TANGLED INTERACTIVE PROOFS

DANIEL D. MOSKOVICH[1] AND AVISHY Y. CARMI[2]

[1] *DIVISION OF MATHEMATICAL SCIENCES*
NANYANG TECHNOLOGICAL UNIVERSITY, SINGAPORE 637371

[2] *FACULTY OF ENGINEERING SCIENCES*
BEN-GURION UNIVERSITY OF THE NEGEV, ISRAEL

Abstract. We present a formalism to distribute an interactive proof between multiple interacting verifiers. Each verifier has a belief as to whether the claim to be proven is true or false, and verifiers convince one another based on input from a prover/oracle subject to a deformation parameter $\delta \in (0, 1)$ which introduces 'noise' into the system. Usual interactive proofs are recovered at the limit $\delta \to 1$. The utility of our theory is demonstrated by a network of nonadaptive 3-bit query PCP verifiers whose resulting soundness improves over the best known result for a single verifier of this class. There is a natural equivalence relation on such 'tangled interactive proofs' stemming from their analogy with low-dimensional topological objects. Two such proofs are considered equivalent if, roughly, one can be completely reconstructed from the other. This induces a notion of zero knowledge for our tangled interactive proofs.

1. Introduction

1.1. **A network of interacting verifiers.** The decision of a crowd can converge to a correct answer even when each individual has limited knowledge. This phenomenon is known as *wisdom of the crowds* (Surowiecki, 2005). In line with 'the many being smarter than the few', we extend the notion of interactive proof systems (Goldwasser, 1989) to a system in which a collection of verifiers interact to prove a claim together. Such a crowd of verifiers can collaborate to prove more than could be proven by any individual verifier in the crowd.

Consider a crowd of people called *verifiers* amongst whom rumours propagate about the veracity of a claim. For concreteness, we take the claim to be that a given word x is an element of a given set L called a *language*. At each discrete time $t = 0, 1, 2, \ldots$, one verifier called the *agent at time t* may choose to share a rumour with any number of other verifiers called *patients at time t*. Whether or not patient W's belief changes as a result of hearing a rumour from agent V depends on the content the rumour (does the rumour state $x \in L$ or does it state $x \notin L$), on chance (probability), and on input from the *prover* which has access to a knowledgeable entity called an *oracle*. Verifiers are memoryless and have no communication besides rumour sharing and prover consultation (or, in Section 5, direct oracle consultation). A verifier may be a patient at some times and an agent at other times.

Figure 1 illustrates the propagation of beliefs in a group of four people A, B, C, and D. An arrow going between any two cells indicates the two people have interacted within the designated time-frame. An arrow emanates from agent to patients. Note that there is only one agent in each time-frame. On the other hand, one agent may have many patients.

Can such a network be used to ultimately prove or disprove a claim? This depends on the beliefs of its constituents; If there is at least one group member whose beliefs reflect the true nature of the underlying claim, then such a system can be used as a theorem proving mechanism.

1.2. **Result summary.**

- In Section 2 we define a complexity class BraidIP (*braided interactive polynomial time*) which deforms and extends the complexity class IP (Theorem 2.4). BraidIP is in turn

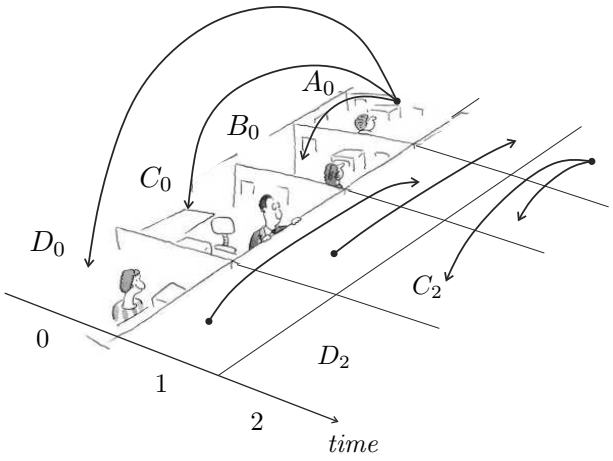


Figure 1. Rumour-passing group members and their evolving belief states.

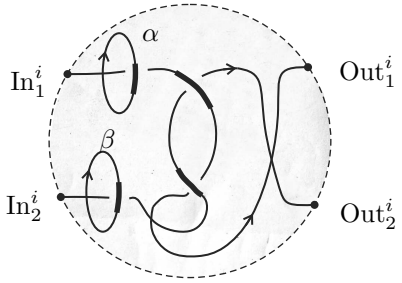
extended by a class TangIP (*tangled interactive polynomial time*). Unlike class IP, in classes BraidIP and TangIP the *soundness parameter* s and the *completeness parameter* c are required only to satisfy $0 < s < c \leq 1$ (it is possible that both constants are below $\frac{1}{2}$ or that both above $\frac{1}{2}$ and bounded away from 1). In both BraidIP and TangIP, the *verifier* of interactive proof theory is replaced by a network of interacting verifiers in which each agent verifier V communicates with its patients subject to a *deformation parameter* or *noise parameter* $\delta_V^t \in \mathbb{Q} \cap (0, 1)$.

