More ties than we thought

Dan Hirsch

Upstanding Hackers Inc.

`thequux@upstandinghackers.com`

Ingemar Markström

KTH Royal Institute of Technology, Stockholm

`ingemarm@kth.se`

Meredith L. Patterson

Upstanding Hackers Inc.

`mlp@upstandinghackers.com`

Anders Sandberg

Oxford University

`anders.sandberg@philosophy.ox.ac.uk`

Mikael Vejdemo-Johansson

KTH Royal Institute of Technology, Stockholm

Jožef Štefan Institute, Ljubljana

Institute for Mathematics and its Applications, Minneapolis

`mvj@kth.se`

May 5, 2015

Abstract

We extend the existing enumeration of neck tie-knots to include tie-knots with a textured front, tied with the narrow end of a tie. These tie-knots have gained popularity in recent years, based on reconstructions of a costume detail from The Matrix Reloaded, and are explicitly ruled out in the enumeration by Fink and Mao (2000).

We show that the relaxed tie-knot description language that comprehensively describes these extended tie-knot classes is context free. It has a regular sub-language that covers all the knots that originally inspired the work.

From the full language, we enumerate 266 682 distinct tie-knots that seem tie-able with a normal neck-tie. Out of these 266 682, we also enumerate 24 882 tie-knots that belong to the regular sub-language.

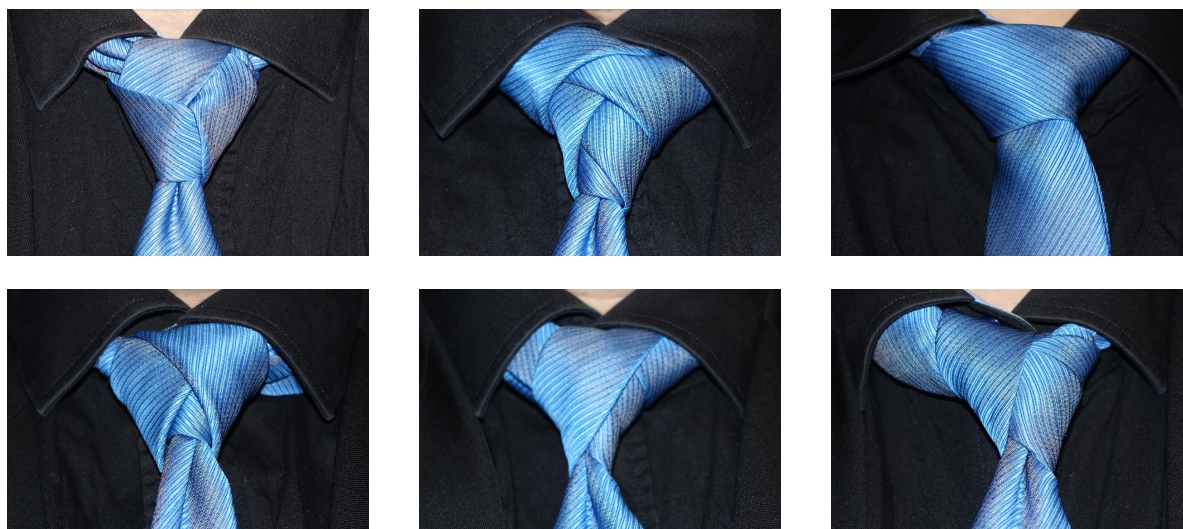



Figure 1: Some specific tie-knot examples. Top row from left: the Trinity (L-110.4), the Eldredge (L-373.2) and the Balthus (C-63.0, the largest knot listed by Fink and Mao). Bottom row randomly drawn knots. From left: L-81.0, L-625.0, R-353.0

1   Introduction

There are several different ways to tie a necktie. Classically, knots such as the four-in-hand, the half windsor and the full windsor have commonly been taught to new tie-wearers. In a sequence of papers and a book, Fink and Mao [2–4] defined a formal language for describing tie-knots, encoding the topology and geometry of the knot tying process into the formal language, and then used this language to enumerate all tie-knots that could reasonably be tied with a normal-sized necktie.

The enumeration of Fink and Mao crucially depends on dictating a particular finishing sequence for tie-knots: a finishing sequence that forces the front of the knot – the façade – to be a flat stretch of fabric. With this assumption in place, Fink and Mao produce a list of 85 distinct tie-knots, and determine several novel knots that extend the previously commonly known list of tie-knots.

In recent years, however, interest has been growing for a new approach to tie-knots. In *The Matrix Reloaded* [13], the character of "The Merovingian" has a sequence of particularly fancy tie-knots. Attempts by fans of the movie to recreate the tie-knots from the Merovingian have led to a collection of new tie-knot inventions, all of which rely on tying the tie with the thin end of the tie – the thin blade. Doing this allows for a knot with textures or stylings of the front of the knot, producing symmetric and pleasing patterns.

Knorr [6] gives the history of the development of novel tie-knots. It starts out in 2003 when the *edeity knot* is published as a PDF tutorial. Over the subsequent 7 years, more and more enthusiasts involve themselves, publish new approximations of the Merovingian tie-knot as PDF files or YouTube videos. By 2009, the new tie-knots are featured on the website Lifehacker and go viral.

In this paper, we present a radical simplification of the formal language proposed by Fink and Mao, together with an analysis of the asymptotic complexity class of the tie-knots language. We produce a novel enumeration of necktie-knots tied with the thin blade, and compare it to the results of Fink and Mao.

