

FIXED-PARAMETER TRACTABLE COMPUTATION OF KNOT POLYNOMIALS OF RESHETIKHIN–TURAEV TYPE VIA TENSOR NETWORKS

SHANA YUNSHENG LI

ABSTRACT. We give a thorough analysis of the time complexity of computing knot polynomials of Reshetikhin–Turaev type via tensor contractions on the associated tensor networks, showing that the computation is fixed-parameter tractable with respect to a parameter that is at most linear in the treewidth of the input knot diagram. When combined with an existing algorithm for approximate tree decomposition, this recovers the sub-exponential bound $e^{\mathcal{O}(\sqrt{n})}$ for the time complexity of computing any knot polynomial of Reshetikhin–Turaev type.

CONTENTS

1. Introduction	1
2. Setup	4
2.1. Long knot diagrams and rotation numbers	4
2.2. The Reshetikhin–Turaev functor	5
2.3. Tensor network and contractions	6
3. Bounding the time complexity	8
3.1. Contraction-width	8
3.2. Proof of Theorem 1.2 and Theorem 1.3	9
Acknowledgements	10
References	10

1. INTRODUCTION

Knot polynomials are one of the most important kinds of knot invariants, and have been widely studied in knot theory ever since the first knot polynomial, the Alexander polynomial [Ale28], was discovered. They are powerful tools for distinguishing knots [HTW98, Bur20, Thi25], and have deep connections with various areas of mathematics and physics. As is the case for many invariants, efficient computation is essential in providing insights for studying their properties [BN05, Shu11, GL25, GLY], and is thus naturally an important question. Although there exist knot polynomials with intriguing properties that can be computed in polynomial time [BNvdV19, BNvdV25], the majority of knot polynomials adopt definitions via the Reshetikhin–Turaev functor [RT90, Tur88, GK26], and are generally $\#P$ -hard to

Date: March 25, 2026.

Key words and phrases: knots, knot polynomials, Reshetikhin–Turaev functor, fixed-parameter tractable, algorithm, computational topology, quantum topology .

compute [JVW90]. For $\#P$ -hard problems, the best algorithm one could expect is *fixed-parameter tractable*, which in general means that the input can be subdivided into families \mathcal{F}_i parametrized by $i \in \mathbb{N}$, such that the time complexity for an input in \mathcal{F}_i of size n is

$$O(f(i) \cdot p(n)), \quad (1)$$

where p is a polynomial and f is an arbitrary computable function. An algorithm satisfying (1) is *fixed-parameter tractable with respect to i* .

For computations of knot invariants, i is often a graph-theoretical width of planar diagrams of knots. Some relevant examples of widths are the *tree-width*, *cut-width*, *carving-width*, and, proposed in this paper, *contraction-width*. Among them, tree-width is the most studied one, hence we briefly recall the definition of tree-width below and list its relationship with other widths in Table 1. The definition of contraction-width is given in Section 3.1.

Definition 1.1. A tree decomposition of a graph $\Gamma = (V, E)$ is a tree T whose vertices consist of subsets X_i of V such that

- (1) $\bigcup_i X_i = V$.
- (2) If $v \in X_i \cap X_j$, then any X_k in the path connecting X_i and X_j also contains v .
- (3) For every edge $(v, w) \in E$, there exists X_i such that $v, w \in X_i$.

The width of a tree decomposition is $\max_i |X_i| - 1$. The *tree-width* of Γ is the minimum width among all possible tree decompositions of Γ .

Name of width	Notation	Relationship with tree-width	Asymptotic behaviour	Reference
Tree-width	$tw(\Gamma)$	itself	$O(\sqrt{n})$	[LT79]
Cut-width	$c(\Gamma)$	$c(\Gamma) = O(tw(\Gamma) \cdot \Delta(\Gamma) \cdot \log n)$	$O(\sqrt{n} \log n)$	[KS93]
Carving-width	$cw(\Gamma)$	$\frac{2}{3}(tw(\Gamma) + 1) \leq cw(\Gamma) \leq \Delta(\Gamma) \cdot (tw(\Gamma) + 1)$	$O(\sqrt{n})$	[Bie90]
Contraction-width	$w(\Gamma)$	$w(\Gamma) \leq \Delta(\Gamma) \cdot (tw(\Gamma) + 1)$	$O(\sqrt{n})$	Lemma 1.4

TABLE 1. Different widths and their relationship with the tree-width, where Γ is a planar graph with n vertices and maximum degree $\Delta(\Gamma)$.

