TRANSFORMATICS 101 - explained

Joseph Willrich Lutalo*
Nuchwezi Research
joewillrich@gmail.com, jwl@nuchwezi.com

EDITION: $\mathbf{8}^{th}$ **OCT**, 2025

Abstract

In only **10 general foundational results**[1][2] — all **based on earlier mathematical research** by the author, as well as **10 extra abstractions** spanning **how transformatics might be leveraged** as **a basis mathematics** `for problems` in; `number theory` and `cryptography` (o-SSIs); `mathematical statistics` (ADM, TCR, PCR, MSS, SC, PC); `linear algebra` (super sequences); `computer science` (TEA language, transformer-chains/sequence-transformers) and `information theory` (entropy of a sequence) among others, we distill and establish, in a summarized form, the **core foundations of Transformatics**[3] as its own mathematical theory, branch, or **discipline**. In this **mini-thesis**. Each foundational result presented is likewise accompanied by one or **more references** to some earlier work where the idea or presented concept was first developed, presented, or applied by the author during their research. **Useful notes and commentary** are also included where necessary, and in the extension decade of **proposed abstractions**, also commentary about how the proposals relate to the decade of foundational results likewise is catered for.

Keywords: Foundations, Transformatics, Sequence Analysis, Sequence Transformers, Thesis

1 Foundations of a New Mathematics of Sequences

1.1 Result 1: The Empty Sequence[4]

$$\Theta = \langle\rangle \implies |\Theta| = 0 \quad \wedge \quad \Theta \equiv \emptyset$$

1.2 Result 2: A Symbol Set[5]

$$\psi_\beta^n = \langle \beta_{i\in[1,n]} \rangle : n \in \mathbb{N} : \mathbb{u}(\beta_i \in \psi_\beta) = 1 \quad \forall \beta_i \in \psi_\beta$$

1.3 Result 3: A Sequence[4]

$$\Theta^n = \langle \theta_{i\in[1,n]} \rangle : \mathbb{N} \times \psi_\beta : n \geq 1$$

*Principal Investigator at Nuchwezi Research, GARUGA, Uganda.
ORCID: `https://orcid.org/0000-0002-0002-4657`

1.4 Result 4: Sequence Cardinality[6][5]

$$\forall \Theta^n = \langle \theta_{i \in [1,n]} \rangle : \mathbb{N} \times \psi_\beta \quad \implies \quad \mathit{u}(\Theta^n) = |\Theta^n| = n \in \mathbb{N}$$

1.5 Result 5: A Sequence Symbol Set[7][5]

$$\psi(\Theta^n) = \langle \theta_{i \in [1,k]} \rangle : k, n \in \mathbb{N} \wedge k \geq 1 : \mathit{u}(\theta_i \in \psi(\Theta^n)) = 1 \quad \forall \theta_i \in \Theta^n : k \leq n : \mathit{u}(\psi(\Theta^n)) = k$$

1.6 Result 6: A Sequence Transformation[3][4]

$$\Theta^n = \langle \theta_{i \in [1,n]} \rangle : \mathbb{N} \times \psi_\beta \rightarrow \Theta^* = \langle \theta^*_{i \in [1,k]} \rangle : \mathbb{N} \times \psi_* : \psi_\beta \neq \psi_* \vee \psi(\Theta^n) \neq \psi(\Theta^*)$$

1.7 Result 7: A Sequence Transformer[8][3]

$$\Theta^n = \langle \theta_{i \in [1,n]} \rangle : \mathbb{N} \times \psi_\beta \xrightarrow{f(\Theta) = \Theta^*} \Theta^* = \langle \theta^*_{i \in [1,k]} \rangle : \mathbb{N} \times \psi_*;$$
$$\psi_\beta \neq \psi_* \vee \psi(\Theta^n) \neq \psi(\Theta^*)$$

1.8 Result 8: A Sequence Filter[3]

$$\Theta^n = \langle \theta_{i \in [1,n]} \rangle : \mathbb{N} \times \psi_\beta \xrightarrow{f(\Theta, \psi_{\beta^*}) = \Theta^*} \Theta^* = \langle \theta^*_{i \in [1,k]} \rangle : \mathbb{N} \times \psi_{\beta^*};$$
$$k \in \mathbb{N} : \psi_{\beta^*} \subseteq \psi_\beta \implies \mathit{u}(\psi_{\beta^*}) = k \leq \mathit{u}(\psi_\beta) \implies \mathit{u}(\Theta^*) \leq \mathit{u}(\Theta^n)$$

1.9 Result 9: A Sequence Generator[3][9]

$$\Theta^n | \emptyset \xrightarrow{f(\psi_\beta, k)) = \Theta^*} \Theta^k = \langle \beta^*_{i \in [1,k]} \rangle : \mathbb{N} \times \psi_\beta : k \in \mathbb{N} : \forall \beta_i \in \Theta^k \implies \beta_i \in \psi_\beta \quad \wedge \quad k \geq 1$$

1.10 Result 10: A Sequence Sampler[10]

$$\Theta^n \xrightarrow{f(\Theta, k)) = \Theta^k} \Theta^k = \langle \theta_{i \in [1,k]} \rangle : \mathbb{N} \times \psi(\Theta^n) : k, n \in \mathbb{N} \wedge k \geq 1$$

2 Proposed Abstractions Leveraging Transformatics

Before we kick-off, note that, as in many previous works on transformatics since [3], when we have any sequence $\Theta^n : \forall i \in [1, n] \quad \exists \theta_i \in \Theta^n$, then, when we wish to talk of the **position index** of some member element or symbol θ_i in Θ^n, such that we mean the member in that sequence that is located or positioned at exactly index i in the range $[1, n]$, then we shall, as introduced and elaborated upon in the introduction section of [3], shall merely use the notation:

$$I(\theta_i, \Theta^n) = i \quad \forall \theta_i \in \Theta^n \tag{1}$$

Further, if we have any two elements from the sequence (whether or not they are equal: $\theta_i = \theta_j$ or not: $\theta_i \neq \theta_j$), and yet, if $i \neq j$, then $I(\theta_i, \Theta^n) \neq I(\theta_j, \Theta^n)$, and if $i \leq j$, then $I(\theta_i, \Theta^n) \leq I(\theta_j, \Theta^n)$.