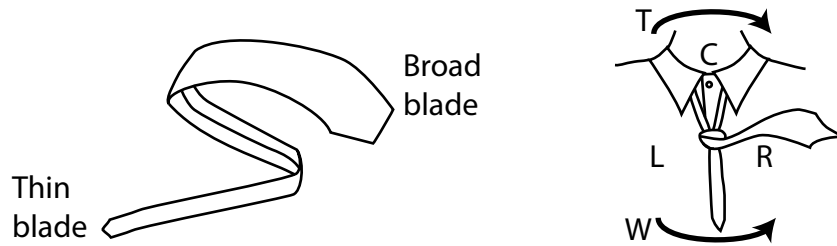



Figure 2: The parts of a necktie, and the division of the wearer's torso with the regions (Left, Center Right) and the winding directions (Turnwise, Widdershins) marked out for reference.

1.1 Formal languages

The work in this paper relies heavily on the language of formal languages, as used in theoretical computer science and in mathematical linguistics. For a comprehensive reference, we recommend the textbook by Sipser [9].

Recall that given a finite set $\mathcal{L}$ called an *alphabet*, the set of all sequences of any length of items drawn (with replacement) from $\mathcal{L}$ is denoted by $\mathcal{L}^*$. A *formal language* on the alphabet $\mathcal{L}$ is some subset $\mathcal{A}$ of $\mathcal{L}^*$. The complexity of the automaton required to determine whether a sequence is an element of $\mathcal{A}$ places $\mathcal{A}$ in one of several complexity classes. Languages that are described by finite state automata are *regular*; languages that require a pushdown automaton are *context free*; languages that require a linear bounded automaton are *context sensitive* and languages that require a full Turing machine to determine are called *recursively enumerable*. This sequence builds an increasing hierarchy of expressibility and computational complexity for syntactic rules for strings of some arbitrary sort of tokens.

One way to describe a language is to give a *grammar* – a set of production rules that decompose some form of abstract tokens into sequences of abstract or concrete tokens, ending with a sequence of elements in some alphabet. The standard notation for such grammars is the Backus-Naur form, which uses ::= to denote the production rules and ⟨*some name*⟩ to denote the abstract tokens. Further common symbols are ∗, the Kleene star, that denotes an arbitrary number of repetitions of the previous token (or group in brackets), and |, denoting a choice of one of the adjoining options.

2 The anatomy of a necktie

In the following, we will often refer to various parts and constructions with a necktie. We call the ends of a necktie *blades*, and distinguish between the *broad blade* and the *thin blade*[1] – see Figure 2 for these names. The tie-knot can be divided up into a *body*, consisting of all the twists and turns that are not directly visible in the final knot, and a *façade*, consisting of the parts of the tie actually visible in the end. In Figure 3 we demonstrate this distinction. The body builds up the overall shape of the tie-knot, while the façade gives texture to the front of the knot. The enumeration of Fink and Mao only considers knots with trivial façades, while these later inventions all consider more interesting façades. As a knot is in place around a wearer, the Y-shape of the tie divides the torso into 3 regions: Left, Center and Right – as shown to the right in Figure 2.

A tie-knot has to be tied by winding and tucking one of the two blades around the other: if both blades are active, then the tie can no longer be adjusted in place for a comfortable fit. We shall refer to the blade used in tying the knot as the *leading blade* or the *active blade*. Each time the active blade is moved across the tie-knot – in front or in back – we call the part of the tie laid on top of the knot a *bow*.

---

[1]There are neckties without a width difference between the ends. We ignore this distinction for this paper.

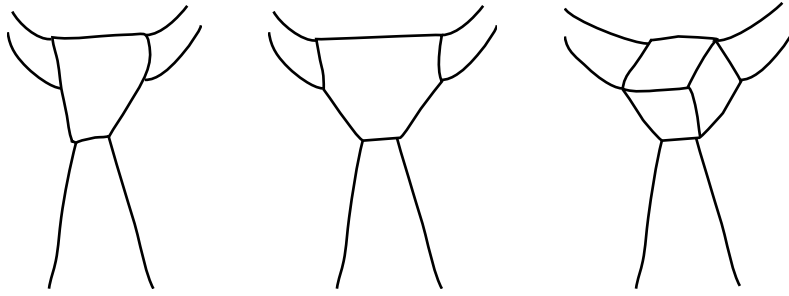



Figure 3: Different examples of tie-knots. Left, a 4-in-hand; middle, a double windsor; right a trinity. The 4-in-hand and double windsor share the flat façade but have different bodies producing different shapes. The trinity has a completely different façade, produced by a different wind and tuck pattern.

3   A language for tie-knots

Fink and Mao [3] observe that once the first crossing has been made, the wrapping sequence of a classical tie-knot is completely decided by the sequence of regions into which the broad blade is moved. Adorning the region specifications with a direction – is the tie moving away from the wearer or towards the wearer – they establish a formal alphabet for describing tie-knots with 7 symbols. We reproduce their construction here, using U for the move to tuck the blade *U*nder the tie itself[2]. The notation proposed by Fink and Mao [3] interprets repetitions U^k of U as tucking the blade k bows under the top. It turns out that the complexity analysis is far simpler if we instead write U^k for tucking the blade under the bow that was produced $2k$ windings ago. This produces a language on the alphabet:

$$\{L_\otimes, L_\odot, C_\otimes, C_\odot, R_\otimes, R_\odot, U\}$$

They then introduce relations and restrictions on these symbols:

$\mathcal{T}ie_1$ No region (L, C, R) shall repeat: after an L only C or R are valid next regions. U moves do not influence this.

$\mathcal{T}ie_2$ No direction ($\odot$ – out of the paper, $\otimes$ – in towards the paper) shall repeat: after an outwards move, the next one must go inwards. U moves do not influence this.