For fixed-parameter tractable algorithms discussed in this paper, we have $f(i) = e^{O(i)}$ and $i = O(\sqrt{n})$ in terms of (1). Previously, fixed-parameter tractable algorithms were established for the Jones polynomial [Jon87, Mak05] and then for all Reshetikhin–Turaev knot polynomials [Mar21], and the latter is fixed-parameter tractable with respect to the carving-width. It follows that the algorithm in [Mar21] gives a sub-exponential bound $e^{O(\sqrt{n})}$ for computing Reshetikhin–Turaev knot polynomials. However, the algorithm in [Mar21] involves classifying morphism compositions into 7 elementary pieces and dealing with them case by case. To the best of the author’s knowledge, no publicly available implementation of the algorithm in [Mar21] exists as of this writing.

On the other hand, the computation of knot polynomials of Reshetikhin–Turaev type naturally translates into tensor contractions on the corresponding tensor networks. Tensor networks have been intensively studied in recent years [MS08, Orfrm[o]–9], and implementation

of tensor contractions is straightforward. Experiments have revealed this kind of computation to be quite efficient for computing knot polynomials [MK19, GL25]. The author gave an implementation in [GL24] accompanying [GL25] using a heuristic method for obtaining contraction sequences, with which the V_2 -polynomial was computed for millions of knots including all those up to 16 crossings. However, no thorough analysis of time complexity of such computation of polynomials has appeared in the literature.

In this note, we show that the computation of Reshetikhin–Turaev knot polynomials via tensor networks is in fact fixed-parameter tractable with respect to the contraction-width, and recovers the sub-exponential bound given in [Mar21]. The contraction-width, defined in Section 3.1, is formalized by the author from the practice in [GL25], and naturally leads to the results here. More precisely, we show the following:

Theorem 1.2. *Given an oriented knot diagram D with n crossings and a contraction sequence on D with contraction-width w , for any k -variable knot polynomial RT of Reshetikhin–Turaev type, computing $RT(D)$ by contracting the given sequence is*

$$O(d^w n^{k+2} (\log n)^3), \tag{2}$$

where d is the dimension of V and V is a vector space such that the R -matrix for RT lives in $\text{Aut}(V \otimes V)$.

With minimal tweaks, the code in [GL24] turns into an implementation of Theorem 1.2. The author is also working on implementing an improved version of [GL24] in SnapPy [CDGW]. For practical estimation of running time, we give a more detailed version of Theorem 1.2 as follows

Theorem 1.3. *Given an oriented knot diagram D with n crossings and a contraction sequence on D with contraction-width w , for any k -variable knot polynomial RT of Reshetikhin–Turaev type, computing $RT(D)$ by contracting the given sequence is proportionally controlled by*

$$d^w \cdot k(4nM)^k (2n + M^k)n(\log(ndCM^k))^2 \log(4nM + 1), \tag{3}$$

where d is the dimension of V , V is a vector space such that the R -matrix R and ribbon element h for RT live in $\text{Aut}(V \otimes V)$ and $\text{Aut}(V)$ respectively, C is the maximum absolute value of the coefficients of the entries in $(R_{e_i \otimes e_j}^{e_k \otimes e_l})$ and $(h_{e_i}^{e_j})$, and M is the maximum degree of all variables in the entries in $(R_{e_i \otimes e_j}^{e_k \otimes e_l})$ and $(h_{e_i}^{e_j})$.

By proportionally controlled we mean that there exists a universal constant U such that computing $RT(D)$ is no more expensive than U times (3) for any RT and D .

The following is a direct consequence of [MS08, Lemma 4.4]:

Lemma 1.4. For any graph Γ ,

$$w(\Gamma) \leq \Delta(\Gamma) \cdot (tw(\Gamma) + 1), \tag{4}$$

where $\Delta(\Gamma)$ is the maximum degree of Γ .

Combining [MS08, Proposition 4.2] and [Kor22], we can drop the contraction sequence from the input of Theorem 1.2, obtaining

Corollary 1.5. Given an oriented knot diagram D with n crossings, for any k -variable knot polynomial RT of Reshetikhin–Turaev type, computing $RT(D)$ is

$$O(d^{2w(D)}n^{k+2}(\log n)^3 + 2^{10.8w(D)}n), \quad (5)$$

where d is the dimension of V and V is a vector space such that the R -matrix for RT lives in $\text{Aut}(V \otimes V)$.

Applying Lemma 1.4 and substituting $tw(D)$ with $O(\sqrt{n})$, this translates to

Corollary 1.6. Given an oriented knot diagram D with n crossings, any knot polynomial of Reshetikhin–Turaev type of D can be computed in time $e^{O(\sqrt{n})}$.

This recovers the sub-exponential bound given in [Mar21]. Note, however, that Theorems 1.2 and 1.3 are not necessarily equivalent to the results in [Mar21], as the author is not aware of a direct relationship between the carving-width and the contraction-width, even though they are both bounded by $\Delta(\Gamma) \cdot (tw(\Gamma) + 1)$ from above.