The combinatorics of *tangle machines*, not the combinatorics of graphs, underlies classes BraidIP and TangIP(Carmi & Moskovich, 2014a,b). These particular deformations of IP are suggested by the tangle machine formalism, which originate in low-dimensional topology. Roughly, we can say that tangle machines capture the *distributive* nature of interactions in which a verifier queries a prover.

- In Section 5 we construct networks of PCP verifiers. We propose a particular network configuration which uses a number of 3-bit nonadaptive PCP verifiers with perfect completeness. This enables us to improve over the best achievable soundness for a single verifier in this class of PCPs.

  Three-bit query PCP verifiers exhibit a tradeoff between non-adaptiveness of the verifier and perfect completeness. An adaptive 3-bit PCP with perfect completeness and soundness above $\frac{1}{2}$ can be constructed (Guruswami, 1998). When the verifier is restricted to be nonadaptive, however, the best soundness parameter of 3-bit PCPs has been conjectured to be $\frac{5}{8}$ (Zwick, 1998). This conjecture implies that getting a $\frac{5}{8}$-approximation for 3-bit satisfiable CSP is NP-hard. The best known result for such a non-adaptive verifier has soundness parameter $\frac{20}{27} + \sigma$ for some arbitrarily small $\sigma > 0$ (Khot, 2006).

  We let a number of basic nonadaptive 3–bit PCP verifiers all of whose soundness is $\frac{6}{8}$ to interact according to the following tangle diagram.



We show that this configuration achieves perfect completeness, and soundness of $\frac{5}{8} + \frac{1}{24}$, which is near the conjectured bound for a *single verifier* in PCP. In this sense, a tangle of verifiers behaves almost like one single very good verifier.

It is worthwhile noting that our networks do not reproduce the effect of adaptiveness in the usual sense, as our verifiers do not correlate their queries.

- The formalism of tangle machines provides a natural notion of equivalence for networks of interactions which decide languages in BraidIP and in TangIP. Namely, two such networks are equivalent if they are related by a finite sequence of certain local modifications, called *local moves* listed in Figure 9. If two networks of interactions are equivalent, then any interactive proof carried out on the basis of one of them can uniquely be simulated on the other. In particular, equivalent networks decide the same languages. The sense in which these local moves are canonical is discussed in (Carmi & Moskovich, 2014a), and their low dimensional topological interpretation is discussed in (Carmi & Moskovich, 2014b). The two most important local moves are R2 and R3. The R2 move tells us that our computations are reversible in distribution— *i.e.* that given the statistics $|W_{t+1})$ for the patient belief, and $|V_t)$ for the agent belief, we can uniquely reconstruct $|W_t)$. The R3 move induces a notion of *zero knowledge* for our tangled networks.

This note arose from discussions with M. Buliga and L. Kauffman about the meaning of tangle machines in the theory of computation. A number of simple computational interpretations for tangle machines are discussed in (Carmi & Moskovich, 2014a), and the present note discusses another such interpretation. The goal of this note is not to improve parameters of interactive provers. Rather, it is to explore a new framework for interactive proofs, whose inspiration stems from low dimensional topology.

The literature contains several other topologically inspired models of computation, in which topologically equivalent objects represent bisimilar computations (see *e.g.* (Kauffman, 1994; Meredith & Snyder, 2010; Vicary, 2012)) or in which low dimensional topological modifications are used directly to compute (see *e.g.* (Buliga, 2011b; Buliga & Kauffman, 2013)). To the best of the authors' understanding, none of these are directly related to the present work.

1.3. **Organization of this note.** The next section introduces the basic ingredients of a network and explains the notation we use throughout this work to represent its statistics. It proceeds in defining the class of languages decided by a network, BraidIP, in anticipation to the result IP $\subseteq$ BraidIP which is obtained in the subsequent section. It also describes the diagrammatic language used inhere to represent networks.

Section 3 provides means to deform any standard IP system. The deformation thus obtained may then be used to compose a network. A proof to the above theorem is then provided which describes a particular network that decides any $L \in$ IP.

Section 4 extends the former one by describing more efficient network configurations. It also defines the class TangIP which includes BraidIP.

Section 5 demonstrates the utility of the theory in the context of PCP verifiers. One of the interesting consequences of this is described in the Result Summary above.

Section 6 discusses the notion of network equivalence. It constitutes a link between interactive proofs and low-dimensional topology.

Some open issues are listed in the final section.

2. Distributing knowledge by deformation

How does our formalism fit into the landscape of interactive proof systems?

2.1. **Deformation of a single interaction.** The belief of a verifier W at time t is modeled as a Bernoulli random variable W_t whose realizations w_t are either |True) or |False). We interpret $w_t =$ |True) as 'W believes at time t that $x \in L$', and we interpret $w_t =$ |False) as 'W believes at time t that $x \notin L$'.