As you shall come to appreciate when studying formulations and expressions in transformatics, the position operator or measure, $I(\cdot, \cdot)$ is very fundamental, useful and indispensable especially when treating of **ordered sequences** or of order in any sequence.

2.1 Proposal 1: The o-SSI Sequence from any Sequence[7]

Assume we consider the sequence $\Theta : \mathbb{N} \times \psi_*$ of any length and spanning an arbitrary symbol set ψ_*. Then, in case we wish to reduce Θ to a representative sequence Θ_{ossi} that only contains elements of Θ spanning the symbol set, ψ_β, of some particular base, β; in which case, Θ_{ossi} would be an **orthogonal symbol set identity** (o-SSI) sequence under base-β, then, the following o-SSI-generator — also **an extractor**[1] and **a reducer** transformer, defined using the calculus and semantics of transformatics, would suffice[2]:

Transformer 1 (The **o-SSI Extractor**). $\Theta\langle\theta_{i\in[1,n]}\rangle : \mathbb{N} \times \psi_* \xrightarrow{O_{filter}(\Theta,\psi_\beta)} \Theta^k\langle\theta_i\rangle : \mathbb{N} \times \psi_\beta = \Theta_{ossi}$;
$0 \leq k \leq \text{ɯ}(\psi_\beta) \leq \text{ɯ}(\Theta) = n \in \mathbb{N} : \forall\theta_i \in \Theta^k \quad \exists\theta_i \in \psi_\beta$
$\wedge \quad \forall i, j \in \mathbb{N} \wedge \theta_i, \theta_j \in \Theta^k : I(\theta_i, \Theta) \leq I(\theta_j, \Theta) \implies I(\theta_i, \Theta^k) \leq I(\theta_j, \Theta^k)$

Note that, the **o-SSI Sequence**, Θ_{ossi}, of any sequence Θ under β would likewise be related to the concept of the **Specific Symbol Set**[3], $\hat{\psi}(\Theta)$, of that sequence, in the sense that, at most, Θ_{ossi} is equivalent to $\hat{\psi}_\beta(\Theta)$ or that their cardinalities and memberships are related as:

$$\text{ɯ}(\Theta_{ossi}) \leq \text{ɯ}(\hat{\psi}_\beta(\Theta)) \leq \text{ɯ}(\psi_\beta) \leq \text{ɯ}(\Theta) \tag{2}$$

and as for membership:

$$\psi(\Theta_{ossi}) \subseteq \hat{\psi}_\beta(\Theta) \subseteq \psi_\beta \subseteq \Theta \tag{3}$$

2.2 Proposal 2: The Anagram Distance Between Any Two Sequences[6]

Occasionally, one shall find that they need to quantify how different some two or more sequences are. In the simplest case, we might merely sort a collection of sequence by the relative size of each sequence, and thus by their sequence cardinality for example — $\text{ɯ}(\Theta)$ Vs $\text{ɯ}(\Theta^*)$ for some two sequences involved in a general transformation such as:

Transformation 1. $\Theta \to \Theta^* \quad \implies \quad \nabla_{\Theta,\Theta^*} = (\text{ɯ}(\Theta) - \text{ɯ}(\Theta^*))$;

$$\nabla_{\Theta,\Theta^*} = \begin{cases} < 0 & \textit{compression} \\ 0 & \textit{conservation} \\ > 0 & \textit{protraction} \end{cases}$$

However **Transformation 1** only caters for comparisons or analysis relative to size of the sequences being compared. And such that ∇_{Θ,Θ^*} shall work just fine where the purpose of the sequence analysis is just concerned with the relative sizes of the sequences. But, and as we established in the **Anagram Distance Theory**(ADT) paper[6], there are many important cases when it is necessary to compare two or more sequences relative to the relative ordering of members in them and not just by their member composition or frequencies. Thus the **Anagram**

[1] In the sense that, where a sequence already contains the essential symbols spanning some particular symbol set, and even though they might be scattered about or 'buried' around in the sequence surrounded by 'noise', the function would merely extract those essential symbols [only,] from that sequence in their natural order of first occurrence.

[2] Say, and as almost all transformatics might be in practice — readily translatable into a working or equivalent computer program based on or defined using the chaining of sequence transformers such as in a language as TEA[11].

[3] Refer to **Definition 3** in [7]

Distance Measure(ADM), and which, for any two sequences as presented above, the essential measure would be:

$$\tilde{A}(\Theta \to \Theta^*) = \frac{1}{\mathit{ɯ}(\Theta)} \times \sum_{\forall \theta_i \ \in \Theta}^{\mathit{ɯ}(\Theta)} |I(\theta_i, \Theta) - I(\theta_i, \Theta^*)| \tag{4}$$

Which, as we saw in [6], has the interesting interpretation as:

$$\tilde{A}(\Theta \to \Theta^*) = \begin{cases} 0, & \Theta = \Theta^*, \\ \leq 1, & \Theta \approx \Theta^*, \\ > 1, & \Theta \ll \Theta^*. \end{cases} \tag{5}$$

And thus can help us observe/compute, quantify and appreciate, the **relative disorder** resulting from one sequence being transformed into another (of potentially similar cardinality and member composition/distribution). A very important formalism in mathematical statistics dealing with measures of variability where traditional measures such as the range, mean deviation, variance and standard deviation statistics can't catch any actual differences between two or more sequences as we quantifiably/provably demonstrated in [3], but also, for simpler difference measures such as just counts/cardinality as we have seen in **Transformation 1**, and earlier on in [10].