$\mathcal{T}ie_3$ Tucks (U) are valid after an outward move.

$\mathcal{T}ie_4$ A tie-knot can end only on one of $C_\otimes, C_\odot$ or U. In fact, almost all classical knots end on U.[3]

$\mathcal{T}ie_5$ A k-fold tuck U^k is only valid after at least $2k$ preceding moves. Fink and Mao [3] do not pay much attention to the conditions on k-fold tucks, since these show up in their enumeration as stylistic variations, exclusively at the end of a knot.

This collection of rules allow us to drastically shrink the tie language, both in alphabet and axioms. Fink and Mao are careful to annotate whether tie-knot moves go outwards or inwards at any given point. We note that the inwards/outwards distinction follows as a direct consequence of axioms $\mathcal{T}ie_2$, $\mathcal{T}ie_3$ and $\mathcal{T}ie_4$. Since non-tuck moves must alternate between inwards and

---

[2]Fink and Mao used T for *T*uck

[3]The exemption here being the Onassis style knot, favored by the eponymous shipping magnate, where after a classical knot the broad blade is brought up with a $C_\odot$ move to fall in front of the knot, hiding the knot completely.

89 outwards, and the last non-tuck move must be outwards, the orientation of any sequence of
90 moves follows by backtracking from the end of the string.

91   Hence, when faced with a non-annotated string like

$$\texttt{RCLCRCLCRCLRURCLU}$$

92 we can immediately trace from the tail of the knot string: the last move before the final tuck
93 must be outwards, so that L must be a $L_\odot$. So it must be preceded by $R_\odot C_\otimes$. Tracing backwards,
94 we can specify the entire string above to

$$R_\otimes C_\odot L_\otimes C_\odot R_\otimes C_\odot L_\otimes C_\odot R_\otimes C_\odot L_\otimes R_\odot U C_\otimes R_\odot C_\otimes L_\odot U$$

95   Next, the axiom $\mathcal{T}ie_1$ means that a sequence will not contain either of $\texttt{LU}^*\texttt{L}, \texttt{CU}^*\texttt{C}, \texttt{RU}^*\texttt{R}$ as
96 subsequences[4]. Hence, the listing of regions is less important than the direction of transition:
97 any valid transition is going to go either clockwise or counterclockwise[5]. Writing $\texttt{T}$ for clockwise[6]
98 and $\texttt{W}$ for counterclockwise[7], we can give a strongly reduced tie language on the alphabet $\texttt{T, W,}$
99 $\texttt{U}$. To completely determine a tie-knot, the sequence needs a starting state: an annotation on
100 whether the first crossing of a tie-knot goes across to the right or to the left. In such a sequence,
101 a $\texttt{U}$ instruction must be followed by either $\texttt{T}$ or $\texttt{W}$ dictating which direction the winding continues
102 after the tuck, unless it is the last move of the tie: in this case, the blade is assumed to continue
103 straight ahead – down in front for most broad-blade tie-knots, tucked in under the collar for
104 most thin-blade knots.

105   The position of the leading blade after a sequence of $\texttt{W}/\texttt{T}$ windings is a direct result of $\#W -$
106 $\#T$ (mod 3). This observation allows us to gain control over several conditions determining
107 whether a distribution of $\texttt{U}$ symbols over a sequence of $\texttt{W}/\texttt{T}$ produces a physically viable tie-knot.

108 **Theorem 1.** *A position in a winding sequence is valid for a k-fold tuck if the sub-sequence of*
109 *the last $2k$ $\texttt{W}$ or $\texttt{T}$ symbols is such that either*

110   *1. starts with $\texttt{W}$ and satisfies $\#W - \#T = 2$ (mod 3)*

111   *2. starts with $\texttt{T}$ and satisfies $\#T - \#W = 2$ (mod 3)*

112 *Proof.* The initial symbol produces the bow under which the tuck will go. If the initial symbol
113 goes, say, from $\texttt{R}$ to $\texttt{L}$, then the tuck move needs to come from $\texttt{C}$ in order to go under the bow.
114 In general, a tuck needs to come from the one region not involved in the covering bow. Every
115 other bow goes in front of the knot, and the others go behind the knot. Hence, there are $2k - 1$
116 additional winding symbols until the active blade returns to the right side of the knot. During
117 these $2k - 1$ symbols, we need to transition one more step around the sequence of regions. The
118 transitions $\texttt{W}$ and $\texttt{T}$ are generator and inverse for the cyclic group of order 3, concluding the
119 proof.   □

120   It is worth noticing here that a particular point along a winding can be simultaneously valid
121 for both a k-fold and an m-fold tuck for $k \neq m$. One example would be in the winding string
122 $\texttt{TWTT}$: ending with $\texttt{TT}$, it is a valid site for a 1-fold tuck producing $\texttt{TWTTU}$, and since $\texttt{TWTT}$ starts
123 with $\texttt{T}$ and has 2 more $\texttt{T}$ than $\texttt{W}$, it is also a valid site for a 2-fold tuck producing $\texttt{TWTTUU}$. We
124 will revisit this example below, in Section 4.2.

125   We may notice that with the usual physical constraints on a tie – where we have experimen-
126 tally established that broad blade ties tend to be bounded by 9 moves, and thin blade ties by
127 15 moves, we can expect that no meaningful tuck deeper than 7 will ever be relevant; 4 for the
128 broad blade ties. The bound of 4 is achieved in the enumeration by Fink and Mao [4].