We give a brief review of the Reshetikhin–Turaev functor in Section 2, explaining how the computation translates into tensor contractions, and then give the detailed analysis for the time complexity in Section 3, proving Theorem 1.2. More details on the material in Section 2 can be found in [GL25]. Throughout this paper, knot polynomials of Reshetikhin–Turaev type are assumed to be given by R -matrices and ribbon elements with matrix representations whose entries are Laurent polynomials with integer coefficients.

We remark that our results easily generalize to links and possibly with differently colored components, but we talk about knots exclusively for simplicity and because “knot polynomial” is a much more common phrase than “link polynomial”.

2. SETUP

We give some basic descriptions about our settings in this section, laying the groundwork for the discussion of time complexity in Section 3.

2.1. Long knot diagrams and rotation numbers. Given a planar diagram of an oriented knot, we convert it into an oriented long knot diagram by cutting an arc open and pulling the two open ends towards infinity in opposite directions without creating any self-intersection. Following the convention in [BNvdV24], we regard the out-pointing strand to be pulled vertically upward, and the other one vertically downward. We further require that the arcs always point upward in sufficiently small neighborhoods of each crossing by applying an ambient isotopy which may rotate the crossings.

The vertical axis now gives us a natural height function, which restricts to a Morse function along the long knot diagram, whose local maxima and minima occur only on the arcs and never at the crossings. The rotation number associated to each arc in the oriented long knot diagram is defined as follows.

Definition 2.1. The *rotation number* of an arc \mathcal{A} in an oriented long knot diagram equipped with a height function ℓ as described above, is the integer

$$\sum_{p \in \{d\ell=0\} \cap \mathcal{A}} (-1)^{\delta_\ell(p)} \varepsilon_{\mathcal{A}}(p),$$

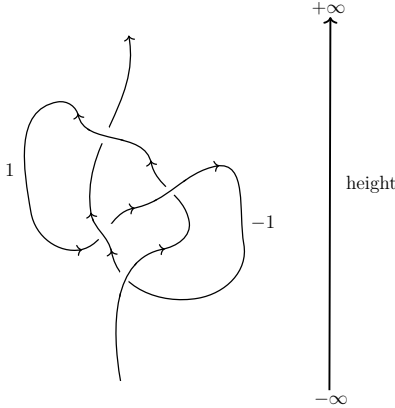


FIGURE 1. An oriented long knot diagram of the 4_1 knot, with nonzero rotation numbers labeled.

where $\delta_\ell(p) = 1$ if p is a local maximum of ℓ and 0 otherwise; $\varepsilon_{\mathcal{A}}(p) = 1$ if the oriented arc \mathcal{A} points to the right at p , and 0 otherwise.

Note that the rotation number is determined not only by the combinatorial information of the oriented long knot diagram, but also by how the diagram is embedded in the plane, inducing the height function. Given a planar diagram of an oriented knot with n crossings, there is an $O(n)$ algorithm that converts it into an oriented long knot diagram with rotation numbers in $\{0, \pm 1\}$ [BNvdV24]. A description of the algorithm can be found in [GL25, Section 2.2.1].

2.2. The Reshetikhin–Turaev functor. Depending on the context, the precise constructions of the Reshetikhin–Turaev functor vary, but they all share some common structures, of which only those relevant to computation are described here; we refer the reader to [Oht02, GK26] for in-depth introductions of the theoretical aspects.

The central idea of the Reshetikhin–Turaev functor is to view oriented long knot diagrams as an analogue of 1-dimensional cobordisms, assembled from elementary pieces, and to map the pieces into the category of vector spaces. The composition of the obtained linear maps is thus the desired knot invariant.

Therefore, as with any topological quantum field theory, the Reshetikhin–Turaev functor is determined by its image on those elementary pieces. The information needed consists of:

- A vector space V over a field \mathbb{F} .
- An R -matrix $R \in \text{Aut}(V \otimes V)$.
- A ribbon element $h \in \text{Aut}(V)$.