2.3 Proposal 3: The Transformer Compression Ratio of any Sequence Transformation[3]

In **Transformation 1** we have seen that there might be cases in which a transformation results in a sequence that is observably larger or smaller than the source sequence; **protraction** or **compression** respectively. And as was laid down in [3] — refer to **Section 4.3** of [3] — we can measure the *overall* effect of a transform on the *entire* sequence merely by comparing the sequence cardinalities of the source and resultant sequence as such:

$$\xi(\Theta \to \Theta^*) = \frac{\mathit{ɯ}(\Theta^*)}{\mathit{ɯ}(\Theta)} \tag{6}$$

The measure, $\xi(\Theta \to \Theta^*)$ is called the **Transformer Compression Ratio** (TCR), and when compared against ∇_{Θ,Θ^*}, note that TCR is always positive, whereas the later, as we saw in **Transformation 1**, might be negative too. In fact, TCR spans $\mathbb{R}^+$ as was laid out in [3], and as we saw in that work, given $\mathit{ɯ}(\Theta) \geq 1$, can be interpreted as such:

$$\xi(\Theta \to \Theta^*) = \begin{cases} 1, & \text{the size of } \Theta \text{ was preserved}, \\ 0, & \Theta \text{ was reduced to nothing}, \\ > 1, & \Theta \text{ was extended/protracted}, \\ < 1, & \Theta \text{ was reduced/compressed}. \end{cases} \tag{7}$$

2.4 Proposal 4: The Piecemeal Compression Ratio of Any Sequence Transformation[3]

Further, with regards to how we might quantify the effects of some sequence transformation, note that, in cases such as:
Transformation 2. $\Theta : \mathbb{N} \times \psi_\beta \to \Omega : \mathbb{N} \times \psi_\alpha$;
$\psi_\beta \neq \psi_\alpha \quad \vee \quad \exists \omega \in \Omega : \omega \notin \Theta$

As explained in [3] — refer to **Section 4.3** of [3] — when the source and resultant sequence do not share any members in common, then we can only compare them using a measure such as **TCR**. However, in case they **at least** share 1 member in common, then, we can restrict our comparison to only their common aspects — i.e $\psi(\Theta) \cap \psi(\Theta^*)$, and thus, as laid out in **Definition 3** in [3], we can compute a special measure called the **Piecemeal Compression Ratio**(PCR) as such:

$$\Psi(\Theta \to \Theta^*) = \frac{1}{\mu(\psi_\Theta \cap \psi_{\Theta^*})} \times \sum_{\forall x \in (\psi_\Theta \cap \psi_{\Theta^*})} \frac{\mu(x \in \Theta^*) - \mu(x \in \Theta)}{\mu(x \in \Theta)} \tag{8}$$

And for the case of **Transformation 2**, can be re-written as:

$$\Psi(\Theta \to \Theta^*) = \frac{1}{\mu(\psi_\beta \cap \psi_\alpha)} \times \sum_{\forall x \in (\psi_\beta \cap \psi_\alpha)} \frac{\mu(x \in \Theta^*) - \mu(x \in \Theta)}{\mu(x \in \Theta)} \tag{9}$$

And as was laid out in that foundational paper on transformatics[3], we can interpret **PCR** values as such:

$$\Psi(\Theta, \Theta^*) = \begin{cases} undefined, & \psi_\Theta \cap \psi_{\Theta^*} = \emptyset \implies \text{no members in common,} \\ 0, & \mu(x \in \Theta) = \mu(x \in \Theta^*) \forall x \in \psi_\Theta \cap \psi_{\Theta^*} \implies \text{conservation,} \\ > 0, & \exists x \in \psi_\Theta \cap \psi_{\Theta^*} : \mu(x \in \Theta^*) > \mu(x \in \Theta) \implies \text{protraction,} \\ < 0, & \exists x \in \psi_\Theta \cap \psi_{\Theta^*} : \mu(x \in \Theta^*) < \mu(x \in \Theta) \implies \text{compression.} \end{cases} \tag{10}$$

2.5 Proposal 5: The Modal Sequence Statistic of Any Sequence[3]

For especially situations when we are not interested in comparing one sequence to another, but when we merely wish to better understand some particular sequence — like say, not just its size as could be told by computing the sequence cardinality, nor about its membership as we might tell using the sequence symbol set, there might be a case when we want to combine those two concepts into one; such as when we wish to **compare the relative frequency of members in a particular sequence**. Essentially, we might be faced with the problem:

Problem 1. *Given*
$\Theta^n : \mathbb{N} \times \psi_\beta : \theta_i, \theta_j \in \Theta^n : \theta_i \neq \theta_j \quad \wedge \quad i, j \in [1, n] : n \in \mathbb{N} : 1 \leq \mu(\theta_i \in \Theta^n) < n$,
is it true that $\mu(\theta_i \in \Theta^n) \geq \mu(\theta_j \in \Theta^n)$*?*

As laid out in **Section 4.1** of [3], we can approach a solution to **Problem 1** by leveraging the concept of the **modal sequence statistic** (MSS). Essentially, we would compute $\overset{>}{\Theta}^n$, which, by **Definition 1** in that foundation paper on transformatics[3], would give us a resultant sequence based on/derived from the source sequence Θ^n, such that the elements/members of the former exactly span the symbol set of the later/source sequence — i.e. $\psi(\Theta^n)$, and yet, there are not only no duplicates, but that the elements are sorted such that the member/symbol that occurs the most (with highest frequency) in Θ^n, occurs earliest in $\overset{>}{\Theta}^n$ than any other element, or at least (such as where any two elements have the same relative frequency in Θ^n), they occur in their natural order of first-occurrence.