Consider an interaction at time t with agent V, one of whose patients is W. The realization w_{t+1} of W_{t+1} may equal either the belief of the agent v_t or the belief of the patient w_t. In other words, W either retains his belief or is 'convinced' by V to change his belief to that of V (we use male pronouns for verifiers and a female pronoun for the prover). Whether or not V 'succeeds in convincing W' depends on a message ξ_t from a *prover* Π with access to an *oracle*.

Only the belief of patients changes at an interaction. The agents and of verifiers who do not participate in the interaction do not change their beliefs, so in particular $v_{t+1} = v_t$ always.

There are three constants associated to the agent V at an interaction at time t: A *completeness parameter* c_V^t, a *soundness parameter* s_V^t with $0 < s_V^t < c_V^t \leq 1$, and a *deformation parameter* $\delta_V^t \in \mathbb{Q} \cap (0,1)$. In this note we will assume that these three parameters are the same for all agents in the network, and s_v^t, c_V^t, δ_V^t will be written s, c, δ correspondingly.

Our basic requirement for an interaction is that the following pair of inequalities be satisfied:

$$\begin{aligned} &\textit{(deformed completeness)} \quad x \in L \longrightarrow \Pr(W_{t+1} = v_t \mid V_t = v_t) \geq c\delta; \\ &\textit{(deformed soundness)} \quad\;\; x \notin L \longrightarrow \Pr(W_{t+1} = v_t \mid V_t = v_t) \leq s\delta. \end{aligned} \tag{1}$$

Remark 2.1. The deformed completeness and soundness, $c\delta$ and $s\delta$, may both be below $\frac{1}{2}$ or may both be above $\frac{1}{2}$ and bounded away from 1.

In the limit $\delta \to 1$, specific values of c and of s turn the pair of inequalities (1) into familiar pairs of inequalities in interactive proof theory (Arora & Barak, 2009). For example, for $s = 2^{-|x|^a}$ and $c = 1 - 2^{-|x|^b}$, where $a, b > 0$, we obtain the completeness and soundness constraints of an IP verification.

2.2. **Statistics of beliefs and interactions.** When keeping track of the beliefs of many different verifiers at many different times, it is cumbersome to work directly with 1. Instead, we introduce a shorthand to keep track of the belief of a verifier, both if $x \in L$ and also if $x \notin L$, in a single expression.

The *belief statistics* $|W_t)$ of verifier W at time t is written:

$$|W_t) \stackrel{\text{def}}{=} a\,|\text{True}) + b\,|\text{False})\,, \tag{2}$$

where $a \in [0,1]$ denotes the *greatest lower bound* for the belief of W that $x \in L$ at time t conditioned on this belief being indeed true, and $b \in [0,1]$ denotes the *greatest lower bound* for the belief of W that $x \notin L$ at time t conditioned on this belief being indeed true. Note that $a + b$ need not equal 1. In particular:

$$\begin{aligned} x \in L \quad &\longrightarrow\; a \leq \Pr(W_t = |\text{True})); \\ x \notin L \quad &\longrightarrow\; b \leq \Pr(W_t = |\text{False})). \end{aligned} \tag{3}$$

An interaction between W and V at time t is concluded with W either accepting the belief of V or sticking to his own belief. Denoting by h the probability (or more precisely a lower bound on it) of W accepting the belief of V conditioned on either parties beliefs, $w_t \neq v_t$, an interaction is described by:

$$|W_{t+1}) = |W_t)^{|V_t)} \stackrel{\text{def}}{=} (1-h)\,|W_t) + h\,|V_t)\,. \tag{4}$$

Remark 2.2. Belief statistics written as in (2) facilitate calculations of probabilities across a network. This tool is used throughout this work in an essential way (some examples would be given shortly). Here we explain how (2) and (4) combine to give a compact way of representing *two entirely different interactions*, one assuming $x \in L$ and the other assuming $x \notin L$. This may be slightly confusing at first, so the reader's attention is called to this point.

Owing to (4) the probabilities anywhere in the network depend on the parameter h. We may do all calculations and treat it as a formal parameter. Having in mind that in practice an interaction is a procedure terminating with the statistics in (1), the parameter h is set using either one of the values $h \stackrel{\text{def}}{=} c\delta$ or $h \stackrel{\text{def}}{=} s\delta$, depending on whether or not x is in L. To verify that a network decides L we will repeat the calculations twice, first for the case where $x \in L$ and then for $x \notin L$. In the former case we will be interested only in the coefficient of $|\text{True})$ whereas in the latter case we will be interested only in the coefficient of $|\text{False})$.

Here is an illustrative calculation for a single interaction. Let $W_t = |\text{False})$ and $V_t = |\text{True})$. Invoke (4) using the completeness and soundness parameters in (1): first using $h = c\delta$ and then

using $h = s\delta$. Hence,

$$|W_{t+1}) = c\delta\,|\text{True}) + (1 - s\delta)\,|\text{False}) \longrightarrow \begin{cases} c\delta\,|\text{True}) + (\cdots)\,|\text{False})\,, & x \in L; \\ (\cdots)\,|\text{True}) + (1 - s\delta)\,|\text{False})\,, & x \notin L. \end{cases}$$

This interaction decides L only if $c\delta > \frac{1}{2}$ and $1 - s\delta > \frac{1}{2}$.

