Kalpa Publications in Computing

Volume 18, 2024, Pages 69–80

LPAR 2024 Complementary Volume

Automated Reasoning with Tangles: towards Quantum Verification Applications

Andrew Fish[1,2] and Alexei Lisitsa[1]

[1] Department of Computer Science, University of Liverpool, Ashton St, Liverpool, UK
{andrew.fish,lisitsa}@liverpool.ac.uk
[2] Distributed Analytics Solutions Ltd

Abstract

We demonstrate utility of generic automated reasoning (AR) methods in the computational topology domain, evidencing the benefit of the use of existing AR machinery within the domain on the one hand, whilst providing a pathway into a rich playground with potential to drive future AR requirements. We also progress towards quantum software verification contribution, via a recent proposal to use tangles as a representation of a certain class of quantum programs. The general methodology is, roughly speaking, to transform tasks of equivalence of topological objects (tangles) into equivalence of algebraic objects (pointed quandles) and those in turn translate into AR tasks. To enhance trust in automated checks, this can be followed by interpretation of AR outputs as human-readable output, either in symbolic algebraic form suitable for domain experts or ideally in visual topological form, potentially suitable for all. We provide formalisation via an appropriate class of labelled tangles (suitable for Quantum Verification) with associated algebraic invariants (pointed involutory quandles) and translate equivalence checking of these invariants to equational reasoning tasks. Furthermore, subsequent to automated proof creation for simple quantum verification (QV) examples, we demonstrate manual extraction of visual proof rules and visual equivalence, utilising proof graphs as a bridging step, progressing towards the automation of human-readable visual proofs.

1 Introduction

Generic automated reasoning (AR), such as automated theorem proving and disproving, or SAT-solving, provide a powerful alternative to human reasoning and for specialised algorithms in mathematics. While applications in symbolic logic and algebra have been known for a long time, the applications in topology have occurred more recently. For example, in [3, 5, 6] automated reasoning was applied to two of the probably most known problems in computational topology, *unknot detection* [3, 5] and *knot equivalence* [6]. These applications rely on the idea that relevant topological properties can be faithfully characterised by the properties of algebraic structures/invariants associated with knots, and establishing the properties of algebraic structures can be delegated to automated reasoning. While applications of AR to unknot detection and knot equivalence have turned out to be practically efficient in many cases, they posed some conceptual and algorithmic challenges and problems for further exploration. For example, can one extract untangling sequence for a knot from the first-order equational proof of its triviality (unknotedness), or prove knot equivalence/ non-equivalence by SAT-solving?

In this paper we apply automated reasoning to algorithmic problems for tangles, close relatives of knots and links. Informally speaking, tangles may be thought of entangled pieces of rope in space with some fixed endpoints. We adopt particular algebraic invariants, that is involutory pointed quandles, to partially address the tangle equivalence problem, expanding on our previous work on knots and links. A particularly appealing motivation for tangle equivalence problem comes from the domain of quantum computations.

Quantum computing promises great computational power, and the development of quantum hardware and software has accelerated enormously in recent years. Software testing is significantly more difficult in a quantum setting than for classical programs since any attempt to check properties of a quantum state during run time leads to changes in the quantum state. This makes the verification of quantum programs (i.e. assurances of their correctness), which is vital to ensure reliability, extremely challenging. Recently, [13] proposed a topological approach in which quantum circuits/programs are modelled by tangles and verification of program equivalence is essentially reduced to visually appealing tangle equivalence/isotopy. The approach currently lacks automation and implementation. The tangles required for quantum verification are close relatives to knots, and expanding on previous work [3, 4, 5], we propose to bridge the automation gap, adapting automated reasoning with associated quandles for tangles to be used to establish or refute required equivalences (isotopies) of tangles.

Paper Contributions: We propose a methodology for the use of generic AR methods in the computational topology domain, via translation of associated algebraic objects to AR tasks, enabling automatic production of proofs of equivalence, and non-equivalence. We demonstrate the utility via a case study of simple worked examples drawn from the Quantum Verification domain, automatically extracting proofs of equivalence of associated algebraic objects, with indications of progress towards the possibility of extracting tangle isotopy moves from ATP output. This is a significant step towards the goals of automated human-readable (visual) proof construction, with potential for application in automating quantum program verification, and for generalisation to knot/link isotopy (with applications in many fields).