129   In our enumeration, we will for the sake of comfort focus on ties up to 13 moves.

---

[4]Recall that the Kleene star F^* is used to denote sequences of 0 or more repetitions of the string F.
[5]Say, as seen on the mirror image. Changing this convention does not change the count, as long as the change
is consequently done.
[6]$\texttt{T}$ for Turnwise
[7]$\texttt{W}$ for Widdershins

4 Language complexity

In this section, we examine the complexity features of the tie-knot language. Due to the constraints we have already observed on the cardinality of W and T, we will define a grammar for this language. We will write this grammar with a Backus-Naur form. Although in practice it is only possible to realise finite strings in the tie-knot language due to the physical properties of fabric, we assume an arbitrarily long (but finite), infinitely thin tie.

4.1 Single-depth tucks

The classical Fink and Mao system has a regular grammar, given by

$$\langle tie \rangle ::= \texttt{L}\langle L \rangle$$
$$\langle lastR \rangle ::= \texttt{L}\langle lastL \rangle \,|\, \texttt{C}\langle lastC \rangle \,|\, \texttt{LCU}$$
$$\langle lastL \rangle ::= \texttt{R}\langle lastR \rangle \,|\, \texttt{C}\langle lastC \rangle \,|\, \texttt{RCU}$$
$$\langle lastC \rangle ::= \texttt{L}\langle lastL \rangle \,|\, \texttt{R}\langle lastR \rangle$$

We use the symbol $\langle lastR \rangle$ to denote the rule that describes what can happen when the last move seen was an R. Hence, at any step in the grammar, some tie knot symbol is emitted, and the grammar continues from the state that that symbol was the last symbol emitted.

The above grammar works well if the only tucks to appear are at the end. For intermediate tucks, and to avoid tucks to be placed at the back of the knot (obeying $\mathcal{T}ie_3$), we would need to keep track of winding parity: tucks are only valid an even number of winding steps from the end.

We can describe this with a regular grammar. For the full tie-knot language, the grammar will end up context-free, as we will see in Section 4.2.

$$\langle tie \rangle ::= \langle prefix \rangle (\langle pair \rangle \,|\, \langle tuck \rangle) * \langle tuck \rangle$$
$$\langle prefix \rangle ::= \texttt{T} \,|\, \texttt{W} \,|\, \epsilon$$
$$\langle pair \rangle ::= (\texttt{T}|\texttt{W})(\texttt{T}|\texttt{W})$$
$$\langle tuck \rangle ::= \texttt{TTU} \,|\, \texttt{WWU}$$

The distribution of T and W varies by type of knot: for classical knots, $\#W - \#T = 2$ (mod 3); for modern knots that tuck to the right, $\#W - \#T = 1$ (mod 3); and for modern knots that tuck to the left, $\#W - \#T = 0$ (mod 3). This grammar does not discriminate between these three sub-classes. In order to track these sub-classes, the RLC-notation is easier.

In order to rebuild this grammar to one based on the RLC-notation, note that from L a T takes us to C and a W takes us to R. So from a $\langle lastT \rangle$ residing at L, we have the options: W to R, T to C, or TU to C. In particular, there is a $\langle lastT \rangle$ at L if we arrived from R. Hence, the TU option can be seen as being a TTU option executing from the preceding R state.

There is thus, at any given position in the tie sequence, the options of proceeding with a T or a W, or to proceed with one of TTU or WWU. In the latter two cases, we can also accept the string. Starting at L, these options take us – in order – to C, to R, to CRU and to RCU respectively. This observation extends by symmetry to all stages, giving the grammar below.

$$\langle lastR\rangle ::= \text{LR}\langle lastR\rangle \,|\, \text{CR}\langle lastR\rangle \,|\, \text{LC}\langle lastC\rangle \,|\, \text{CL}\langle lastL\rangle \,|$$
$$\text{LCU}[\langle lastC\rangle] \,|\, \text{CLU}[\langle lastL\rangle]$$
$$\langle lastL\rangle ::= \text{RL}\langle lastL\rangle \,|\, \text{CL}\langle lastL\rangle \,|\, \text{RC}\langle lastC\rangle \,|\, \text{CR}\langle lastR\rangle \,|$$
$$\text{RCU}[\langle lastC\rangle] \,|\, \text{CRU}[\langle lastR\rangle]$$
$$\langle lastC\rangle ::= \text{LC}\langle lastC\rangle \,|\, \text{RC}\langle lastC\rangle \,|\, \text{LR}\langle lastR\rangle \,|\, \text{RL}\langle lastL\rangle \,|$$
$$\text{LRU}[\langle lastR\rangle] \,|\, \text{RLU}[\langle lastL\rangle]$$
$$\langle tie\rangle ::= \text{L}(\langle lastL\rangle \,|\, R\langle lastR\rangle \,|\, C\langle lastC\rangle)$$

159 By excluding some the exit rules, this allows us to enumerate novel tie-knots with a specific
160 ending direction, which will be of interest later on.