2.3. **Expressive power of a network: The class** BraidIP**.** Verifiers in our framework are assumed to be probabilistic polynomial-time Turing machines whose beliefs are either internal states or stored on tapes. Similarly, an interaction is a polynomial-time procedure. Consider now a crowd of verifiers $V^1, V^2, \ldots, V^\mu$ whose initial beliefs at time $t = 0$ are $\left|V_0^1\right), \left|V_0^2\right), \ldots, |V_0^\mu)$. Allow them to interact at times $t = 0, 1, 2, \ldots, \chi$, subject to parameters $0 < c < s \leq 1$ and $\delta \in \mathbb{Q} \cap (0, 1)$. We write M for this sequence of interactions. A language $L \subseteq \{0,1\}^*$ is said to be *decided* by M if M contains a verifier V whose belief at time χ is $|V_\chi) \stackrel{\text{def}}{=} a\,|\text{True}) + b\,|\text{False})$, such that for a fixed constant $\kappa > 0$, we have:

$$\begin{aligned} x \in L &\longrightarrow a \geq \tfrac{1}{2} + |x|^{-\kappa}\,; \\ x \notin L &\longrightarrow b \geq \tfrac{1}{2} + |x|^{-\kappa}\,. \end{aligned} \tag{5}$$

This definition depends on the choice of $\kappa > 0$. The class *braided interactive polynomial time* (BraidIP) consists of those languages which are decidable for any fixed $\kappa > 0$ by some network M in time χ, polynomial in $|x|$. We denote this class BraidIP$\{\delta, \chi\}$ where χ is *the number of interactions in* M.

Remark 2.3. The definition of class BraidIP is similar in spirit to the class BPP. We will see below that it includes class IP. Letting (5) reflect the class PP (*i.e.* taking strict inequalities and right-hand constants equal to $\frac{1}{2}$) will result in networks that decide any $L \in$ IPP.

In Section 3 we will prove the following result:

Theorem 2.4. IP $\subseteq$ BraidIP $\{\delta, \chi\}$ *where:*

$$I(c\delta) < \chi < \frac{1}{I(1 - s\delta)}, \tag{6}$$

with $I(p) \stackrel{\text{def}}{=} -p^{-1} \log p$*. The growth rate of* χ *is* $\mathcal{O}(\frac{1}{\delta})$ *as* $\delta \to 0$*.*

2.4. **Tangle of beliefs.** The networks inhere admit a convenient diagrammatic description. We represent an interaction as a wire cutting through other wires (see Figure 2). The overcrossing wire, which becomes slightly thickened in an interaction, carries the belief statistics of an agent whereas the undercrossing wires carry the belief statistics of her patients. An example of many concatenated interactions is given in Figure 3.

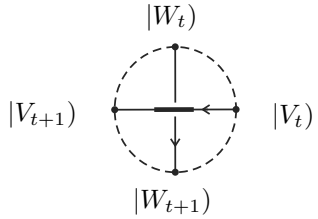
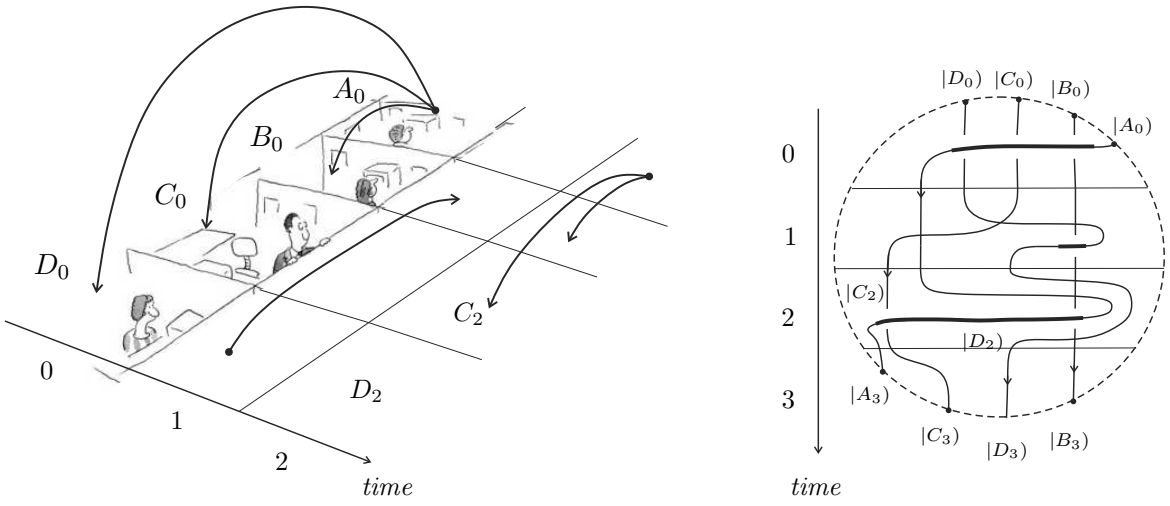


Figure 2. Diagrammatic representation of an interaction.