The Reshetikhin–Turaev functor is then determined as follows:

$$\begin{array}{ccccccc}
\begin{array}{c} \uparrow \\ \mapsto \\ \begin{array}{c} V \\ \uparrow 1_V \\ V \end{array} \end{array} & \begin{array}{c} \curvearrowright \\ \mapsto \\ \begin{array}{c} V \otimes V \\ \uparrow R \\ V \otimes V \end{array} \end{array} & \begin{array}{c} \curvearrowleft \\ \mapsto \\ \begin{array}{c} V^* \otimes V \\ \uparrow u \\ \mathbb{F} \end{array} \end{array} & \begin{array}{c} \curvearrowright \\ \mapsto \\ \begin{array}{c} V \otimes V^* \\ \uparrow u' \\ \mathbb{F} \end{array} \end{array} \\
\begin{array}{c} \downarrow \\ \mapsto \\ \begin{array}{c} V^* \\ \uparrow 1_{V^*} \\ V^* \end{array} \end{array} & \begin{array}{c} \curvearrowleft \\ \mapsto \\ \begin{array}{c} V \otimes V \\ \uparrow R^{-1} \\ V \otimes V \end{array} \end{array} & \begin{array}{c} \curvearrowright \\ \mapsto \\ \begin{array}{c} \mathbb{F} \\ \uparrow n \\ V \otimes V^* \end{array} \end{array} & \begin{array}{c} \curvearrowleft \\ \mapsto \\ \begin{array}{c} \mathbb{F} \\ \uparrow n' \\ V^* \otimes V \end{array} \end{array}
\end{array}$$

where u and u' are defined by

$$u(1) = \sum_i e_i^* \otimes h(e_i), \quad u'(1) = \sum_i e_i \otimes e_i^*,$$

where $\{e_i\}$ is any basis of V and $\{e_i^*\}$ is the dual basis of $\{e_i\}$, and n and n' are defined by

$$n(x \otimes f) = f(h^{-1}(x)), \quad n'(f \otimes x) = f(x)$$

for any $x \in V$ and $f \in V^*$.

Since $(n \otimes 1_V) \circ (1_V \otimes u) = 1_V$ and $(n' \otimes 1_{V^*}) \circ (1_{V^*} \otimes u') = 1_{V^*}$, it is easy to see that the result of applying the Reshetikhin–Turaev functor is determined by the combinatorial information of the planar diagram along with the rotation numbers. With the rotation numbers living in $\{0, \pm 1\}$, the result can be described by $O(n)$ elementary pieces, hence it is $O(n)$ to convert an oriented long knot diagram into compositions of linear maps as above.

The result of applying the Reshetikhin–Turaev functor to an oriented long knot diagram is a diagonalizable linear map in $\text{End}(V)$ whose eigenvalues are all the same. To obtain a knot polynomial, indeterminates are plugged into the definitions of R and h , turning the resulting linear map into a parametrized family, and the eigenvalue into the polynomial.

2.3. Tensor network and contractions. We briefly review some standard terminology of tensor networks and tensor contractions. We refer the reader to [GL25] for slightly more detailed descriptions, and to [BC17] for an extensive and graphical introduction to this topic.

Definition 2.2. Let \mathcal{R} be a ring. An n -tensor T over \mathcal{R} is a tuple $(T_{i_1, \dots, i_n})_{(i_1, \dots, i_n) \in \mathcal{S}}$ where $T_{i_1, \dots, i_n} \in \mathcal{R}$ and the index set \mathcal{S} is of the form

$$\mathcal{S} = \{1, \dots, m_1\} \times \dots \times \{1, \dots, m_n\} \subset \mathbb{N}^n,$$

We call the integer m_k ($k \in \{1, \dots, n\}$) the dimension of the leg k of the tensor T .

Definition 2.3. A tensor network over ring \mathcal{R} is a (multi)graph Γ where each vertex v is associated with a tensor T_v over \mathcal{R} whose legs are bijectively associated with the ends of the edges incident to v . Such an association must satisfy that for each edge in Γ , its two ends are associated with two different legs with the same dimension of one or two tensors.

Given a tensor network Γ over \mathcal{R} , for an n -tensor T and an n' -tensor T' in Γ , if two legs k of T and k' of T' are associated with the two ends of an edge e in Γ , we can contract T

and T' along e to obtain an $(n + n' - 2)$ -tensor T'' , defined by

$$T''_{i_1, \dots, \widehat{i}_k, \dots, i_n, j_1, \dots, \widehat{j}_{k'}, \dots, j_{n'}} := \sum_{i_k = j_{k'} \in \{1, \dots, m_k\}} T_{i_1, \dots, i_k, \dots, i_n} T'_{j_1, \dots, j_{k'}, \dots, j_{n'}}$$

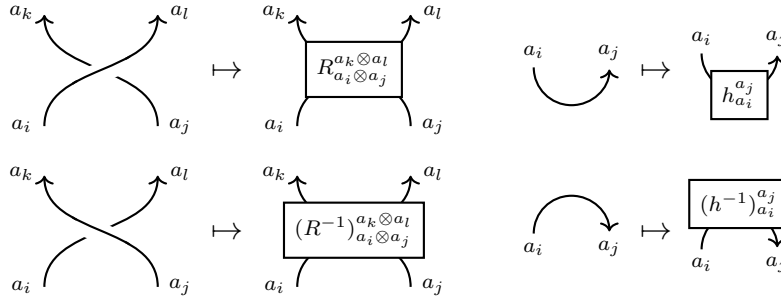
where the hats indicate that the corresponding indices are deleted. Moreover, we can remove the edge e and replace T and T' and their corresponding vertices with T'' and a new vertex, obtaining a tensor network with one fewer vertex and edge than Γ .