2 Tangles, Quandles, Pointed Quandles and Automated Reasoning

We provide a brief introduction to a range of existing concepts and indicate specialisations we adopt in the following. Similar to the case of knots, for tangles one can work at topological level and consider them as generic embeddings of unions of a finite number of unit intervals and unit circles, in an orientable surface, and then ambient isotopy is the notion of equivalence. Informally, one may think of having a cylinder with n pieces of string having their ends (not meeting and being fixed) on the top and bottom boundaries of the cylinder (this is a bit simplified to make it easier to envisage), and k pieces of string whose ends are glued together instead (but free to move within the cylinder - each one of these is, in fact, just a knot), and then equivalence (isotopy) can be thought of as allowing the strands to move freely away from the boundary (with fixed endpoints) without allowing strand to pass through each other. However, one may adopt a combinatorial perspective, and work with tangle diagrams, which

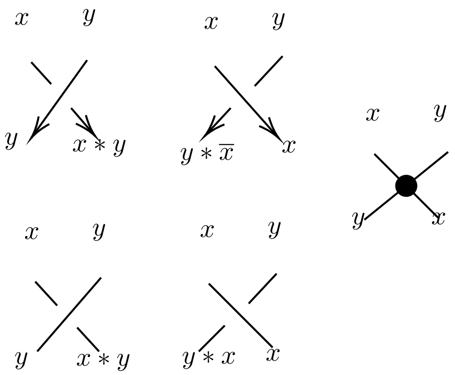

Figure 1: The crossing types: top are positive and negative classical crossings, middle is a virtual crossing, and the bottom are unoriented classical crossings. The effect on quandle colouring of each crossing type is also shown, considering labelling by x, y at the top of the diagrams: top row shows the two operations for quandles (one is essentially an inverse operation), whilst the bottom row shows the effect for involutory quandles, where the inverse operation is the same as the usual operation - in this case local orientation is unimportant.

are combinatorial objects, where equivalence is defined by have tangle diagrams differing by a sequence of moves of certain types on diagrams. They may be viewed as projections of the tangles. The moves (called Reidemeister moves for the classical case, and extended to virtual Reidemeister moves for the generalisation to the virtual case) encapsulate isotopy.

Definition 1. *An* oriented tangle diagram T *is a generic immersion of a union of a finite number of unit intervals and unit circles, in a disc in* $\mathbb{R}^2$*, such that the only points on the boundary are the distinct endpoints of the unit intervals, and there are finitely many transverse double points in the interior. For* classical *oriented tangle diagrams, we have two types of internal crossing, positive or negative according to the relative orientations of the arcs meeting at the crossing, whilst for* virtual *oriented tangle diagrams, we allow another type of crossing, called a virtual crossing. The arcs of the tangle diagram,* $arc(T)$*, are the unbroken segments of the diagram, which are partitioned into external arcs* $earc(T)$*, having at least one end point on the boundary, and internal arcs* $iarc(T)$*. The external arcs are further partitioned*[1] *into* start arcs *(or tails),* $starc(T)$*, and* end arcs *(or heads),* $endarc(T)$*. At each internal classical crossing, the "unbroken" arc is the* overcrossing *arc and the "broken" arcs form the* undercrossing*. A tangle diagram is* ordered *if we have a fixed ordering of the endpoints.*

Figure 1 shows the different crossing types, whilst Figure 2 shows the classical Reidemeister moves; the virtual extensions can be found in [9], but we will focus on the classical setting here to avoid further complexity of exposition. In general, we will simplify language and use the term tangle synonynously with tangle diagram. Aiming to keep the set-up as simple as possible, whilst still being suitable to demonstrate our intent, we place some contextual restrictions (even if generalisations will be interesting and valuable in their own right). So, we consider unoriented but ordered tangles throughout.

Definition 2. *An involutory quandle is an algebraic structure, that is a set* Q *with an operation* $*$ *satisfying the following properties*

[1] For the purposes of this paper, we also require unoriented tangles to also have these partitions, so that one can identify the start and end of each strand.

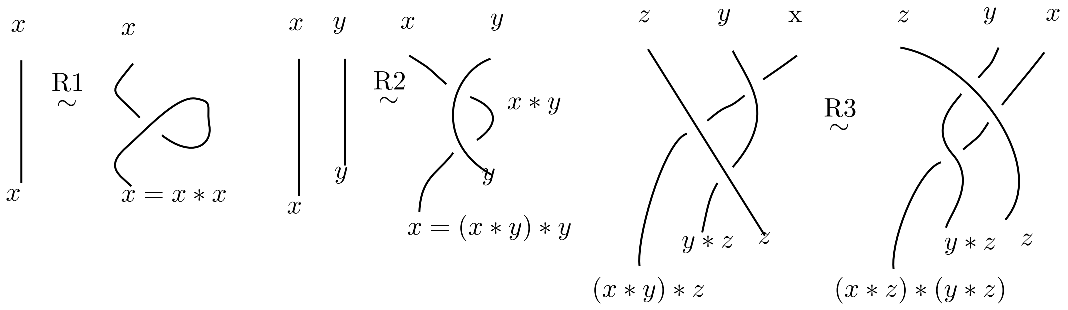

Figure 2: The Reidemeister moves for classical diagrams. These are shown unoriented - for oriented versions, all choices of orientation are permitted. The intuition is that one can move strands as one would naturally perceive, so the $R3$ move can be though of as moving one strand (z) over another crossing (or indeed moving one strand (x) under another crossing). The labelling by quandle elements also demonstrates the known fact that the axioms of the involutory quandle (see Definition 2) are precisely those preserved by these moves. Observe that the ordering of the top labels of the $R3$ move here are z, y, x - it is a simple exercise to draw the corresponding move with ordered top labels x, y, z. Extended moves for virtual diagrams can be found in [9].