The diagram representing the rumour passing system in Figure 1 is a *tangle*. It is shown on the right in Figure 3. Inhere such pictures are used to represent the flow of beliefs within a network of interacting machines (verifiers). This by itself is a sufficient reason to name them *tangle machines*.



rumour-passing network *tangle machine (flow of beliefs)*

Figure 3. A rumour-passing network and its respective diagram.

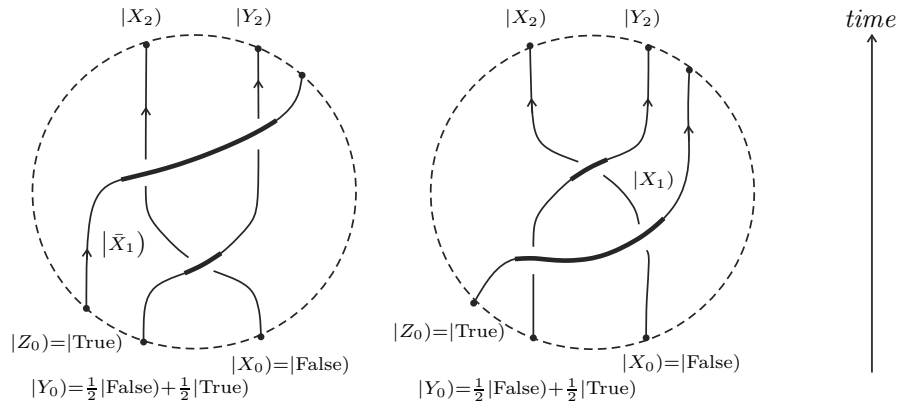


Figure 4. Equivalent networks of interactions.

2.5. **An example.** The capacity of a network to prove or disprove a claim is an emergent property. Out of a number of uncertain interactions, none of which prove the claim, the truth may eventually materialize. As an example, consider the two pairs of two consecutive interactions pictured in Figure 4. Both sequences involve three verifiers, designated X, Y and Z. Their initial beliefs are shown at the bottom.

Set the parameters to $s \stackrel{\text{def}}{=} \frac{1}{2}$, $c \stackrel{\text{def}}{=} 1$, and $\delta \stackrel{\text{def}}{=} \frac{1}{2}$. Allow the verifiers to interact in the left diagram. At time $t = 2$, beliefs X_0 and Y_0 of X and of Y have been updated to X_2 and to Y_2 (Z does not change his belief). The distributions are:

$$|X_2) = \Big(|X_0)^{|Y_0)}\Big)^{|Z_0)} \longrightarrow \begin{cases} \frac{3}{8}\,|\text{False}) + \frac{5}{8}\,|\text{True})\,, & x \in L; \\ \frac{21}{32}\,|\text{False}) + \frac{11}{32}\,|\text{True})\,, & x \notin L. \end{cases} \tag{7}$$

Hence,

$$\begin{aligned} |X_2) &= \Big(|X_0)^{|Y_0)}\Big)^{|Z_0)} &&= \tfrac{21}{32}\,|\text{False}) + \tfrac{5}{8}\,|\text{True})\,, \quad \text{(left network)}; \\ |X_2) &= \Big(|X_0)^{|Z_0)}\Big)^{(|Y_0)^{|Z_0)})} &&= \tfrac{21}{32}\,|\text{False}) + \tfrac{5}{8}\,|\text{True})\,, \quad \text{(right network)}. \end{aligned} \tag{8}$$

Similarly,

$$|Y_2) = |Y_1) = |Y_0)^{|Z_0)} = \frac{3}{8}\,|\text{False}) + \frac{3}{4}\,|\text{True})\,. \tag{9}$$

From this we see that X decides correctly at time $\chi = 2$ with probability at least $\frac{5}{8}$ or $\frac{21}{32}$, depending on whether $x \in L$ or $x \notin L$. Both of these are greater that $\frac{1}{2}$, so the pair of inequalities 5, for a suitable $\kappa > 0$, a protocol underlied by the above interactions will succeed in deciding, at $|X_2)$, whether or not $x \in L$.

Note again that

$$|Y_1) = |Y_0)^{|Z_0)} = h\,|Z_0) + (1-h)\,|Y_0)\,,$$

and we evaluate with $h = \frac{1}{2}$ for the coefficient of |True) and with $h = \frac{1}{4}$ for the coefficient of |False). This is the same for all interactions in both diagrams.

Note that both diagrams in Figure 4 have the same *initial beliefs* $|X_0)$, $|Y_0)$, and $|Z_0)$ and the same *terminal beliefs* $|X_2)$, $|Y_2)$, and $|Z_2)$, and differ only in the belief of X at time $t = 1$. Thus, these two diagrams underlie *equivalent* deformed interactive proofs, each of which can uniquely be reconstructed from the other, which decide the same languages, but which differ at an intermediate step. This equivalence is the topic of Section 6.