More generally, if legs k_1, \dots, k_s of T and legs k'_1, \dots, k'_s of T' are associated with the two ends of edges e_1, \dots, e_s in Γ respectively, we can contract T and T' along e_1, \dots, e_s at once to obtain an $(n + n' - 2s)$ -tensor T'' defined by

$$T''_{i_1, \dots, \widehat{i}_{k_1}, \dots, \widehat{i}_{k_s}, \dots, i_n, j_1, \dots, \widehat{j}_{k'_1}, \dots, \widehat{j}_{k'_s}, \dots, j_{n'}} := \sum_{\substack{i_{k_1} = j_{k'_1} \in \{1, \dots, m_{k_1}\} \\ \vdots \\ i_{k_s} = j_{k'_s} \in \{1, \dots, m_{k_s}\}}} T_{i_1, \dots, i_n} T'_{j_1, \dots, j_{n'}} \quad (6)$$

We can then remove the edges e_1, \dots, e_s and replace T and T' and their corresponding vertices with T'' and a new vertex, obtaining a tensor network with one fewer vertex and s fewer edges than Γ . Similarly, we can define contractions of more than two tensors at once.

Example 2.4. Given an R -matrix $R \in \text{Aut}(V \otimes V)$ and a ribbon element $h \in \text{Aut}(V)$ determining a Reshetikhin–Turaev functor, we fix a basis $\mathcal{B} := \{e_1, \dots, e_d\}$ of V . Under \mathcal{B} , $R^{\pm 1}$ becomes a 4-tensor $(R^{\pm 1})_{e_i \otimes e_j}^{e_k \otimes e_l}$ as (i, j, k, l) varies in $\{1, \dots, d\}^4$. Similarly $h^{\pm 1}$ becomes a 2-tensor $(h^{\pm 1})_{e_i}^{e_j}$. For an oriented long knot diagram with n crossings, the Reshetikhin–Turaev functor turns it into a tensor network by the following assignment:



That is, it turns each crossing into a degree 4 vertex associated with a 4-tensor $(R^{\pm 1})_{e_i \otimes e_j}^{e_k \otimes e_l}$, and each cup or cap going rightwards into a degree 2 vertex associated with a 2-tensor $(h^{\pm 1})_{e_i}^{e_j}$. With rotation numbers in $\{0, \pm 1\}$, we obtain a tensor network with no more than $2n$ vertices, where all legs are of dimension d .

Example 2.5. Figure 2 illustrates the tensor network obtained by applying the Reshetikhin–Turaev functor on an oriented long knot diagram of the 4_1 knot. With the symbols in Figure 2, one can check that the corresponding knot polynomial, as discussed at the end of Section 2.2, can be expressed as the following finite sum:

$$\sum_{\substack{a_1, \dots, a_9 \in \mathcal{B} \\ a_0 = a_{10} = e_1}} R_{a_0 \otimes a_7}^{a_8 \otimes a_1} \cdot h_{a_3}^{a_4} \cdot (R^{-1})_{a_4 \otimes a_8}^{a_9 \otimes a_5} \cdot R_{a_5 \otimes a_1}^{a_2 \otimes a_6} \cdot (h^{-1})_{a_6}^{a_7} \cdot (R^{-1})_{a_9 \otimes a_2}^{a_3 \otimes a_{10}}. \quad (7)$$

Recall that $\mathcal{B} = \{e_1, \dots, e_d\}$ is the basis of V .

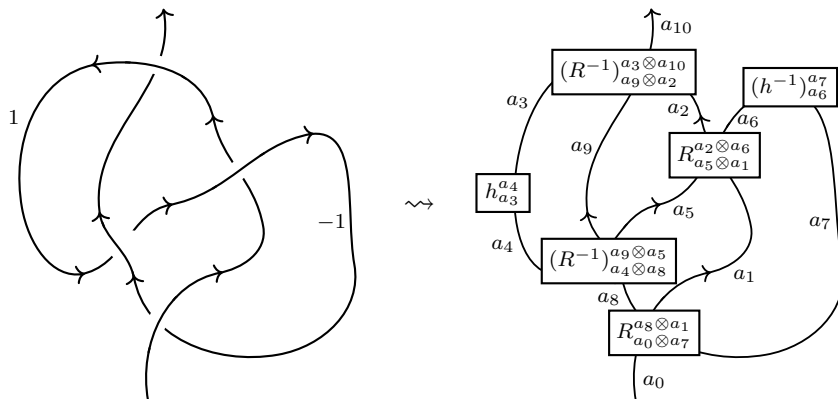


FIGURE 2. The tensor network resulting from an oriented long knot diagram of the 4_1 knot.