161 ## 4.2   Recursive tucks

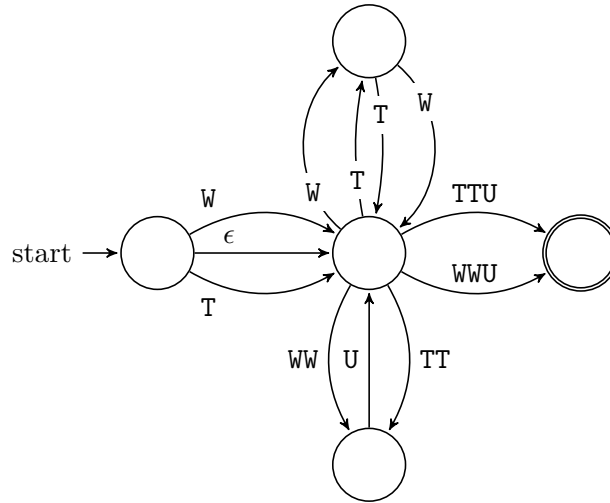
162 We can write a context-free grammar for the arbitrary depth tuck tie-knots.

$$\langle tie\rangle ::= \langle prefix\rangle(\langle pair\rangle \,|\, \langle tuck\rangle) * \langle tuck\rangle$$
$$\langle prefix\rangle ::= \text{T} \,|\, \text{W} \,|\, \epsilon$$
$$\langle pair\rangle ::= (\text{T}|\text{W})(\text{T}|\text{W})$$
$$\langle tuck\rangle ::= \langle ttuck2\rangle \,|\, \langle wtuck2\rangle$$
$$\langle ttuck2\rangle ::= \text{TT}\langle w0\rangle\text{U} \,|\, \text{TW}\langle w1\rangle\text{U}$$
$$\langle wtuck2\rangle ::= \text{WW}\langle w0\rangle\text{U} \,|\, \text{WT}\langle w2\rangle\text{U}$$
$$\langle w0\rangle ::= \text{WW}\langle w1\rangle\text{U} \,|\, \text{WT}\langle w0\rangle\text{U} \,|\, \text{TW}\langle w0\rangle\text{U}|\text{TT}\langle w2\rangle\text{U} \,|$$
$$\langle ttuck2\rangle\text{'}\langle w2\rangle\text{U} \,|\, \langle wtuck2\rangle\text{'}\langle w1\rangle\text{U} \,|\, \epsilon$$
$$\langle w1\rangle ::= \text{WW}\langle w2\rangle\text{U} \,|\, \text{WT}\langle w1\rangle\text{U} \,|\, \text{TW}\langle w1\rangle\text{U}|\text{TT}\langle w0\rangle\text{U} \,|$$
$$\langle ttuck2\rangle\text{'}\langle w0\rangle\text{U} \,|\, \langle wtuck2\rangle\text{'}\langle w2\rangle\text{U}$$
$$\langle w2\rangle ::= \text{WW}\langle w0\rangle\text{U} \,|\, \text{WT}\langle w2\rangle\text{U} \,|\, \text{TW}\langle w2\rangle\text{U}|\text{TT}\langle w1\rangle\text{U} \,|$$
$$\langle ttuck2\rangle\text{'}\langle w1\rangle\text{U} \,|\, \langle wtuck2\rangle\text{'}\langle w0\rangle\text{U}$$

163 Note that the validity of a tuck depends only on the count of T and W in the entire sequence
164 comprising the tuck, and not the validity of any tucks recursively embedded into it. For instance,
165 TWTT is a valid depth-2-tuckable sequence, as is its embedded depth-1-tuckable sequence TT.
166 However, TTWT is also a valid depth-2-tuckable sequence, even though WT is not a valid depth-
167 1-tuckable sequence.
168 We introduce the symbol ' to delineate different tucks that may come in immediate sequence,
169 such as happens in the tie knot TWTTU'UU.

170 ## 4.3   Classification of the tie-knot language

171 If we limit our attention to only the single-depth tie-knots described in Section 4.1, then the
172 grammar is regular, proving that this tie language is a regular language and can be described
173 by a finite automaton. In particular this implies that the tie-knot language proposed by Fink
174 and Mao [4] is regular. In fact, an automaton accepting these tie-knots is given by:

175

After the prefix, execution originates at the middle node, but has to go outside and return before the machine will accept input. This maintains the even length conditions required by $\mathcal{T}ie_3$.

As for the deeper tucked language in Section 4.2, the grammar we gave shows it to be at most context-free. Whether it is exactly context-free requires us to exclude the existence of a regular grammar.

Theorem 2. *The deeper tucked language is context-free.*

Proof. Our grammar in Section 4.2 already shows that the language for deeper tucked tie-knots is either regular or context-free: it produces tie-knot strings with only single non-terminal symbols to the left of each production rule.

It remains to show that this language cannot be regular. To do this, we use the pumping lemma for regular languages. Recall that the pumping lemma states that for every regular language there is a constant p such that for any word w of length at least p, there is a decomposition $w = xyz$ such that $|xy| \leq p$, $|y| \geq 1$ and $xy^i z$ is a valid string for all $i > 0$.

Since the reverse of any regular language is also regular, the pumping lemma has an alternative statement that requires $|yz| \leq p$ instead. We shall be using this next.

Suppose there is such a p. Consider the tie-knot $\texttt{TTW}^{6q-2}\texttt{U}^{3q}$ for some $q > p/3$. Any decomposition such that $|yz| \leq p$ will be such that y and z consist of only $\texttt{U}$ symbols. In particular y consists of only $\texttt{U}$ symbols. Hence, for sufficiently large values of i, there are too few preceding $\texttt{T}/\texttt{W}$-symbols to admit that tuck depth.

Hence the language is not regular. □

5   Enumeration

We can cut down the enumeration work by using some apparent symmetries. Without loss of generality, we can assume that a tie-knot starts by putting the active blade in region $\texttt{R}$: any knot starting in the region $\texttt{L}$ is the mirror image of a knot that starts in $\texttt{R}$ and swaps all $\texttt{W}$ to $\texttt{T}$ and vice versa.

5.1   Generating functions

Generating functions have proven a powerful method for enumerative combinatorics. One very good overview of the field is provided by the textbooks by Stanley [10, 11]. Their relevance to formal languages is based on a paper by Chomsky and Schützenberg [1] that studied context-free grammars using formal power series. More details will appear in the (yet unpublished) Handbook AutoMathA [5].