1. $x * x = x$
2. $(x * y) * y = x$
3. $(x * y) * z = (x * z) * (y * z)$.

Definition 3. *The involutory quandle crossing relations are depicted at the bottom of Figure 1. They are relations that must hold in the quandle (and can be viewed as a means of computing the third label give the existence of any two labelled arcs of the crossing).*

Definition 4. *A pointed involutory quandle $\langle Q, *, a_1, \dots a_n\rangle$ is an involutory quandle $\langle Q, *\rangle$ equipped with a sequence of distinguished elements $a_1, \dots, a_n \in Q$. In logical terms, this is a first-order structure (a model) for the vocabulary $\{*, a_1, \dots, a_n\}$, where an interpretation $[*]$ of $*$ is a binary operation satisfying the involutory quandle axioms, and $a_1, \dots a_n$ are distinct constants interpreted by not necessarily distinct elements $[a_1], \dots, [a_n]$ of Q. A pointed involutory quandle $\langle Q, *, a_1, \dots a_n\rangle$ is called* g-pointed *if interpretations of constants $a_1, \dots, a_n$ form a generating set for Q.*

Definition 5. *Two pointed involutory quandles $\langle Q_1, *_1, a_1, \dots a_n\rangle$ and $\langle Q_2, *_2, b_1, \dots b_m\rangle$ are strongly isomorphic if $n = m$, $a_i \equiv b_i$ for $i = 1, \dots n$, where $\equiv$ denotes syntactic equality[2], and there is an involutory quandle isomorphism $i : \langle Q_1, *_1\rangle \to \langle Q_2, *_2\rangle$ such that $i([a_i]) = [b_i]$.*

For a tangle T and a set S, a mapping $c : arc(T) \to S$ is called a *coloring* of T by elements of S. With any tangle T and any c of T we can associate an involutory quandle presentation $IQ(T, c) = \langle G, R\rangle$ where $G = Im(c)$ is the set of generators determined by the image under c of the set of arcs of T, and R is a set of defining relations, defined as follows. For each crossing t of T, the set R contains a defining relation $a_i * a_j = a_k$, where a_i is the colour of an incoming

[2] Two expressions are syntactically equal if they are literally made of the same characters in the same order.

under-crossing arc of t, a_j is a colour of over-crossing arc of t, and a_k is a colour of outgoing under-crossing arc of t[3]. Recall that $arcs(T)$ is split into two disjoint subsets, $earcs(T)$ of end arcs and $iarcs(T)$ of internal arcs.

For an involutory quandle presentation $IQ = \langle G, R \rangle$ and a defining relation $\tau = a \in R$, we define an operation of *reduction* of IQ by $\tau = a$ as: 1) removal of a from G; 2) removal of $\tau = a$ from R; 3) replacement of all occurrences of a in all (other) defining relations in R by τ. For $IQ = \langle G, R \rangle$ and $IQ' = \langle G', R' \rangle$ we define the corresponding reduction relation $IQ \Rightarrow IQ'$ as IQ' is obtained from IQ by the reduction by some $\tau = a \in R$. It is easy consequence of the definitions that if $IQ \Rightarrow^* IQ'$ then IQ and IQ' define (present) isomorphic involutory quandles.

Definition 6. *Let T be a tangle, S a set of colours and $c : arc(T) \to S$ an injective coloring. Denote by $IQ^r(T, c)$ a* fully reduced *involutory quandle presentation $\langle G, R \rangle$, that is, a presentation such that: 1) the initial colouring $IQ(T, c) \Rightarrow^* IQ^r(T, c)$; 2) G is the set of colours of exactly the external arcs, $G = Im(c|_{earc(T)})$; 3) R contains only defining relations for generators from G.*

Thus, the generators in a fully reduced involutory quandle presentation for a tangle T are distinct colours of external arcs of T; the colours of all internal arcs are uniquely determined by involutory quandle operation repeatedly applied to the colours of external arcs.

We make two observations:

- It is not necessarily the case that for every T there exists a fully reduced presentation $IQ^r(T, c)$ (so this is a restriction on the class of T considered, but this is reasonable in the quantum setting due to the compositionality in the tangle encoding).
- For a tangle T and a colouring c, $IQ(T, c)$ and $IQ^r(T, c)$ present isomorphic involutory quandles.

Definition 7. *A tangle T is called* end-colourable *if $IQ^r(T, c)$ exists for some c, and* end-coloured *if each end arc has been assigned a colour (which are sufficient to deduce the colours of the rest of the arcs of T).*

From now one we will consider only end-colourable, ordered tangles (with an identification of start and end extenral arcs). We denote the set of such tangles by $\mathcal{T}$. This class includes, amongst others, most of the tangles occurring in the verification of quantum program in [13].