Examples 2.4 and 2.5 show that all knot polynomials of Reshetikhin–Turaev type can be expressed as a sum similar to (7). The sum is finite, hence we can compute it piece by piece in an arbitrary order, which naturally translates into doing a sequence of tensor contractions on the tensor network until only one vertex remains. Such a sequence is called a *contraction sequence*. A contraction sequence can be encoded by an ordered list of sets of edges removed at each step, and it plays a crucial role in the time complexity of computing the contractions, which we discuss in detail in the following section. Note that we can eliminate two free legs of the tensor network by setting $a_0 = a_{10} = e_1$ in Example 2.5 at the beginning.

3. BOUNDING THE TIME COMPLEXITY

3.1. Contraction-width. We now restrict ourselves to tensor networks obtained by applying the Reshetikhin–Turaev functor to oriented long knot diagrams, where all legs of tensors have dimension $d = \dim V$.

It is clear from (6) that it takes $O(c \cdot d^s)$ additions and multiplications in \mathcal{R} to compute a single entry in a tensor resulting from contracting c tensors along s edges at once. Let l_1, \dots, l_c be the number of legs of the c tensors, the resulting tensor thus has $-2s + \sum_{i=1}^c l_i$ legs, hence consists of $d^{-2s + \sum_{i=1}^c l_i}$ entries. Therefore the total number of additions and multiplications required to build the resulting tensor is

$$O(c \cdot d^s) \cdot d^{-2s + \sum_{i=1}^c l_i} = O(c \cdot d^{-s + \sum_{i=1}^c l_i}).$$

This motivates the following definition.

Definition 3.1. Given a contraction sequence \mathcal{S} on a tensor network Γ , for a step in \mathcal{S} that contracts c tensors with l_1, \dots, l_c legs respectively along s edges at once, its *local contraction-width* is $-s + \sum_{i=1}^c l_i$. The *contraction-width* $w(\mathcal{S})$ of \mathcal{S} is the maximum local contraction-width of all its steps, and the contraction-width $w(\Gamma)$ of Γ is the minimal contraction-width among all possible contraction sequences on Γ .

Note that the contraction-width is independent of the dimension d , hence it makes sense to consider (abstract) contraction sequences on a graph Γ and define its contraction-width $w(\Gamma)$ similarly. Note also that adding degree 2 vertices does not affect the contraction-width of a nontrivial graph. These justify the use of $w(D)$ in Section 1.

It is clear that adding more tensors to a step of contraction never decreases its local contraction-width, hence at best has no effect on the exponential factor of the number of additions and multiplications required. Therefore we further restrict ourselves to contraction sequences where each step contracts exactly 2 tensors. In particular, for a tensor network with $2n$ vertices, all contraction sequences consist of $2n - 1$ steps. For a contraction sequence \mathcal{S} with width w , it then takes $O(n \cdot d^w)$ additions and multiplications in \mathcal{R} to perform all steps in \mathcal{S} .

In [MS08], Markov and Shi restricted themselves to contraction sequences where each step contracts only one edge. For such a contraction sequence \mathcal{S} , they defined its *contraction complexity* $cc(\mathcal{S})$ as the maximum degree of a merged vertex during the contraction process, and hence the contraction complexity $cc(\Gamma)$ of a tensor network Γ is the minimum complexity among all possible \mathcal{S} where each step contracts along only one edge. It is immediate from definition that $w(\mathcal{S}) = cc(\mathcal{S}) + 1$, hence $w(\Gamma) \leq cc(\Gamma) + 1$, where the inequality may be strict since the contraction-width is defined over a larger family of contraction sequences. This translates [MS08, Lemma 4.4] into Lemma 1.4 as we stated.

3.2. Proof of Theorem 1.2 and Theorem 1.3. Given an oriented knot diagram D with n crossings and a contraction sequence on D with contraction-width w , we cut an arbitrary arc in D open, turning it into an oriented long knot diagram D' . As discussed in Section 2.2, the arc cut open can be eliminated immediately after we apply the Reshetikhin–Turaev functor to D' , obtaining a tensor network with at most $2n$ vertices along with a contraction sequence whose width is at most w . This procedure is $O(n)$ and is dominated by subsequent computation.