Thus, if we compute a MSS for the sequence specified in **Problem 1** as such:

Transformation 3. $\Theta^n \xrightarrow{O_{mss}(\Theta^n)} \overset{>}{\Theta}^n$

Then, we can solve the problem thus:

Solution 1. $\text{ɯ}(\theta_i \in \Theta^n) \geq \text{ɯ}(\theta_j \in \Theta^n)? = \begin{cases} true & I(\theta_i, \overset{>}{\Theta}^n) \leq I(\theta_j, \overset{>}{\Theta}^n) \\ false & otherwise \end{cases}$

Not only shall we come to appreciate and love the MSS for being a statistical measure first introduced in a work on transformatics[3], but that, like the traditional statistical measure it is inspired by — the **mode** of any dataset/sequence — it is indispensable as a summary statistic, and yet, unlike and though as powerful as the mode, gives us a whole collection of summary statistics in one measure (as a **sequence or vector of modes**, and not merely as a single scalar value which is all that the traditional mode is).

Also, we have argued in several works exploring and applying transformatics — such as in [3], [6] and [10] — that the modal sequence statistic can be more useful to compute than many traditional summary statistics, but also that, it can help us arrive at other, more useful statistics (such as the next two sections are going to tackle), and for domains or problems such as in **statistical artificial intelligence**, machine learning and data mining to name but a few, is one of few core measures that can **greatly simplify decision making problems in an autonomous way**.

2.6 Proposal 6: The Characteristic of Any Sequence[10]

In **Proposal 5**, we have encountered the MSS, which is a neat summary statistic applicable to any sequence. However, and as you might notice when we are presented with a sequence such as:

$$\text{golden ratio} \implies \Theta_{gr} = \langle 1, 6, 1, 8, 0, 3, 3, 9, 8, 8, 7, 4, 9, 8, 9, 5 \rangle \tag{11}$$

Which is essentially the golden ratio[4] reduced to a sequence of just its first 16 significant digits, and which, if we must summarize it using say $\overset{>}{\Theta}_{gr}$ as such:

Transformation 4. $\Theta_{gr} \xrightarrow{O_{mss}(\cdot)} \overset{>}{\Theta}_{gr} = \langle 8, 9, 1, 3, 6, 0, 7, 4, 5 \rangle$

maps to a summary statistic, its modal sequence, of cardinality 9, and yet, if one wanted to solve the following problem:

[4] Basically, $\frac{(1+\sqrt{5})}{2}$

Problem 2. *Given the sequence* Θ_{gr} *in* ***Equation 11****, how can we exactly tell if indeed* $\mathit{ɯ}(3, \Theta_{gr}) > \mathit{ɯ}(6, \Theta_{gr})$*, or if* $\mathit{ɯ}(3, \Theta_{gr}) = \mathit{ɯ}(6, \Theta_{gr})$*?*

Note that though we might attempt to solve **Problem 2** naively by merely counting the terms in the sequence under question it is not the best way to arrive at a solution. Moreover, and as we have encountered in the previous proposal, and especially since we know that elements that tie on frequency shall occur adjacent to each other in the MSS of a sequence — ignoring their order of first-occurrence — and yet, without having a way to look at the frequencies of each element in the MSS, we might not correctly solve this problem *elegantly* and/or optimally (for very large sequences such as the kind we saw when treating of genetic code sequences[10] that might span millions of terms).

And so, as laid out in a treatise on how transformatics might be applied in the sequence-heavy science of genetics[10], in particular, **Section 6.4** of [10], instead of just the MSS, we can instead compute the related **sequence characteristic** (SC) statistic, and which, as per **Definition 8** in that work, consists of both the same exact elements as in the MSS, but with each distinct symbol followed by/annotated by its relative frequency in the original/source sequence. Essentially, given some sequence Θ, and given we know its modal sequence statistic is $\overset{>}{\Theta}$, the SC is as:

$$\hbar(\overset{>}{\Theta}) = \prod_{\omega_i \in \psi(\Theta)} \omega_i \cdot f_i \tag{12}$$

Which, if we apply **Equation 12** to our example sequence Θ_{gr}, produces the corresponding sequence characteristic as such:

Transformation 5. $\Theta_{gr} \xrightarrow{O_{mss}(\cdot)} \overset{>}{\Theta}_{gr} \xrightarrow{O_{sc}(\cdot)} \hbar(\overset{>}{\Theta}_{gr}) \rightarrow 8_4.9_3.1_2.3_2.6_1.0_1.7_1.4_1.5_1$
$\rightarrow$ ***8***4.***9***3.***1***2.***3***2.***6***1.***0***1.***7***1.***4***1.***5***1 $\rightarrow$ ***8******4******9******3******1******2******3******2******6******1******0******1******7******1******4******1******5***1

Note that, in **Transformation 5**, we taken the liberty to express the sequence statistic in 3 different ways other than what was originally demonstrated in [10]. In particular, the alternative expressions for the SC thus shown, shall, such as in this case when we are dealing with a sequence made of entirely numbers/digits, help to make the SC more readable/understandable; instead of merely concatenating the computed frequency of each symbol to to the symbol such as the basic definition for the SC calls for (see **Definition 8** in [10]), and yet, without loss of generality, we can redefine the SC as such:

$$\hbar(\overset{>}{\Theta}) = \prod_{\omega_i \in \psi(\Theta)} \omega_{i f_i} \tag{13}$$