Proposition 1. *Two $g-$pointed involutory quandles $\langle Q_1, *_1, a_1, \dots a_n \rangle$ and $\langle Q_2, *_2, a_1, \dots a_n \rangle$ given by presentations $\langle G, R_1 \rangle$ and $\langle G, R_2 \rangle$, with $G = \{a_1, \dots, a_n\}$, are strongly isomorphic if and only if*

- *$AX_{IQ} \cup R_1 \vdash R_2$ and $AX_{IQ} \cup R_2 \vdash R_1$.*

Proof. This is an immediate consequence of the definitions, since the set of equations provable from $AX_{IQ} \cup R_1$ and $AX_{IQ} \cup R_2$ are the same. □

Proposition 1 provides an opportunity to use automated reasoning that is first-order theorem proving and disproving to establish strong isomorphism and non-isomorphism of $g-$pointed involutory quandles. Now, $g-$pointed involutory quandles are invariants of end-coloured tangles, as stated in the following proposition.

[3] Notice that for involutory quandles this definition is actually invariant with respect to swapping incoming and outgoing undercrossing arcs.

Proposition 2. *If two ordered and end-coloured tangles $\langle T, e_1, \dots e_{2n}\rangle$ and $\langle T', e'_1, \dots e'_{2n}\rangle$ are isotopic then the g−pointed involutory quandles presented by $IQ^r(T, c)$ and $IQ^r(T', c')$ are strongly isomorphic. Here $e_1, \dots e_{2n}$ and $e'_1, \dots, e'_{2n}$ are end arcs of T and T' respectively, and $c(e_i) = c'(e'_i) = a_i$ for $i = 1, \dots, 2n$.*

Proof. By induction on the length of a sequence of Reidemeister moves demonstrating isotopy. It is straightforward to check that any move (see Fig 2) applied to a tangle T, results in a tangle T' such that $IQ(T, c)$ and $IQ(T', c')$ are equivalent (the set of equalities are mutually derivable). Hence $IQ^r(T, c)$ and $IQ^r(T', c')$ are equivalent. □

Thus, Proposition 2 associates the checking of isotopy of end-coloured tangles to the checking of strong isomorphism of associated pointed involutory quandles, which taken together with Proposition 1, further reduces it to automated reasoning tasks.

3 General Methodology

Figure 3 shows a general overview of the methodology. Given a pair of ordered, end-coloured tangles T_1, T_2 (with no closed loops), one can compute their pointed involutory quandle presentations, $PIQ(T_1), PIQ(T_2)$. We construct TP tasks which are the logical equivalent of checking presentational inference. We do this in both directions to check for equivalence rather than one directional deduction (i.e. $LPIQ(T_1) \models LPIQ(T_2)$ and $LPIQ(T_2) \models LPIQ(T_1)$). Then a theorem prover (we use ProverX and Prover9) creates proofs, when one exists, certifying equivalence of pointed involutory quandles. From this output, and the associated proof graphs, one may try to extract isotopy moves (currently manually), together with context and some application ordering, where possible. These may help to guide or directly construct an isotopy sequence between the original tangles, where possible.

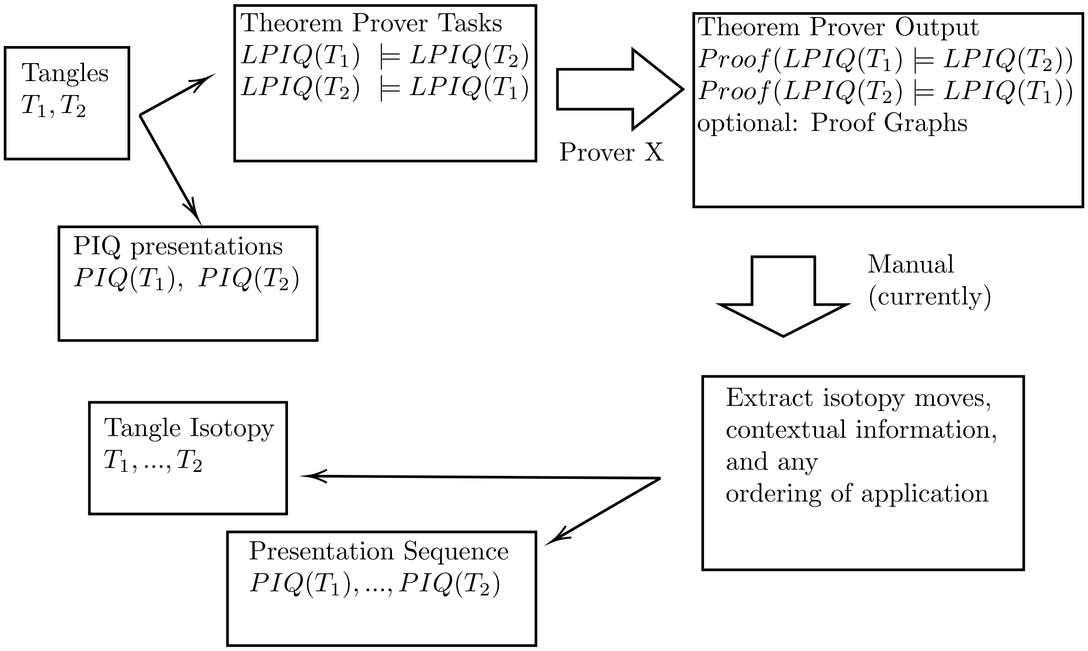

Figure 3: General Methodology Overview.