A generating function for a series a_n of numbers is a formal power series $A(z) = \sum_{j=0}^{\infty} a_j z^j$ such that the coefficient of the degree k term is precisely a_k. Where a_k and b_k are counts of "things of size k" of type a or b respectively, the sum of the corresponding generating functions is the count of "things of size k" across both categories. If gluing some thing of type a with size j to some thing of type b with size k produces a thing of size $j + k$, then the product of the generating functions measures the counts of things you get by gluing things together between the two types.

For our necktie-knot grammars, the sizes are the winding lengths of the ties, and it is clearly the case that adding a new symbol extends the size (thus is a multiplication action), and taking either one or another rule extends the items considered (thus is an additive action).

The `Maple`[8] package `combstruct` has built-in functions for computing a generating function from a grammar specification. Using this, and the grammars we state in Section 4.1, we are able to compute generating functions for both the winding counts and the necktie counts for both Fink and Mao's setting and the single-depth tuck setting.

- The generating function for Fink and Mao necktie-knots is

$$\frac{z^3}{(1+z)(1-2z)} = z^3 + z^4 + 3z^5 + 5z^6 + 11z^7 + 21z^8 + 43z^9 + O(z^{10})$$

- The generating function for single tuck necktie-knots is

$$\frac{2z^3(2z+1)}{1-6z^2} = 2z^3 + 4z^4 + 12z^5 + 24z^6 + 72z^7 + 144z^8 + 432z^9 +$$
$$864z^{10} + 2\,592z^{11} + 5\,184z^{12} + 15\,552z^{13} + O(z^{14})$$

- By removing final states from the BNF grammar, we can compute corresponding generating functions for each of the final tuck destinations.
  For an `R`-final tuck, we remove all final states except for `CRU` and `LRU`, making the non-terminal symbol mandatory for all other tuck sequences. For `L`, we remove all but `CLU` and `RLU`. For `C`, we remove all but `RCU` and `LCU`. This results in the following generating functions for `R`-final, `L`-final and `C`-final sequences, respectively.

$$\frac{z^3(2z^3 - 2z^2 + z + 1)}{1-6z^2} = z^3 + z^4 + 4z^5 + 8z^6 + 24z^7 + 48z^8 + 144z^9 +$$
$$288z^{10} + 864z^{11} + 1\,728z^{12} + 5\,184z^{13} + O(z^{14})$$

$$\frac{2z^4(2z^2 - 2z - 1)}{1-6z^2} = 2z^4 + 4z^5 + 8z^6 + 24z^7 + 48z^8 + 144z^9 +$$
$$288z^{10} + 864z^{11} + 1\,728z^{12} + 5\,184z^{13} + O(z^{14})$$

$$\frac{z^3(2z^3 - 2z^2 + z + 1)}{1-6z^2} = z^3 + z^4 + 4z^5 + 8z^6 + 24z^7 + 48z^8 + 144z^9 +$$
$$288z^{10} + 864z^{11} + 1\,728z^{12} + 5\,184z^{13} + O(z^{14})$$

- Removing the references to the tuck move, we recover generating functions for the number of windings available for each tie length. We give these for `R`-final, `L`-final and `C`-final

---

[8]Maple is a trademark of Waterloo Maple Inc. The computations of generating functions in this paper were performed by using Maple.

respectively. Summed up, these simply enumerate all possible `T`/`W`-strings of the corre-
sponding lengths, and so run through powers of 2.

$$\frac{z^3}{1-z-2z^2} = z^3+z^4+3z^5+5z^6+11z^7+21z^8+43z^9+85z^{10}+171z^{11}+341z^{12}+683z^{13}+O(z^{14})$$

$$\frac{2z^4}{(1-2z)(1+z)} = 2z^4+2z^5+6z^6+10z^7+22z^8+42z^9+86z^{10}+170z^{11}+342z^{12}+682z^{13}+O(z^{14})$$

$$\frac{z^3}{1-z-2z^2} = z^3+z^4+3z^5+5z^6+11z^7+21z^8+43z^9+85z^{10}+171z^{11}+341z^{12}+683z^{13}+O(z^{14})$$

- For the full grammar of arbitrary depth knots, we set w to be a root of $(8z^6 - 4z^4)\zeta^3 + (-8z^6+18z^4 - 7z^2)\zeta^2 + (-16z^6+14z^4 - 6z^2+2)\zeta - 12z^4+9z^2 - 2 = 0$ solved for ζ. Then the generating function for this grammar is:

$$-\frac{1}{8z^4 - 11z^2+3}(64w^2z^7 - 128wz^7 + 32w^2z^6 - 64wz^6 - 48z^5w^2 + 216z^5w - 24w^2z^4-$$
$$-96z^5+108wz^4+8w^2z^3 - 48z^4 - 110wz^3+4w^2z^2+82z^3 - 55z^2w+41z^2+16zw - 16z+8w - 8) =$$
$$2z^2 + 4z^3 + 20z^4 + 40z^5 + 192z^6 + 384z^7 + 1896z^8 + 3792z^9+$$
$$19320z^{10} + 38640z^{11} + 202392z^{12} + 404784z^{13} + 2169784z^{14} + O(z^{15})$$

5.2 Tables of counts

For ease of reading, we extract the results from the generating functions above to more easy-to-reference tables here. Winding length throughout refers to the number of `R`/`L`/`C` symbols occurring, and thus is 1 larger than the `W`/`T` count.

The cases enumerated by Fink and Mao [3] are

Winding length	2	3	4	5	6	7	8	total
# tie-knots	1	1	3	5	11	21	43	85

A knot with the thick blade active will cover up the entire knot with each new bow. As such, all thick blade active tie-knots will fall within the classification by Fink and Mao [3].