So that, as we have seen in the first SC expression above, we can merely write the SC for Θ_{gr} as such:

$$\hbar(\overset{>}{\Theta}_{gr}) = 8_4.9_3.1_2.3_2.6_1.0_1.7_1.4_1.5_1 \rightarrow 8_4 9_3 1_2 3_2 6_1 0_1 7_1 4_1 5_1 \tag{14}$$

The final form in **Equation 14** also kicking out the "." symbols meant to show that we are merely concatenating the terms to each other. In fact, the SC could be also expressed as a sequence of **2-grams**[5]:

$$\hbar(\overset{>}{\Theta}) = \prod_{\omega_i \in \psi(\Theta)} \langle \omega_i, f_i \rangle \tag{15}$$

So that, we can again further express the sequence characteristic for the golden ratio as such:

$$\hbar(\overset{>}{\Theta}_{gr}) = \langle\langle 8,4\rangle, \langle 9,3\rangle, \langle 1,2\rangle, \langle 3,2\rangle, \langle 6,1\rangle, \langle 0,1\rangle, \langle 7,1\rangle, \langle 4,1\rangle, \langle 5,1\rangle\rangle \tag{16}$$

At this juncture, and returning to **Problem 2**:

Solution 2. *Given Θ_{gr}, we can merely compute its sequence characteristic as shown in **Equation 16**, and by inspecting the entries in that resulting SC for the two symbols under question — '3' and '6', which are known to map to the entries: $\langle ...\langle 3,2\rangle, \langle 6,1\rangle ...\rangle$ in the SC, confirm that actually $\mu(3,\Theta_{gr}) > \mu(6,\Theta_{gr})$* □

.

Before we move on, and as we shall come to appreciate when we come to **Proposal 9**, we can solve many problems not only by leveraging the theory and mathematics of transformatics, but also by leveraging its philosophy, which, as shall establish, underlies how the **TEA** computer programming language[11] works. Essentially, and with reference to the present proposal, note that we can readily and elegantly compute the sequence characteristic of any sequence[6] by leveraging **sequence-transformers** and **sequence-processors** as shown in the following TEA program that can compute the SC of any input sequence, and produce an SC expression similar to what we saw in **Transformation 5**:

TEA Program: SC-Transformer

```
#SC-Computer
#i!:{1618033988749895}
v:vIN
u!:|v:vMSS| v:vSC:{}
l:lPROC
y:vMSS|d!:^.|v:vW
y:vMSS|d:^.|v:vMSS|y:vW
f:^$:lEND
y:vIN|d*!:vW|v:|v!:|v:vF|v:vD:_|g*:_:vW:vF:vD|v:vSC1 |
```

[5] In **Chapter 3** of [10] we already introduced **N-grams** or rather higher-order sequences, but shall revisit them again in **Proposal 8**.

[6] Sometimes though, after slight modifications or clean-up of the sequence expression such as we saw in **Equation 11**.

```
g*:{}:vSC:vSC1| v:vSC| j:lPROC
l:lEND
y:vSC | r!:__:.|d:.$
#(=8_4.9_3.1_2.3_2.6_1.0_1.7_1.4_1.5_1)
```

Figure 1: TEA EXAMPLE: **SC-Computer**: transforms any input sequence into its sequence characteristic!

In case you have a particular sequence of interest that you wish to analyze or for which you wish to compute the SC, then, based on the above theory and the sample TEA program we have just looked at, you can go to the **WEB TEA IDE** — `https://tea.nuchwezi.com`, and paste in your sequence into the 'TEA INPUT' section (for example paste in your long genetic code sequence, a text file, votes/elections tally data well-encoded etc.), and by running it against the SC-program — which you might load as:

`https://tea.nuchwezi.com/?fc=https://gist.githubusercontent.com/mcnemesis/b5b9957179a4a1d43b965959f9bb29fb/raw/sc_generator.tea`

You shall then be able to replicate the results and analyses we have conducted in this proposal.

2.7 Proposal 7: The Population Characteristic of Any Collection of Sequences[10]

In **Proposal 6**, we have dealt with the matter of applying transformatics extending as well as simplifying an important aspect of statistics — statistical analysis in particular, and which might also underlie both descriptive (generation of graphs and visualizations summarizing some dataset or sequence) and computational statistics (such as in modeling or implementing statistical artificial intelligence programs that reason about the statistical patterns in a dataset or sequence).
In this section though, and building upon what we have already encountered, we are to re-iterate the concept of the **Population Characteristic** (PC), that was first presented in the earlier work about transformatics in GENETICS[10]. That book, in **Section 6.6**, presented a problem for which this proposed concept is the solution. More specifically, and without repeating what was already well discoursed then, we note that the PC has the following useful properties:

1. Given any **flat-structure**/first-order sequence, Θ, the PC is essentially the same as the SC — in terms of what its computation produces.
2. In case two or more sequences are to be reduced to a single characteristic — in which case it becomes the **population characteristic** for the combined set, the PC thus computed shall be equivalent to the SC of a sequence generated from flattening and combining all the involved sequences into a single sequence.
3. Where two or more sequences are involved, and where the SC for each sequence is already known or computed, the PC for the entire collection can readily be derived from the SCs thus already computed.

In summary though, and given a collection of n sequences[7] — basically a sequence of n-subsequences — denoted Θ^n, we then can compute the PC as such:

$$\hbar(\overset{>}{\Theta^n}) = \prod_{\forall \omega_i \in \psi(\Theta^n)} \omega_i \cdot \text{ɯ}(\omega_i \in \Theta^n) \tag{17}$$

[7]Potentially of arbitrary length and composition.