3.1 Detailed worked example

We examine a simple example, shown in Figure 4, to demonstrate the overall approach and to facilitate discussion about the effects of alternative task encoding including variations in output. We also explore the potential for identification of isotopy moves on tangles from the outputs (of Prover 9 and Prover X). Whilst the ultimate goal may be fully automated tangle isotopy sequence construction, whenever possible, the possibility of heuristically assisting a manual proof construction is valuable.

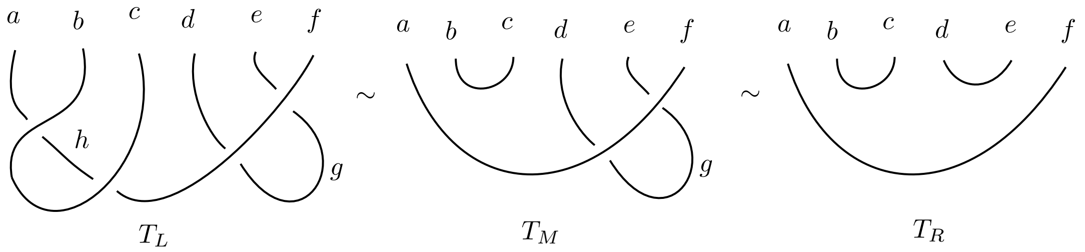

Figure 4: The left and right tangles, T_L and T_R, specify the task, arising from Figure 11 of [13]. An isotopy between T_L and T_R can be seen as 2 $R2$-moves, with T_M being T_L after apply one $R2$-move to strand bc, and the subsequent $R2$-move of strand de yields T_R.

The following is the proof obtained from Prover 9 using the **Full-Label Encoding** in which all tangles arcs are labelled by unique labels in both left and right hand tangles, T_L and T_R, with consistent labelling on all tangle ends. Notice that this can be seen as an application of Proposition 1 for the case of non-reduced presentation and only one proof task. Within the logical viewpoint, this treats each label as a constant. We observe that this enables a check of the deduction of equalities of T_R from T_L, for the example in Figure 4, but not vice versa, since there are labels (h, g, representing constants) in T_L that are not in T_R. However, if one can find an isotopy sequence from T_L to T_R, then it is reversible, so this would be sufficient to deduce equivalence. Figure 5 shows the proof graph produced by Prover X, together with a possible interpretation in terms of tangle isotopy.

```
%Assumptions arising from $T_L$.
a*b=h.
h*c=f.
b=c.
d*f=g.
g*f=e.
```

```
% Involutory quandle axioms
x * x = x.
(x * y) * y = x.
(x * y) * z = (x * z) * (y * z).

%Goals arising from $T_R$.
a=f & b=c & d=e.
```

```
% Proof 1 at 0.01 (+ 0.00) seconds.
.....
1 a = f & b = c & d = e # label(non_clause) # label(goal).  [goal].
3 (x * y) * y = x.  [assumption].
6 a * b = h.  [assumption].
7 h * c = f.  [assumption].
8 b = c.  [assumption].
9 c = b.  [copy(8),flip(a)].
10 d * f = g.  [assumption].
11 g * f = e.  [assumption].
12 f != a | c != b | e != d.  [deny(1)].
13 a != f | e != d.  [copy(12),rewrite([9(4)]),flip(a),xx(b)].
14 h * b = f.  [back_rewrite(7),rewrite([9(2)])].
```

```
20 a = f.  [para(6(a,1),3(a,1,1)),rewrite([14(3)]),flip(a)].
24 e != d.  [back_rewrite(13),rewrite([20(1)]),xx(a)].
26 $F.  [para(10(a,1),3(a,1,1)),rewrite([11(3)]),unit_del(a,24)].
```