We have established in the previous section that it takes $O(n \cdot d^w)$ additions and multiplications in \mathcal{R} to compute the desired knot polynomial, hence it remains only to bound the time complexity of performing additions and multiplications in \mathcal{R} .

For a k -variable knot polynomial of Reshetikhin–Turaev type, we have $\mathcal{R} = \mathbb{Z}[t_1^{\pm 1}, \dots, t_k^{\pm 1}]$. Let M be the maximum degree of all variables in the entries of $(R_{e_i \otimes e_j}^{e_k \otimes e_l})$ and $(h_{e_i}^{e_j})$. The maximum degree for each variable that can appear throughout the process is thus $2nM$. Hence all polynomials have at most $(2nM)^k = O(n^k)$ terms. Therefore the naive number of additions and multiplications in \mathbb{Z} required for each polynomial addition or multiplication is $O(n^{2k})$. Alternatively, we use Kronecker substitution to turn multivariable polynomials into univariate ones and apply Fast Fourier transform to compute their multiplications, which reduces the required number of additions and multiplications to $O(k(4nM + 1)^k \log(4nM + 1)) = O(n^k \log n)$ [vzGG99, Section 8.4].

We now bound the integers that may appear during the process. Let C be the maximum absolute value of coefficients of the entries of $(R_{e_i \otimes e_j}^{e_k \otimes e_l})$ and $(h_{e_i}^{e_j})$. Since there are at most M^k distinct monomials appearing in the entries, the maximum absolute value of integers that can appear throughout the process is bounded by the maximum absolute value of the

coefficients of

$$d^{2n} \left(\sum_{i=1}^{M^k} C m_i \right)^{2n} = \sum_{\substack{p_1 + \dots + p_{M^k} = 2n \\ p_1, \dots, p_{M^k} \geq 0}} \frac{d^{2n} \cdot (2n)! \cdot C^{2n}}{(p_1!) \cdots (p_{M^k}!)} \cdot m_1^{p_1} \cdots m_{M^k}^{p_{M^k}}. \quad (8)$$

where each m_i represents a monic monomial. Hence it is bounded by

$$\sum_{\substack{p_1 + \dots + p_{M^k} = 2n \\ p_1, \dots, p_{M^k} \geq 0}} \frac{d^{2n} \cdot (2n)! \cdot C^{2n}}{(p_1!) \cdots (p_{M^k}!)} \leq \binom{2n + M^k}{M^k} \cdot (2ndC)^{2n} \leq (2ndCM^k)^{2n+M^k}, \quad (9)$$

so the bit-size of the integers is $O(n \log n)$. Therefore Theorem 1.2 follows from the fact that adding and multiplying integers with bit-size in $O(n \log n)$ are

$$O((n \log n) \cdot \log(n \log n)) = O(n(\log n)^2). \quad (10)$$

Replacing all terms with precise expressions above gives Theorem 1.3.

Acknowledgements. The author wishes to thank Nathan Dunfield, Stavros Garoufalidis, Saul Schleimer, and Jonathan Spreer for useful conversations. The author also specially thanks Mingde Ren, from whom the author learned about tensor networks. This material is based upon work supported by the National Science Foundation under Grant No. DMS-2424139 while the author was in residence at the Simons Laufer Mathematical Sciences Institute in Berkeley, California, during the Spring 2026 semester. The author was also supported by the National Science Foundation under Grant No. DMS-2303572 during the same semester.

REFERENCES

- [Ale28] J. W. Alexander, *Topological invariants of knots and links*, Trans. Amer. Math. Soc. **30** (1928), no. 2, 275–306.
- [BC17] Jacob Bridgeman and Christopher Chubb, *Hand-waving and interpretive dance: an introductory course on tensor networks*, J. Phys. A **50** (2017), no. 22, 223001, 61.
- [Bie90] Dan Bienstock, *On embedding graphs in trees*, J. Combin. Theory Ser. B **49** (1990), no. 1, 103–136.
- [BN05] Dror Bar-Natan, *Khovanov’s homology for tangles and cobordisms*, Geom. Topol. **9** (2005), 1443–1499.
- [BNvdV19] Dror Bar-Natan and Roland van der Veen, *A polynomial time knot polynomial*, Proc. Amer. Math. Soc. **147** (2019), no. 1, 377–397.
- [BNvdV24] ———, *A perturbed-Alexander invariant*, Quantum Topol. **15** (2024), no. 3, 449–472.
- [BNvdV25] Dror Bar-Natan and Roland van der Veen, *A fast, strong, topologically meaningful and fun knot invariant*, 2025, 2509.18456.
- [Bur20] Benjamin Burton, *The next 350 million knots*, 36th International Symposium on Computational Geometry, LIPIcs. Leibniz Int. Proc. Inform., vol. 164, Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2020, pp. Art. No. 25, 17.
- [CDGW] Marc Culler, Nathan Dunfield, Matthias Goerner, and Jeffrey Weeks, *SnapPy, a computer program for studying the geometry and topology of 3-manifolds*, Available at <http://snappy.computop.org>.
- [GK26] Stavros Garoufalidis and Rinat Kashaev, *Multivariable Knot Polynomials from Braided Hopf Algebras with Automorphisms*, Publ. Res. Inst. Math. Sci. **62** (2026), no. 1, 75–114.