3. Deformation of an IP system

In this section, we show how we may deform an IP system with any soundness parameters $0 < s < \frac{1}{2} < c \leq 1$, for any deformation parameter $\delta \in \mathbb{Q} \cap (0,1)$. The completeness and soundness parameters of the deformed system will be $s\delta$ and $c\delta$ correspondingly. The deformation parameter δ serves to introduce noise between the prover and the verifiers. In the $\delta \to 1$ limit we recover IP, and the information obtained by a verifier at each interaction shrinks as $\delta \to 0$. But Theorem 2.4 proves that we can recover IP from BraidIP by concatenating many consecutive deformed interactions.

Before describing how IP may be deformed, we outline the major differences between a single interaction in IP and BraidIP:

Description	IP system	Deformed IP system
Participants	Verifier, Prover	Many verifiers (patient), Verifier (agent), Prover
Verifier 'state of mind'	Accept/Reject	Belief True/False
Conclusion	Verifier decides Accept/Reject	If the two verifiers do not agree then the patient may change his belief.
Completeness, Soundness	c, s	$c\delta, s\delta$

Verifiers may be thought of as probabilistic polynomial-time algorithms running on Turing machines. The prover is assumed to be an all-powerful Turing machine with an access to a proof or certificate to the effect that $x \in L$ (Goldwasser, 1989). A deformed system requires any verifier to be equipped with an internal state or tape cell storing his current belief.

3.1. **Two approaches to deform IP.**

We present two approaches to deform an IP protocol. The end result is the same, but the 'story' is different.

3.1.1. *Agent and patient as a single verifier.* We may think of a patient W and an agent V of an interaction at time t as representing different aspects of a single verifier. In this approach we conceive of W and V as being a single unit (W, V). The verifier (W, V) transmits to the prover Π the belief of both W and V. We may imagine W and V as litigants in a court case, presenting their claims to the judge Π, where W is the defendant and V is the plaintiff. If both W and V make the same claim, then Π throws the case out (*i.e.* $W_{t+1} = w_t$ and $V_{t+1} = v_t$). On the other hand, if V disagrees with W, then (W, V) query the prover Π according to the original interactive protocol. If according to the original protocol, W's claim should be accepted, then Π rules in W's favour (*i.e.* $W_{t+1} = w_t$ and $V_{t+1} = v_t$). But if according to the original protocol W's claim should be rejected and V's claim should be accepted, then Π picks an integer uniformly

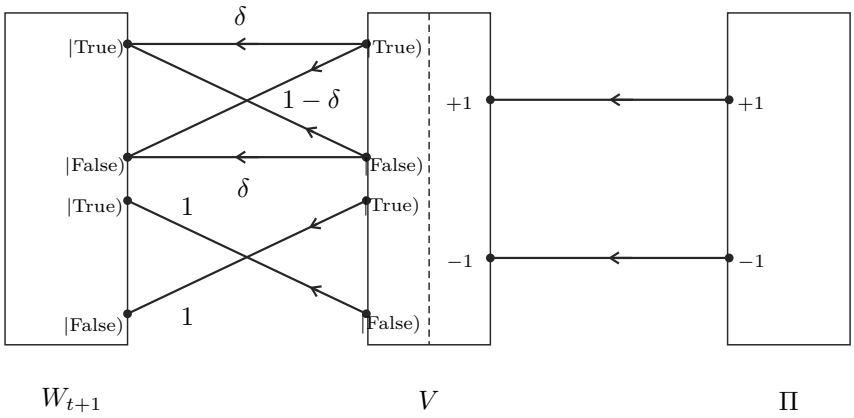


Figure 5. Communication between W and V when $v_t \neq w_t$. The channels between W and V are symmetric. The labels above edges indicate transition probabilities. The box of V is divided into two sections representing her belief v_t (left section) and the outcome of his verification ξ_t (right section).

at random between 1 and N. If the number Π picked is less than δN, then Π rules in favour of V (*i.e.* $W_{t+1} = v_t$ and $V_{t+1} = v_t$). Otherwise she rules in favour of W.

Perhaps δ represents a chosen standard of 'reasonable doubt'. Constants s and c perhaps represent constants associated with the mechanics of the courthouse procedure. Note that as $\delta \to 1$, a single interaction involving two verifiers with opposite beliefs recovers IP.

3.1.2. *Agent and patient as Alice and Bob.* The following approach has an information-theoretic interpretation. Consider the prover Π as an information source, the agent verifier V as an encoder, and the patient verifier W as a decoder. The query information transmitted from W to Π is relayed via a perfect communication channel (*i.e.* there is no loss of information in this direction). The replies from Π are passed on to V who encodes them and transmits them back to W, this time through a noisy channel. This means that the prover replies emerge corrupted on W's end, which consequently influences his decision.

Introducing a noisy channel into the formalism restricts the information obtained by the patient verifier from the prover. It tempting to state that the combination prover-agent-noisy channel behaves like a mendacious agent, in that the agent V decided whether to 'tell the truth' or to 'lie' to W. But to think of V as a mendacious agent is inaccurate. For one thing, the agent's strategy whether or not to reliably relay Π's replies to W must account for the beliefs of both V and W, either one of which may not be correct. His behavior does not stem from him being more knowledgeable; rather we may think of it as a manifestation of his own beliefs.