The modern case, thus, deals with thin blade active knots. As evidenced by the Trinity and the Eldredge knots, thin blade knots have a wider range of interesting façades and of interesting tuck patterns. For thick blade knots, it was enough to assume that the tuck happens last, and from the C region, the thin blade knots have a far wider variety.

The case remains that unless the last move is a tuck – or possibly finishes in the `C` region – the knot will unravel from gravity. We can thus expect this to be a valid requirement for the enumeration. There are often more valid tuck sites than the final position in a knot, and the tuck need no longer come from the `C` region: `R` and `L` are at least as valid.

The computations in Section 5.1 establish

Winding length	3	4	5	6	7	8	9	10	11	12	13	total
# left windings	0	2	2	6	10	22	42	86	170	342	682	1 364
# right windings	1	1	3	5	11	21	43	85	171	341	683	1 365
# center windings	1	1	3	5	11	21	43	85	171	341	683	1 365
# left knots	0	2	4	8	24	48	144	288	864	1 728	5 184	8 294
# right knots	1	1	4	8	24	48	144	288	864	1 728	5 184	8 294
# center knots	1	1	4	8	24	48	144	288	864	1 728	5 184	8 294
# single tuck knots	2	4	12	24	72	144	432	864	2 592	4 146	15 552	24 882
total # knots	2	4	20	40	192	384	1 896	3 792	19 320	38 640	202 392	266 682

248 The first point where the singly tucked knots and the full range of knots deviate is at the
249 knots with winding length 4; there are 12 singly tucked knots, and 8 knots that allow for a
250 double tuck, namely:

```
TTTTU    TTWWU    TWTTU    TWWWU
WTTTU    WTWWU    WWTTU    WWWWU
```

251
```
TTUTTU   TTUWWU   WWUTTU   WWUWWU
TTTWUU   TTWTUU   TWTTUU   TWTTU'UU
WTWWUU   WTWWU'UU  WWTWUU   WWWTUU
```

252 The reason for the similarity between the right and the center counts is that the winding
253 sequences can be mirrored. Left-directed knots are different since the direction corresponds to
254 the starting direction. Hence, a winding sequence for a center tuck can be mirrored to a winding
255 sequence for a right tuck.

256 In the preprint version of this paper, we claimed the total count of knots using only single-
257 depth tucks to be $177\,147$. During the revision of the paper, we have discovered two errors in
258 this claim:

259 1. There is an off-by-one error in this count.

260 2. This count was done for tie-knots that allow tucks that are hidden behind the knot.
   Adding this extra space to the generating grammar produces the generating function

$$2z^3 + 6z^4 + 18z^5 + 54z^6 + 162z^7 + 486z^8 + 1\,458z^9 +$$
$$4\,374z^{10} + 13\,122z^{11} + 39\,366z^{12} + 118\,098z^{13} + O(z^{14})$$

260 with a total of $177\,146$ tie-knots with up to 13 moves.

6 Aesthetics

262 Fink and Mao [3] propose several measures to quantify the aesthetic qualities of a necktie-knot;
263 notably *symmetry* and *balance*, corresponding to the quantities $\#R - \#L$ and the number of
264 transitions from a streak of W to a streak of T or vice versa.

265 By considering the popular thin-blade neck tie-knots: the Eldredge and the Trinity, as
266 described in [7, 8], we can immediately note that balance no longer seems to be as important
267 for the look of a tie-knot as is the shape of its façade. Symmetry still plays an important role
268 in knots, and is easy to calculate using the CLR notation for tie-knots.

269

Knot	TW-string	CLR-string	Balance	Symmetry
Eldredge	TTTWWTTUTTWWU	LCRLRCRLUCRCLU	3	0
Trinity	TWWWTTTUTTU	LCLRCRLCURLU	2	1

270 We do not in this paper attempt to optimize any numeric measures of aesthetics, as this
271 would require us to have a formal and quantifiable measure of the knot façades. This seems
272 difficult with our currently available tools.

7 Conclusion

274 In this paper, we have extended the enumeration methods originally used by Fink and Mao [3]
275 to provide a larger enumeration of necktie-knots, including those knots tied with the thin blade
276 of a necktie to produce ornate patterns in the knot façade.

277 We have found $4\,094$ winding patterns that take up to 13 moves to tie and are anchored by
278 a final single depth tuck, and thus are reasonable candidates for use with a normal necktie. We
279 chose the number of moves by examining popular thin-blade tie-knots – the Eldredge tie-knot

uses 12 moves – and by experimentation with our own neckties. Most of these winding patterns allow several possible tuck patterns, and thus the 4 094 winding patterns give rise to 24 882 singly tucked tie-knots.

We have further shown that in the limit, the language describing neck tie-knots is context free, with a regular sub-language describing these 24 882 knots.

These counts, as well as the stated generating functions, are dependent on the correctness of the `combstruct` package in Maple, and the correctness of our encoding of these grammars as Maple code. We have checked the small counts and generated strings for each of the grammars against experiments with a necktie and with the results by Fink and Mao and our own catalogue.

Questions that remain open include:

- Find a way to algorithmically divide a knot description string into a body/façade distinction.

- Using such a distinction, classify all possible knot façades with reasonably short necktie lengths.

We have created a web-site that samples tie-knots from knots with at most 12 moves and displays tying instructions: `http://tieknots.johanssons.org`. The entire website has also been deposited with Figshare[12].