- [GL24] Stavros Garoufalidis and Shana Li, *Values of V_n -polynomials of knots*, <https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/XE4TOF>, 2024.
- [GL25] Stavros Garoufalidis and Shana Yunsheng Li, *Patterns of the v_2 -polynomial of knots*, 2025, 2409.03557.
- [GLY] Stavros Garoufalidis, Shana Yunsheng Li, and Josephine Yu, *Positivity and concavity of the colored Links–Gould polynomials of knots*, In preparation.
- [HTW98] Jim Hoste, Morwen Thistlethwaite, and Jeff Weeks, *The first 1,701,936 knots*, Math. Intelligencer **20** (1998), no. 4, 33–48.
- [Jon87] Vaughan Jones, *Hecke algebra representations of braid groups and link polynomials*, Ann. of Math. (2) **126** (1987), no. 2, 335–388.
- [JVV90] F. Jaeger, D. L. Vertigan, and D. J. A. Welsh, *On the computational complexity of the Jones and Tutte polynomials*, Math. Proc. Cambridge Philos. Soc. **108** (1990), no. 1, 35–53.
- [Kor22] Tuukka Korhonen, *A single-exponential time 2-approximation algorithm for treewidth*, 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science—FOCS 2021, IEEE Computer Soc., Los Alamitos, CA, [2022] ©2022, pp. 184–192.
- [KS93] Ephraim Korach and Nir Solel, *Tree-width, path-width, and cutwidth*, Discrete Appl. Math. **43** (1993), no. 1, 97–101.
- [LT79] Richard J. Lipton and Robert Endre Tarjan, *A separator theorem for planar graphs*, SIAM J. Appl. Math. **36** (1979), no. 2, 177–189.
- [Mak05] J.A. Makowsky, *Coloured tutte polynomials and kauffman brackets for graphs of bounded tree width*, Discrete Applied Mathematics **145** (2005), no. 2, 276–290, Structural Decompositions, Width Parameters, and Graph Labelings.
- [Mar21] Clément Maria, *Parameterized complexity of quantum knot invariants*, 37th International Symposium on Computational Geometry, LIPIcs. Leibniz Int. Proc. Inform., vol. 189, Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2021, pp. Art. No. 53, 17.
- [MK19] Konstantinos Meichanetzidis and Stefanos Kourtis, *Evaluating the Jones polynomial with tensor networks*, Phys. Rev. E **100** (2019), no. 3, 033303, 7.
- [MS08] Igor L. Markov and Yaoyun Shi, *Simulating quantum computation by contracting tensor networks*, SIAM J. Comput. **38** (2008), no. 3, 963–981.
- [Oht02] Tomotada Ohtsuki, *Quantum invariants*, Series on Knots and Everything, vol. 29, World Scientific Publishing Co., Inc., River Edge, NJ, 2002, A study of knots, 3-manifolds, and their sets.
- [Orfrm[o]–9] Román Orús, *Tensor networks for complex quantum systems*, Nature Reviews Physics **1** (2019), no. 9, 538–550.
- [RT90] Nikolai Reshetikhin and Vladimir Turaev, *Ribbon graphs and their invariants derived from quantum groups*, Comm. Math. Phys. **127** (1990), no. 1, 1–26.
- [Shu11] Alexander N. Shumakovitch, *Patterns in odd Khovanov homology*, J. Knot Theory Ramifications **20** (2011), no. 1, 203–222.
- [Thi25] Morwen B. Thistlethwaite, *The enumeration and classification of prime 20-crossing knots*, Algebr. Geom. Topol. **25** (2025), no. 1, 329–344.
- [Tur88] V. G. Turaev, *The Yang-Baxter equation and invariants of links*, Invent. Math. **92** (1988), no. 3, 527–553.
- [vzGG99] Joachim von zur Gathen and Jürgen Gerhard, *Modern computer algebra*, Cambridge University Press, New York, 1999.

(Shana Yunsheng Li) DEPARTMENT OF MATHEMATICS, UNIVERSITY OF ILLINOIS, URBANA, IL, USA
<https://shana-y-li.github.io>
 Email address: yl202@illinois.edu