It is important to note that although V receives the replies from Π he is not allowed to use this information to update her own belief. One can think of protocols taking advantage of the fact that V is not aware of W's queries wherein this restriction follows naturally. For now it is enough to assume that V will not use the prover replies for his own benefit.

Here is how such a protocol may run. Upon disagreement between W and V, *i.e.* $v_t \neq w_t$, the patient W sends his queries to Π. The replies to W's queries are then sent by Π to the agent V. At this point, V, who is a verifier much like W, runs his own verification test on the prover replies. He obtains $\xi_t = 1$ for accept/true and $\xi_t = -1$ for reject/false. In case where $\xi_t = 1$ he tampers with the prover replies such that when they are received by W his verification would indicate v_t with probability δ. In case where $\xi_t = -1$ the agent V tampers with the prover replies such that the test of W would indicate $\neg v_t$.

Implicit in the above protocol is the fact that the capacity of V to deceive W is limited by V's own belief. If her belief, v_t, coincides with the true nature of claim then he may potentially have more power to deceive W.

The protocol just described underlies a noisy symmetric channel between W and V. If $\xi_t = 1$, this channel is characterized by δ and has a capacity of $1 - H_2(\delta)$, where $H_2(\delta)$ is the Shannon entropy of a Bernoulli random variable with parameter δ. It induces a maximal loss of 1 bit of

information for $\delta = \frac{1}{2}$. An illustration of the communication between the three parties patient-agent-prover is given in Figure 5.

3.1.3. *Further metaphors for deformed interactions.* The agents in our picture all receive messages from the same oracle. This essentially suggests that a network of interactions is a construct *quantizing* the oracle knowledge. At every location within the network only a quanta of this knowledge is used by way of interaction between a patient and an agent. Later on we will show that although a single interaction may be limited in its capacity to prove the claim, the proof may yet emerge somewhere in the network depending on its topology.

The triple patients-agent-oracle brings to mind some basic models of reasoning and information transfer. Perhaps a patient is an entity whose beliefs reflect both prior knowledge and observations. The patient is exposed to a genuine phenomenon, which the patient has not seen before. The phenomenon, which is the metaphor for an oracle, is beyond the comprehension of the patient and hence a number of observations are collected in an attempt to reach a definitive conclusion. These observations, however, may be distorted by limitations of the patient measuring apparatus, or perhaps they contradict prevailing explanations and beliefs. In either cases observations contain, or otherwise introduce, uncertainty. Observations are the metaphor for agents. What the patient tries to accomplish underlies the Bayesian inference paradigm.

Here is another metaphor. A patient is a decoder, an oracle is an information source, and an agent is an encoder who relays the encoded oracle message through a noisy communication channel. Alternatively, an agent-patient pair is a verifier and the oracle is a prover who relays a message through a noisy communication channel. All metaphors reflect knowledge transfer subject to uncertainties.

3.2. **Probabilistic theorem proving in networks.**

Proof of Theorem 2.4. We explicitly construct a configuration of interactions which decide a language L in IP. This configuration, which is illustrated in Figure 6 for the case $\chi = 4$ is a scaled up version of that in Figure 4. It involves $\chi + 1$ verifiers W and $V^1, V^2, \ldots, V^\chi$, and χ interactions. The parameters of all interactions are the same, and are c, s, δ.

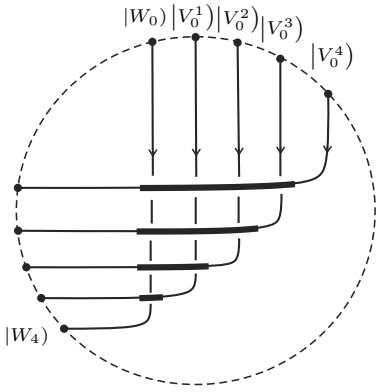


Figure 6. An interactive BraidIP theorem proving network with 5 verifiers W, V^1, V^2, V^3, and V^4, and 4 interactions.

Let $L \in$ IP. The initial beliefs at time $t = 0$ are set to $|W_0\rangle \stackrel{\text{def}}{=} |\text{False}\rangle$ and

$$|V_0^i\rangle \stackrel{\text{def}}{=} \begin{cases} \frac{1}{2}|\text{False}\rangle + \frac{1}{2}|\text{True}\rangle\,, & 1 < i < \chi; \\ |\text{True}\rangle\,, & i = \chi. \end{cases} \tag{10}$$

Thus, at time zero there are two verifiers with opposite beliefs and $\chi - 1$ verifiers whose initial belief is that the claim $x \in L$ is 50% true and 50% false.