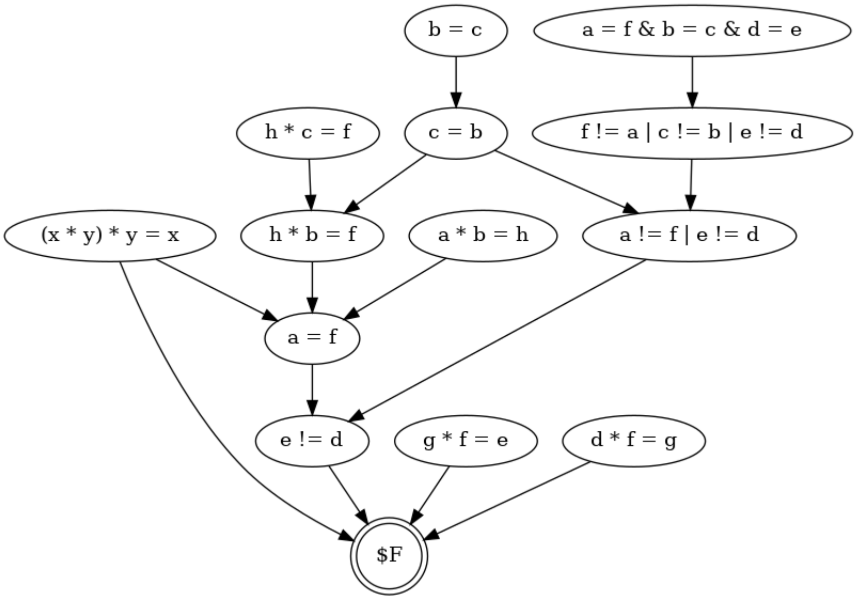

Figure 5: The proof graph obtained with ProverX (the proof is exactly the same as that of Prover9 in this case). The assumption in line 3, $(x * y) * y = x$, corresponds to an $R2$-move, and its application in the context of the crossings (a, b, h) and (h, c, f), from line 6 $a * b = h$ and line 7 $h * c = f$, enable the deduction of line 20 $a = f$. One may interpret this stage as the application of an $R2$-move to T_L yielding T_M. We see that line 3, $(x * y) * y = x$, is also applied again, in the context of the crossings (d, f, g) and (g, f, e), from line 10 $d * f = g$ and line 11 $g * f = e$. One may interpret this stage as the application of an $R2$-move to T_M yielding T_R.

Next, we consider an **End-Label Encoding**, in which labels are assigned only to the external arcs of the tangles. All other arc labels need to be derived in this case. Recall that the class of tangles obtainable in this manner (i.e. such that all arc labels can be derived from the end labels) are called end-colourable tangles. This case corresponds to the application of Proposition 1 to fully reduced presentations. In general, the tangles built for the expression of quantum programs are end-colourable due to their compositional manner of construction. There is an exception for the case in which closed loops are added - this requires an extension with a new label assigned each additional closed loop, and is not explicitly dealt with here.

We use Prover X, and consider the two separate deductions, denoted ⇒ and ⇐. In relation to Figure 4, we forget the labels h, g in T_L, and instead derive labels from the crossing equation for the crossings. For example, instead of having a label h, for a constant, we have a label $a * b$ derived from the top left crossing. Figure 7 shows the proof graphs produced by Prover X, together with a possible interpretation in terms of tangle isotopy.

ProverX, ⇒

```
1 a = f & b = c & d = e # label(non_clause) # label(goal).  [goal].
3 (x * y) * y = x.  [assumption].
```

```
6 (a * b) * c = f.  [assumption].
7 f = (a * b) * c.  [copy(6),flip(a)].
8 b = c.  [assumption].
9 c = b.  [copy(8),flip(a)].
10 (d * f) * f = e.  [assumption].
11 e = d.  [copy(10),rewrite([7(2),9(5),3(6),7(4),9(7),3(8),3(5)]),flip(a)].
12 f != a | c != b | e != d.  [deny(1)].
13 $F.  [copy(12),rewrite([7(1),9(4),3(5),9(4),11(7)]),xx(a),xx(b),xx(c)].
```

ProverX, ⇐

```
1 (a * b) * c = f & b = c & (d * f) * f = e # label(non_clause) # label(goal).  [goal].
3 (x * y) * y = x.  [assumption].
6 a = f.  [assumption].
7 f = a.  [copy(6),flip(a)].
8 b = c.  [assumption].
9 c = b.  [copy(8),flip(a)].
10 d = e.  [assumption].
11 e = d.  [copy(10),flip(a)].
12 (a * b) * c != f | c != b | (d * f) * f != e.  [deny(1)].
13 $F.  [copy(12),rewrite([9(4),3(5),7(2),9(4),7(8),7(10),3(11),11(8)]),xx(a),xx(b),xx(c)].
```

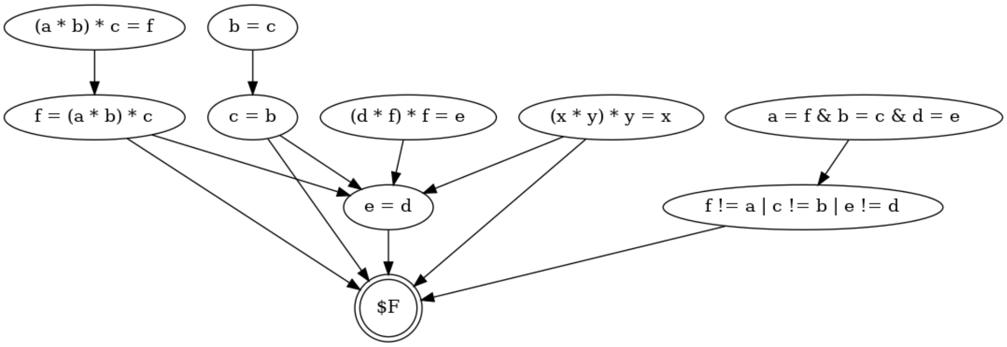

Figure 6: The proof graph obtained with Prover X ⇒. An interpretation may be first performing isotopy of $R2$ move on strand de (since the deduction $e = d$ is closer to the roots of the graph) then an $R2$ move on the strand bc.

