M even better results can be reached. We plan to perform more experiments in that direction.

Another system feature worth of mention regards its ability to converge to reasonable solution for any simulation; even during the development phase, the proposed system never got trapped into very poor sub-optimal solutions. Simulations took a relevant time to converge, but they always converged without any need to restart the process, a common technique for this kind of methods (see e.g. Berg-Kirkpatrick and Klein 2013), confirming the strength of CSA as a function optimisation technique.

There are other approaches to the problem in the literature that we have not explicitly discussed in Section 2 because not strictly devoted to the decipherment of ancient scripts, but address the problem of deciphering substitution or homophonic codes like the famous Zodiac-408 cipher or the Beale cipher (e.g. Ravi and Knight, 2011; Nuhn et al., 2013, 2014). For example Ravi and Knight (2011) proposed a stochastic model taking into account both token n-grams and dictionaries. Knowing for certain the target language, they can esti-

System	**Benchmark Dataset**			
	U/OH	LB/MG	LB/MG names	CS/AG
(Berg-Kirkpatrick and Klein, 2011)	90.4	-	-	-
(Luo et al., 2019)	93.5	84.7	67.3	-
This work (*CSA_OptMatcher*)	**95.5**±0.83 max **96.3**	**89.4**±1.81 max **91.0**	**83.4**±2.50 max **87.0**	**86.3**±1.73 max **87.9**

Table 2: Accuracy results in cognate identification compared with the reference literature.

mate a language model (LM) using a large set of data, even artificially generated, and can take advantage of complete lexica and frequency information for the known language. Unfortunately, when using these methods to solve the decipherment problems on ancient languages often the target language is not known for certain, maybe it is a language from the same area sharing the same data scarcity as the lost one and thus it is not possible to build useful LMs or rely on a complete dictionary. On the contrary, everything is only partially known or unreliable: phonetic values, signs mappings, frequency information and the true underlying language. This facts make it very difficult to use methods like the one proposed by these authors.

Our very promising results in the decipherment of ancient scripts might suggest that these tools can solve all the unsolved problems, of palaeographic, epigraphic and linguistic nature, debated for years by experts. This is naturally not the case. These techniques, even if very promising, also present a large number of problems when applied in real decipherment attempts: (a) first of all, segmented and clean corpora are needed. Building a corpus for an ancient undeciphered script, even in the case where we have already solved the segmentation problems and were able to collect single sign images and sign/word sequences, is not an easy task. Most inscriptions are damaged and many signs are not readable. Broken words and/or partial sentences are also frequent; (b) an extensive cognate list must be available, but in most real cases we only have two word lists that must be matched without any guarantee that lost language cognates are really present in the known language lexicon; (c) in NLP we have to make evaluation on well-known test beds and all the studies we discussed before worked on well known correspondences to prove the system effectiveness. It is an entirely different matter to test the same systems on real cases when we have to deal with unknown writing systems and their corresponding languages.

In the light of these considerations, we agree with Sproat (2020) who suggested that these tools can help paleographers shed light on the decipherment process, but we cannot rely on them only for providing a complete solution to our real problems without any human intervention for guiding the process and interpreting the results. However, our future plans regard the application of the proposed system to undeciphered scripts from the Aegean area, hoping to shed some light on problems unresolved for centuries.

Codes and all datasets for reproducing the experiments are available on github[6].

Acknowledgements

We would like to thank Silvia Ferrara for her help in defining Ancient Greek syllables and the reviewers for their valuable suggestions for improving the final version of the manuscript.

References

Taylor Berg-Kirkpatrick and Dan Klein. 2011. Simple effective decipherment via combinatorial optimization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 313–321, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick and Dan Klein. 2013. Decipherment with a million random restarts. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 874–878, Seattle, Washington, USA. Association for Computational Linguistics.

Alexandre Bouchard-Côté, Thomas L. Griffiths, and Dan Klein. 2009. Improved reconstruction of protolanguage word forms. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 65–73,

[6]https://github.com/ftamburin/CSA_OptMatcher

Boulder, Colorado. Association for Computational Linguistics.

Samuel Xavier de Souza, Johan A. K. Suykens, Joos Vandewalle, and Désiré Bollé. 2010. Coupled simulated annealing. *IEEE Trans. Syst. Man Cybern. Part B*, 40(2):320–335.

Dana Fisman, Joshua Grogin, Oded Margalit, and Gera Weiss. 2022. The normalized edit distance with uniform operation costs is a metric. In *33rd Annual Symposium on Combinatorial Pattern Matching, CPM 2022, June 27-29, 2022, Prague, Czech Republic*, volume 223 of *LIPIcs*, pages 17:1–17:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

David Hall and Dan Klein. 2010. Finding cognate groups using phylogenies. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1030–1039, Uppsala, Sweden. Association for Computational Linguistics.

Almut Hintze. 1993. *A lexicon to the Cyprian syllabic inscriptions*. Buske, Hamburg.

Kevin Knight and Richard Sproat. 2009. Writing systems, transliteration and decipherment. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Tutorial Abstracts*, pages 15–16, Boulder, Colorado. Association for Computational Linguistics.

Yujian Li and Bo Liu. 2007. A normalized levenshtein distance metric. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1091–1095.

Jiaming Luo, Yuan Cao, and Regina Barzilay. 2019. Neural decipherment via minimum-cost flow: From Ugaritic to Linear B. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3146–3155, Florence, Italy. Association for Computational Linguistics.

Jiaming Luo, Frederik Hartmann, Enrico Santus, Regina Barzilay, and Yuan Cao. 2021. Deciphering undersegmented ancient scripts using phonetic prior. *Transactions of the Association for Computational Linguistics*, 9:69–81.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2013. Beam search for solving substitution ciphers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1568–1576, Sofia, Bulgaria.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2014. Improved decipherment of homophonic ciphers. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1764–1768, Doha, Qatar.

Deepesh Patel. 2022. Generating the nth lexicographical element of a mathematical k-permutation using permutational number system. *SSRN*, http://dx.doi.org/10.2139/ssrn.4174035.

Sujith Ravi and Kevin Knight. 2011. Bayesian inference for zodiac and other homophonic ciphers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 239–247, Portland, Oregon, USA.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark. ACL.

Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1048–1057, Uppsala, Sweden. Association for Computational Linguistics.

Richard William Sproat. 2020. Translating lost languages using machine learning? Wellformedness, http://www.wellformedness.com/blog/translating-lost-languages-machine-learning/.

Peng Tian, Jian Ma, and Dong-Mo Zhang. 1999. Application of the simulated annealing algorithm to the combinatorial optimisation problem with permutation property: An investigation of generation mechanism. *European Journal of Operational Research*, 118(1):81–94.

Yuanhao Wang, Qiong Huang, Hongbo Li, Meiyan Xiao, Huang Jianye, and Guomin Yang. 2021. Public key encryption with fuzzy matching. In *Provable and Practical Security, LNCS 13059*, pages 39–62. Springer.