All the code we have used, as well as a table with assigned names for the 2 046 winding patterns for up to 12 moves are provided as supplementary material to this paper. Winding pattern names start with `R`, `L` or `C` depending on the direction of the final tuck, and then an index number within this direction. We suggest augmenting this with the bit-pattern describing which internal tucks have been added – so that e.g. the Eldredge would be L-373.4 (including only the 3rd potential tuck from the start) and the Trinity would be L-110.2 (including only the 2nd potential tuck). Thus, any single-depth tuck can be concisely addressed.

8   Acknowledgements

We would like to thank the reviewers, whose comments have gone a long way to make this a far better paper, and who have caught several errors that marred not only the presentation but also the content of this paper.

Reviewer 1 suggested a significant simplification of the full grammar in Section 4.2, which made the last generating function at all computable in reasonable time and memory.

Reviewer 2 suggested we look into generating functions as a method for enumerations. As can be seen in Section 5.1, this suggestion has vastly improved both the power and ease of most of the results and calculations we provide in the paper.

For these suggestions in particular and all other suggestions in general we are thankful to both reviewers.

References

[1] Noam Chomsky and Marcel P Schützenberger. "The algebraic theory of context-free languages". In: *Studies in Logic and the Foundations of Mathematics* 26 (1959), pp. 118–161.

[2] T. Fink and Y. Mao. *The 85 ways to tie a tie*. Fourth Estate, 2001.

[3] T. Fink and Y. Mao. "Tie knots, random walks and topology". In: *Physica A: Statistical Mechanics and its Applications* 276.1 (2000), pp. 109–121.

[4] T.M. Fink and Y. Mao. "Designing tie knots by random walks". In: *Nature* 398.6722 (1999), pp. 31–32.

[5] Hermann Gruber, Jonathan Lee, and Jeffrey Shallit. "Enumerating regular expressions and their languages". In: *arXiv preprint arXiv:1204.4982* (2012).

[6] A. Knorr. *eldredge reloaded*. Blog post on `http://xirdalium.net`. Accessed on 26 December 2012. June 2010. URL: `http://xirdalium.net/2010/06/20/eldredge-reloaded/`.

[7] A. Krasny. *Eldredge Tie Knot - How to Tie a Eldredge Necktie Knot*. Blog post on `http://agreeordie.com`. Accessed on 26 December 2012. Nov. 2012. URL: `http://agreeordie.com/blog/musings/545-how-to-tie-a-necktie-eldredge-knot`.

[8] A. Krasny. *Trinity Tie Knot - How to Tie a Trinity Necktie Knot*. Blog post on `http://agreeordie.com`. Accessed on 26 December 2012. Dec. 2012. URL: `http://agreeordie.com/blog/musings/553-how-to-tie-a-necktie-trinity-knot`.

[9] Michael Sipser. *Introduction to the Theory of Computation*. Vol. 2. Thomson Course Technology Boston, 2006.

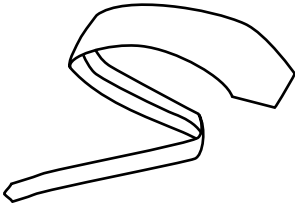
[10] Richard P Stanley. *Enumerative combinatorics*. Vol. 1. Cambridge Studies in Advanced Mathematics 49. Cambridge University Press, Cambridge, 1997, p. 86.

[11] Richard P Stanley. *Enumerative combinatorics*. Vol. 2. Cambridge Studies in Advanced Mathematics 62. Cambridge University Press, Cambridge, 1999.

[12] Mikael Vejdemo-Johansson. "Random Tie Knots webpage". In: (Feb. 2015). URL: `http://dx.doi.org/10.6084/m9.figshare.1300138`.

[13] A. Wachowski et al. *The matrix reloaded*. Warner Bros. Pictures, 2003.

Fig 1A - Anatomy of a tie

The parts of a necktie, and the division of the wearer's torso with the regions (Left, Center Right) and the winding directions (Turnwise, Widdershins) marked out for reference.

Broad
blade

Thin
blade

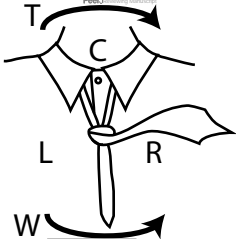
Fig 1B - Left/Center/Right

The parts of a necktie, and the division of the wearer's torso with the regions (Left, Center Right) and the winding directions (Turnwise, Widdershins) marked out for reference.

Fig 2A - 4 in hand

Different examples of tie knots. Left, a 4-in-hand; middle, a double windsor; right a trinity. The 4-in-hand and double windsor share the flat façade but have different bodies producing different shapes. The trinity has a completely different façade, produced by a different wind and tuck pattern.

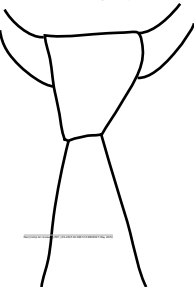



Fig 2B - Double Windsor

Different examples of tie knots. Left, a 4-in-hand; middle, a double windsor; right a trinity. The 4-in-hand and double windsor share the flat façade but have different bodies producing different shapes. The trinity has a completely different façade, produced by a different wind and tuck pattern.

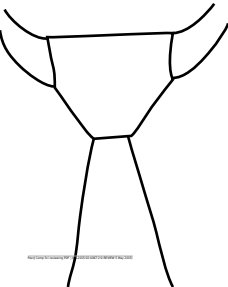



Fig 2C - Trinity

Different examples of tie knots. Left, a 4-in-hand; middle, a double windsor; right a trinity. The 4-in-hand and double windsor share the flat façade but have different bodies producing different shapes. The trinity has a completely different façade, produced by a different wind and tuck pattern.