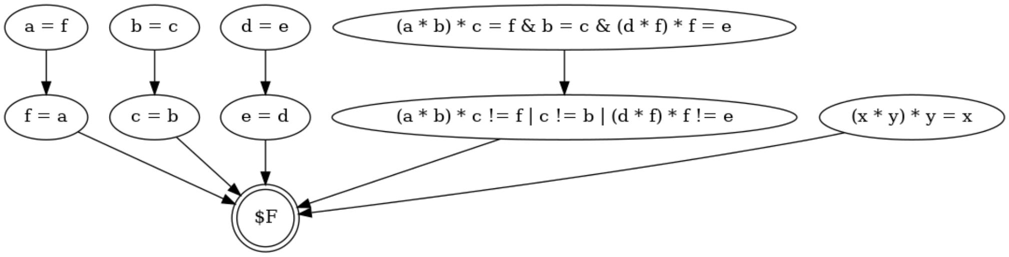
a = f
b = c
d = e
(a * b) * c = f & b = c & (d * f) * f = e
f = a
c = b
e = d
(a * b) * c != f | c != b | (d * f) * f != e
(x * y) * y = x
$F

Figure 7: The proof graph obtained with Prover X ⇐. An interpretation may be performing both isotopies of $R2$ moves on strands de and bc from T_R to T_L - one application for the far left and right disjuncts in the "large nodes" of the proof graph.

3.2 Disproving Examples

We show non-equivalence of the tangles in Figure 8 using finite countermodel finding. More precisely we show that in the theory of involutory quandles.

$$a * b = c \wedge (d * c) * c = e \wedge (f * e) * e = h \wedge b = m \wedge g * h = n$$

$$\not\vdash \;(a * b) * d = c \wedge d * (a * b) = e \wedge (f * e) * e = h \wedge b = m \wedge g * h = n$$

Mace 4 finds the following countermodel instantanously (slight layout alterations made here to output to save space):

```
interpretation( 3, [number = 1,seconds = 0], [
    function(*(_,_), [
        0,0,0,
        2,1,1,
        1,2,2]),
    function(a, [0]), function(b, [0]), function(c, [0]), function(d, [1]), function(e, [1]),
    function(f, [0]), function(g, [0]), function(h, [0]), function(m, [0]), function(n, [0])]).
```

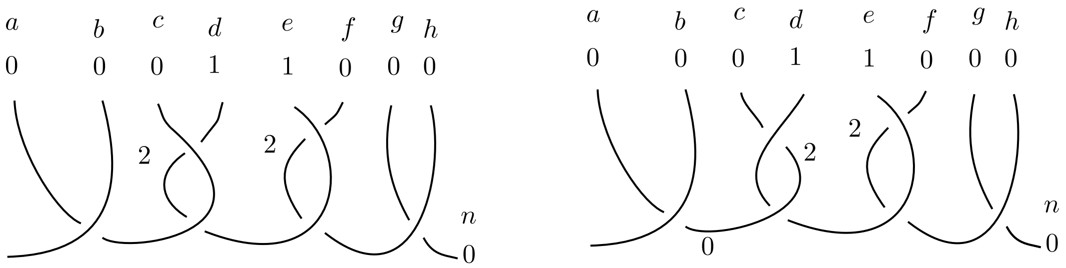

Figure 8: The left hand tangle is the left hand side of Figure 11 of [13], whilst the right hand tangle is a slight modification of the left hand tangle, just changing one crossing so that the tangles are not isotopic. The numbering assigned to the arcs is a visual display of the countermodel found that shows that these two tangles are not isotopic. The left hand tangle displays a colouring by a quandle of size 3 with elements $\{0, 1, 2\}$ such that the colouring equation $2b = a + c \bmod 3$ holds. The right hand tangle shows that the attempt to colour it, with the same end colours, leads to an inconsistency (we see an arcs labelled by both 0 and 2 here, and $0 \neq 2$).

4 Related Work

The history of involutory quandles goes back to 1943 when they were introduced in [15] under the name *keis* in the context of finite geometry. Later, involutory quandles and closely related *quandles*, *racks*, *distributive groupoids* have turned out to be very useful algebraic structures in studying knots, links and other knotted objects. In [8, 10], it was shown that fundamental involutory quandle is an classifying invaraint of knots up to orientation reversing homeomorphisms of knots. In particular, a trivial fundamental involutory quandle characterizes the trivial knot (unknot) [16]. Recent surveys of quandle ideas can be found in [1, 2]. Knotoids/Linkoids are another generalization of knots, closely related to tangles, and theory relating to pointed quandles for them was recently developed [12]. Automated reasoning for quandles and involutory quandles was considered in [3, 5, 6] for solving computational topology problems such as unknot detection and knot equivalence. Automated reasoning for tangles, without using quandles, was considered in [7] for the alternative "visual" proofs of unknotedness of knot diagrams. The main motivation for the equivalence problem for tangles considered in the present paper arose from [13] where a novel topological approach to quantum verification was proposed. While the manual proofs of tangle equivalence proposed there are appealing because of their transparent visual nature, no automation was considered. To perform automated reasoning we have used the automated theorem prover Prover9 and finite model finder Mace4 [11] and the more recent platform ProverX, providing access to both Prover9 and Mace4 via a programmatic interface and providing the ability to produce proof graphs [14].