Where $\forall \Theta_j \in \Theta^n$

$$\mathscr{u}(\omega_i \in \Theta^n) = \sum_{\forall \hbar(\overset{>}{\Theta}_j)} f_{\omega_i} \tag{18}$$

And $\forall i, j \in \mathbb{N} : i < j \implies f_i \geq f_j$ for the frequency terms in $\hbar(\overset{>}{\Theta^n})$.

Note that, as laid out in [10], the PC can be very indispensable in problems **where we need to analyze multiple sequences in-groups or clusters** — such as in classification problems, clustering as part of artificial intelligence systems performing tasks such as **self-modifying maps**, unsupervised learning, etc. All of which are critical matters in statistical artificial intelligence, but also, which could be useful in many other areas of applied mathematics and computation.

Before we conclude this section, note that, just like we did with and saw with the SC in **Proposal 6**, we can use the TEA language to compute a population statistic for arbitrary sequences (after careful and proper pre-processing perhaps), but also, and concerning how we might express the final output/PC, we can chose from several meaningful expression forms as proposed for the SC in **Proposal 6** and depending on where or how the PC expression or statistic is to be utilized.

2.8 Proposal 8: A Super Sequence from One or More Sequences[10]

The idea of treating of sequences in other than their flat-structure form was first presented as part of the discussion in how to apply transformatics in GENETICS[10]; refer to **Chapter 3** of [10]. In that work, we demonstrated how, for cases such as in dealing with problems where a sequence might need to be processed in "chunks" of a fixed size — also known as **n-grams**, we might want to introduce abstractions that can allow us to form smaller-sequences/sub-sequences within a larger/the original flat-structure sequence.
For example, we saw that, in case we have some sequence of symbols Θ^n defined as such:

$$\Theta^n = \langle a_1, a_2, ..., a_n \rangle \tag{19}$$

And that if we wish to re-write the same sequence such that every k terms are grouped in the same sub-sequence, we then can re-write Θ^n differently as such:

$$\Theta^n \to \Theta^* = \Theta_2(n,k) = \langle a_{11}, a_{12}, a_{13}, a_{1k}, a_{21}, a_{22}, ..., a_{(i)(k)}, ..., a_{(\frac{n}{k})(k-1)}, a_{(\frac{n}{k})(k)} \rangle \tag{20}$$

Further, and for any arbitrary sequences, we saw that we could leverage a special kind of transformer that can take a flat-structure or lower-order sequence and from it generate a similar, but higher-order sequence (also understood to mean, a sequence of some n-grams/k-grams such that $k > 1$), as such:

Transformer 2 (The **k-GRAM Generator**). $\Theta^n = \langle a_1, a_2, ..., a_n \rangle \quad \xrightarrow{O_{bundle-k}(\cdot)} \quad \Theta^* \quad ;$
$\Theta^* = \langle \langle a_1, a_2, ..., a_k \rangle_1, \langle a_{k+1}, ..., a_{2k} \rangle_2, ..., \langle a_{(n-k+1)}, ..., a_{(n-k+k-1)}, a_{(n-k+k)} \rangle_{\frac{n}{k}} \rangle$
$\forall a_i \in \Theta \quad \exists a_{ij} \in \Theta^* : a_i = a_{ij} \quad \wedge \quad j \in [1, k]$
For some $n, k \in \mathbb{N}$, and that $\mathscr{u}(\Theta^*) = \frac{n}{k}$ = number of k-gram subsequences generated from Θ. $\square$

For future purposes, and without deviating from the theory already presented in earlier works, we might also refer to such produced sequences as Θ^* in **Transformer 2** as a **Super Sequence**.
We shall note that if Θ^* is a super-sequence of/relative-to another sequence, such as the sub-sequence $\langle a_1, a_2, ..., a_k \rangle_1$ in **Transformer 2**, and which we might denote as Θ_1, then, we can correctly say that:

$$\Theta^* \supset \Theta_1 \quad \wedge \quad \psi(\Theta^*) \supset \psi(\Theta_1) \tag{21}$$

And that generally,

$$\forall \Theta_i \in \Theta^* \quad \Longrightarrow \quad \Theta^* \supset \Theta_i \quad \wedge \quad \psi(\Theta^*) \supset \psi(\Theta_i) \tag{22}$$

Also, and as shown in **Section 6.6.1** of [10], we might also treat of super-sequences as matrices (in the sense of **linear algebra**). In particular, note that if we have a set of n different sequences of arbitrary composition and lengths:

$$\Omega^n = \langle \Omega 1, \Omega 2, \Omega 3, ..., \Omega n \rangle \tag{23}$$

That we might also express them more elaborately via matrix form as:

$$\Omega^n \quad = \quad \begin{bmatrix} \Omega 1\langle \prod_{i=1}^{\varkappa(\Omega 1)=\alpha} a_{1i} \rangle, \\ \Omega 2\langle \prod_{i=1}^{\beta} a_{2i} \rangle, \\ \Omega 3\langle \prod_{i=1}^{\chi} a_{3i} \rangle, \\ \vdots \\ \Omega n\langle \prod_{i=1}^{\zeta} a_{ni} \rangle, \end{bmatrix} \quad = \quad \begin{bmatrix} \Omega 1\langle a_{11}, a_{12}, ..., a_{1\alpha} \rangle, \\ \Omega 2\langle a_{21}, a_{22}, ...a_{2(\beta-1)}, a_{2\beta} \rangle, \\ \vdots \\ \Omega n\langle a_{n1}, a_{n2}, ..., a_{n\zeta} \rangle \end{bmatrix} \tag{24}$$

So that, if we have the special case as what **Transformer 2** would produce if given the right kind of input sequence (particularly, that if Θ^n is to be split into k-grams, then that it has the property that $\frac{n}{k} = m$ is a pure number, and so that, we can then map the flat-structure sequence Θ^n to a higher-order sequence, or super-sequence of **dimensions** $k \times m$, so that the resulting mathematical structure can be neatly expressed as a $k \times m$ or $m \times k$ matrix as shown:

Transformation 6. $\Theta^n = \langle \theta_1, \theta_2, ..., \theta_n \rangle \xrightarrow{O_{bundle-k}(\Theta^n)} \begin{bmatrix} \theta_{11}, \theta_{12}, ..., \theta_{1k}, \\ \theta_{21}, \theta_{22}, ..., \theta_{2k}, \\ \vdots \\ \theta_{m1}, \theta_{m2}, ..., \theta_{mk} \end{bmatrix}$

So that, if $m = k$, then we essentially have a super-sequence that can be expressed as a square-matrix of dimensions $m \times m$. Talking of which, once we can or do express a sequence as a matrix[8], almost all of the mathematical theory of linear-algebra does readily apply on such sequences from transformatics! And thus we sum up our proposal on transformatics and super-sequences.

2.9 Proposal 9: Programs as Chains of Sequence Transformers (Transformer-Chains)[10][11]

First, note that, the useful program we encountered at the end of **Proposal 6** — a program that basically computes the sequence characteristic of any input sequence, and which is basically a *higher-*

[8]Even as a *sparse matrix* such as in cases where a low-order sequence Θ^n needs be transformed into a super-sequence, but when either $\frac{n}{k}$ is not a pure number, or that there are some sub-sequences that might have empty positions.

order transformer that is composed as a chain of smaller sequence transformers — which is what all or most of **TEA primitive instructions**[12] are), and which transformer then (as the "SC-Generator") essentially reduces any input sequence to its summary statistic expression, particularly the sequence characteristic.
Otherwise, note that looking at a TEA program as depicted in **Figure 1** is helpful, since it reflects how the overall [higher-order] transformer program might be composed out of logically-meaningful sections[9] as a software engineer or computer programmer already familiar with the TEA language might approach the SC problem. However, note that, that same exact program is also equivalent to the following two versions — both of which kick-out comments, and only **focus on the individual TEA instructions necessary to make the program work as desired, as well as how they ought be ordered within the program**, but where, the first (**Listing 1**) simplifies the code to just an ordered sequence of TEA instructions one per line, while the second (**Listing 2**) shows the same exact program, in its **minified** form — where, each instruction is delimited from the next by the standard TEA instruction delimiter '|', and nothing else.

TEA Program: SC-Generator

Listing 1: SC-Generator

```
v:vIN
u!:
v:vMSS
v:vSC:{}
l:lPROC
y:vMSS
d!:^.
v:vW
y:vMSS
d:^.
v:vMSS
y:vW
f:^$:lEND
y:vIN
d*!:vW
v:
v!:
v:vF
v:vD:_
g*:_:vW:vF:vD
v:vSC1
g*:{}:vSC:vSC1
v:vSC
j:lPROC
l:lEND
y:vSC
r!:__:.
d:.$
```

And the same program in its minified form is as:

[9] In TEA parlance and in relation to how most computer scientists might think of matter as *scoping*, *code-blocks* or *encapsulation*, for TEA, it is mostly all about **l-blocks** — also known as "labeled blocks"[12].

TEA Program: SC-Generator Minified

Listing 2: SC-Generator Minified

```
v:vIN|u!:|v:vMSS|v:vSC:{}|l:lPROC|y:vMSS|d!:^.|v:vW|y:vMSS|
d:^.|v:vMSS|y:vW|f:^$:lEND|y:vIN|d*!:vW|v:|v!:|v:vF|v:vD:_|
g*:_:vW:vF:vD|v:vSC1|g*:{}:vSC:vSC1|v:vSC|j:lPROC|l:lEND|
y:vSC|r!:__:.|d:.$
```

As one shall establish while studying **the 26 TEA primitives A: to Z:**[12], or if they dive deeper into the potential **182 distinct TEA commands**[12], apart from branching commands such as **F:** (the "Fork") and **Q:** (the "Quit") and commands such as **L:** (the "Label") that don't modify or alter the **Active Input**(AI)[12] — meaning, when they are encountered in a TEA program, they merely pass on the current input to the next instruction unmodified, most, or all of the bulk of TEA programming consists of some sequence processing instruction that accepts some input, processes it or transforms it, and then returns the resultant [sequence] as the **Instruction Output**(IO).
Thus, and without loss of generality, we can model or think of any program expressible in the TEA language as a computer program, abstract machine or automaton, expressible as or reducible to just an ordered chain of sequence transformers. A good demonstration of this is the following basic program, which, when presented with any input text (a sequence of characters), reduces or transforms it into a resultant sequence preserving their original order, but with only UPPER-CASE CONSONANTS left.

TEA Program: CONSONANT FILTER

```
i:{This is a TEST 123} | z!: | d:[AEIOU] | d!:[A-Z]
```

Which program, when run as is, shall use the explicit hard-coded input it opens with — "This is a TEST 123", and shall return "THSSTST". While, if presented with some other input such as "hello world 123?", shall return "HLLWRLD".
Though the **TEA TAZ**[12] is still being worked-on as of this writing, much of what is required to understand or write TEA programs is already covered well in that living document, and so, it must be consulted and utilized by those who wish to understand better, but also apply readily, many ideas encountered while working with or developing ideas using the mathematics of transformatics.