5 Conclusions

We presented our automated reasoning approach to the algorithmic problem of checking equivalence of tangles, topological objects related to knots and links. The problem finds applications in the verification of quantum programs, where tangles model quantum circuits and their equivalence corresponds to the equivalence of quantum circuits. We used algebraic invariants of tangles called pointed involutory quandles, showing various means of faithfully encoding their equivalence checking as automated reasoning tasks. We illustrated the approach by an exploration of worked examples, inspired by the original quantum verification via tangles approach [13], considering several alternative means of encoding tasks. We provide an example of automated finite model finding for quandles, exemplifying a general means to establish non-equivalence of tangles. Furthermore, we demonstrate that automated proofs of strong isomorphism of these pointed involutory quandles may be able to guide an extraction of equivalence transformations for tangles - any progress in assisting/guiding human analysis for visual proof construction is already valuable, even if the ideal is to fully automate visual proof, whenever possible. The current approach has limitations, with automated move extraction not being fully automated, and if pointed quandle invariants are not complete, then they do not fully encapsulate tangle equivalence, so further research is required to establish the completeness of the theorem proving approach. In particular we are interested in conditions under which the converse of Proposition 2 holds.

References

[1] J. Scott Carter. A survey of quandle ideas. *arXiv:1002.4429*, 2010.

[2] Mohamed Elhamdadi and Sam Nelson. *Quandles*, volume 74. American Mathematical Soc., 2015.

[3] Andrew Fish and Alexei Lisitsa. Detecting unknots via equational reasoning, I: exploration. In Stephen M. Watt, James H. Davenport, Alan P. Sexton, Petr Sojka, and Josef Urban, editors, *Intelligent Computer Mathematics - International Conference, CICM 2014, Coimbra, Portugal, July 7-11, 2014. Proceedings*, volume 8543 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2014.

[4] Andrew Fish and Alexei Lisitsa. Detecting unknots via equational reasoning, I: exploration. *CoRR*, abs/1405.4211, 2014.

[5] Andrew Fish, Alexei Lisitsa, and David Stanovský. A combinatorial approach to knot recognition. In Ross Horne, editor, *Embracing Global Computing in Emerging Economies - First Workshop,*

EGC 2015, Almaty, Kazakhstan, February 26-28, 2015. Proceedings, volume 514 of *Communications in Computer and Information Science*, pages 64–78. Springer, 2015.

[6] Andrew Fish, Alexei Lisitsa, David Stanovský, and Sarah Swartwood. Efficient knot discrimination via quandle coloring with SAT and #-sat. In Gert-Martin Greuel, Thorsten Koch, Peter Paule, and Andrew J. Sommese, editors, *Mathematical Software - ICMS 2016 - 5th International Conference, Berlin, Germany, July 11-14, 2016, Proceedings*, volume 9725 of *Lecture Notes in Computer Science*, pages 51–58. Springer, 2016.

[7] Andrew Fish, Alexei Lisitsa, and Alexei Vernitski. Visual algebraic proofs for unknot detection. In P. Chapman, G. Stapleton, A. Moktefi , S. Perez-Kriz, and F. Bellucci , editors, *10th International Conference on Theory and Applications of Diagrams*, volume 10871 of *Lecture Notes in Computer Science*, pages 89–104. Springer, May 2018.

[8] David Joyce. A classifying invariant of knots, the knot quandle. *Journal of Pure and Applied Algebra*, 23(1):37 – 65, 1982.

[9] Louis H. Kauffman. Virtual knot theory. *European Journal of Combinatorics*, 20(7):663–691, 1999.

[10] S. V. Matveev. Distributive groupoids in knot theory. *Mat. Sb. (N.S.)*, 119(161)(1):78–88, 160, 1982.

[11] W. McCune. Prover9 and mace4. `http://www.cs.unm.edu/~mccune/prover9/`, 2005–2010.

[12] Runa Pflume. Generalizations of quandles to multi-linkoids, MSc Thesis. Georg-August-Universität, Göttingen, 2023.

[13] David J. Reutter and Jamie Vicary. Shaded tangles for the design and verification of quantum circuits. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 475(2224):20180338, 2019.

[14] I Binnick Robert. Proverx: rewriting and extending prover9. 2020.

[15] Mituhisa Takasaki. Abstractions of symmetric functions. *Tohoku Math. J.*, 49:143–207, 1943.

[16] Steven K Winker. *Quandles, knot invariants, and the n-fold branched cover*. PhD thesis, University of Illinois at Chicago, 1984.