2.10 Proposal 10: Entropy of a Sequence[4]

Information expressions (IE)[4] are a mathematical structure, $\Omega^{1,\infty} : \mathbb{N} \times \Psi^{\infty}$, to which **Foundation Idea 1.3** applies. The sequence $\Theta_{hw} = \langle \texttt{hello world} \rangle$ is one such example, with the properties:

$$\Theta_{hw} \rightarrow \begin{cases} \mu(\Theta_{hw}) \rightarrow \mathbb{N} & \text{bits in information} \\ \mu(\psi(\Theta_{hw})) \rightarrow \mathbb{N} & \text{cardinality of alphabet} \\ \psi(\Theta_{hw}) \times \psi(\Theta_{hw})^{[1,\mu(\Theta_{hw})]} \quad \vee \quad \psi(\Theta_{hw}) \times \psi(\Theta_{hw})^{[1,\infty]} \rightarrow \mathbb{N} \times \mathbb{R} & \text{derivative languages} \end{cases} \tag{25}$$

Given that, in applying the mathematics of transformatics as laid out in **Foundation Idea 1.3**, we can leverage a mathematical and philosophical abstraction such as the **sequence** and **sequence-transformer** (as especially a transformatic formalism), so that then, we can readily build pragmatic/empirical solutions to any problems for which a solution can be conceptualized, manifested and

proven, based on some model of the problem that allows the problem space to be fully specifiable via the use of one or more sequences — **information expressions**[4] — essentially, *arguments* or parameters — that for example **encode** the character, name, dimensions etc. of the system in which the problem might be rendered tractable/decidable/computable/... so that, then, by applying some particular sequence transformer [program[10]] — itself, an information expression, though, of **a different kind** than the data upon which it might be applied[11] — we can effect sequences that can modify or predictably operate on other sequences — a mechanism and capability which, when well orchestrated and manifested as is the case of the **TEA computer programming language**[12], brings to life a clear arcanum practicum known as a living, reusable, [technology] platform (both in theory/formalism) and experience(executable mathematics) that, at foundation, is beautifully based on and comprehensible via the theory of transformatics as laid out in the **Foundations**.

References

[1] Joseph Willrich Lutalo. **Transformatics 101**. *Thesis*, 10 2025. *A first distillation of the essential foundations of Transformatics as its own field of mathematics. Accessible via:* `https://doi.org/10.6084/m9.figshare.30285106.v1`".

[2] Joseph Willrich Lutalo. **Transformatics 101 (1-Page Executive Summary)**, Oct 2025. *The 1-pager PDF is available at:* `https://doi.org/10.6084/m9.figshare.30284035`.

[3] Joseph Willrich Lutalo. **The Theory of Sequence Transformers & their Statistics**: The 3 information sequence transformer families (anagrammatizers, protractors, compressors) and 4 new and relevant statistical measures applicable to them: Anagram distance, modal sequence statistic, transformation compression ratio and piecemeal compression ratio. *Academia*, 2025. `https://doi.org/10.6084/m9.figshare.29505824.v3`.

[4] Joseph Willrich Lutalo. **Philosophical and Mathematical Foundations of A Number Generating System: The Lu-Number System**. *Academia.edu*, 2025. *Accessible via* `https://doi.org/10.6084/m9.figshare.29262749`.

[5] Joseph Willrich Lutalo. **A general theory of number cardinality**. *Academia.edu*, Jan 2024. *Accessible via* `https://www.academia.edu/43197243/A_General_Theory_of_Number_Cardinality`.

[6] Joseph Willrich Lutalo. **Introducing The Anagram Distance Statistic, $\tilde{\text{A}}$, A Quantifier of Lexical Proximity For Base-10 o-SSI And Any Arbitrary Length Ordered Sequences, Its Relevance And Proposed Applications in Computer Science, Engineering And Mathematical Statistics**. *Academia.edu*, 2025. *Accessible via* `https://doi.org/10.6084/m9.figshare.29402363`.

[7] Joseph Willrich Lutalo. **Concerning A Special Summation That Preserves The Base-10 Orthogonal Symbol Set Identity In Both Addends And The Sum**. *Academia*, 2025. *Accessible via* `https://www.academia.edu/download/122499576/The_Symbol_Set_Identity_paper_Joseph_Willrich_Lutalo_25APR2025.pdf`.

[8] Joseph Willrich Lutalo. **Transformatics for PSYCHOLOGY**. *FigShare*, 9 2025. *Presented at the 13th International East African Psychology Conference, a copy is accessible via:* `https://doi.org/10.6084/m9.figshare.30231748`.

[10] For this case, also equivalently a **definition**, specification, symbol sequence, etc. applicable to **transformers**

[11] Or, as is **the case** in **TEA**; where, as with the programs of a *Von Neumann Architecture* machine, allows the language to allow for a systematic means in which a program can be expressed with explicit data it might process as a default, but also with the possibility of the program operating on information *read* from the environment (when it is available/presented).

[9] Joseph Willrich Lutalo. **Applying TRANSFORMATICS: Sequence Generators**. https://doi.org/10.6084/M9.FIGSHARE.29654645, 2025. FigShare.

[10] Joseph Willrich Lutalo. **Applying TRANSFORMATICS in GENETICS**. *I*POW*, pages 1–206, Aug 2025. A mini-treatise on the mathematics, philosophy and practice of Transformatics for Geneticists. https://doi.org/10.6084/m9.figshare.29941127.

[11] Joseph Willrich Lutalo. **TEA GitHub Project — The Reference Implementation of TEA (Transforming Executable Alphabet) computer programming language**. https://github.com/mcnemesis/cli_tttt/, 2024. *Single Source of Truth concerning the reference standards of TEA on the web and native operating systems.*

[12] Joseph Willrich Lutalo. **TEA TAZ - Transforming Executable Alphabet A: to Z: COMMAND SPACE SPECIFICATION**. *Academia.edu*, 2024. Accessed on 14 May, 2025, via https://www.academia.edu/122871672/TEA_TAZ_Transforming_Executable_Alphabet_A_to_Z_COMMAND_SPACE_SPECIFICATION.