Calculating the output statistic $|W_\chi\rangle$ (which occurs at time χ) yields

$$|W_\chi\rangle \longrightarrow \begin{cases} (1-c\delta)^\chi\,|\text{False}\rangle + [1 - c\delta - (1-c\delta)^\chi]\left(\frac{1}{2}|\text{False}\rangle + \frac{1}{2}|\text{True}\rangle\right) + c\delta\,|\text{True}\rangle\,, & x \in L; \\ (1-s\delta)^\chi\,|\text{False}\rangle + [1 - s\delta - (1-s\delta)^\chi]\left(\frac{1}{2}|\text{False}\rangle + \frac{1}{2}|\text{True}\rangle\right) + s\delta\,|\text{True}\rangle\,, & x \notin L. \end{cases} \tag{11}$$

From (11) we see that the configuration in Figure 6 decides L if and only if:

$$\begin{aligned} x \in L &\longrightarrow (1-c\delta)^{\chi} < c\delta; \\ x \notin L &\longrightarrow (1-s\delta)^{\chi} > s\delta. \end{aligned} \tag{12}$$

From here we obtain the following bounds for χ:

$$\frac{\log(c\delta)}{\log(1-c\delta)} < \chi < \frac{\log(s\delta)}{\log(1-s\delta)}. \tag{13}$$

Equation 6 follows upon noting that $\log(1-p) < -p$ for any $p \in (0,1)$. Therefore:

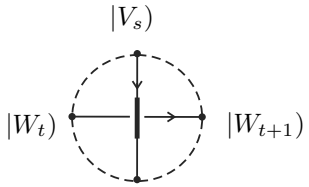
$$-\frac{1-p}{\log(1-p)} > \frac{\log p}{\log(1-p)} > -\frac{\log p}{p}. \tag{14}$$

For χ within these bounds, the above configuration decides L. □

Remark 3.1. Equation (6) tells us that χ has approximately the same behaviour as $\frac{1}{\delta}$. By definition of BraidIP, χ's growth rate is polynomial in the word length $|x|$, and so therefore δ is asymptotically bounded below by approximately one over a polynomial in $|x|$.

4. Efficient IP strategies

4.1. **The complexity class** TangIP**.** We may extend class BraidIP by allowing each verifier to have its own 'local' time parameter, so that a patient belief W_t may interact with an agent belief V_s for $s \neq t$, and become updated to W_{t+1}.



We also allow verifiers to travel backwards or forwards in time and to update their previous or future beliefs, so that V_t may be updated by agent V_s to become V_{t+1}, where V_t and V_s are beliefs of one and the same verifier. We write M for a network of such concatenated interactions, subject to parameters $0 < c < s \leq 1$ and $\delta \in \mathbb{Q} \cap (0,1)$. An example of such a network is given in Section 4.2.

A language $L \subseteq \{0,1\}^*$ is said to be *decided* by M if M contains a verifier V whose belief at time χ is $|V_\chi) \stackrel{\text{def}}{=} a\,|\text{True}) + b\,|\text{False})$, such that for a fixed constant $\kappa > 0$ the inequalities (5) are satisfied.

The class *tangled interactive polynomial time* (TangIP) consists of those languages which are decidable for any fixed $\kappa > 0$ by some network M which contains χ interactions, where χ is polynomial in $|x|$. We denote this class TangIP$\{\delta, \chi\}$.

By Theorem 2.4 we know that

$$\text{IP} \subseteq \text{BraidIP} \subseteq \text{TangIP}.$$

We wonder about the connection between our classes BraidIP and TangIP and multi-prover IP (MIP) (Ben-Or, 1988). In particular, we wonder whether MIP $\subseteq$ BraidIP or MIP $\subseteq$ TangIP, particularly if we allow different interactions to have different parameters (in this note all interactions are required to have the same parameters because that's all we need, but there is no obstruction to considering the more general case).

4.2. **The Hopf–Chernoff configuration.** Consider an IP system whose soundness s is nearly equal to its completeness c but for a small constant $\epsilon(|x|)$ that depends on the word length $|x|$, *i.e.* $c - s = \epsilon(|x|)$. As $\epsilon(|x|) \to 0$, the IP system becomes inefficient in the sense that it accepts every word with probability nearly c regardless of its membership in L. Yet we can still construct a deformed IP system that decides L. One may wonder how the number of interactions in such a system is affected by the decreasing gap $\epsilon(|x|)$.

The number of interactions in a network of the form given in Figure 6 is implicit in Theorem 2.4. Fix $\delta \in \mathbb{Q} \cap (0,1)$ and note from Equation (6) that, as $\epsilon(|x|)$ decreases, the values

bounding χ become nearly identical. But χ is an integer, so the two bounds must have an integer between them. In general, that means that the distance between them is at least 1. Therefore, the number of interactions grows as $\epsilon(|x|)$ decreases. We may need to take $\delta \to 0$ as $\epsilon(|x|) \to 0$. In fact it can be shown that in this case $\delta(|x|) = \mathcal{O}(\epsilon(|x|))$ which means that we require $\chi = \mathcal{O}(1/\epsilon(|x|))$ interactions.

Can fewer interactions decide L ? The configuration in Figure 7 decides L for any $\epsilon(|x|)$ using substantially less than $\mathcal{O}(1/\epsilon(|x|))$ interactions. This machine is a concatenation of a number of identical smaller configurations of interactions, denoted $M_0, M_1, M_2, \ldots$. When concatenated to form a single configuration, we require approximately $\chi = \mathcal{O}(\log(1/\epsilon(|x|)))$ copies of M_0 to decide L (the precise argument is given below). If we were to trace its colours (probability generating functions) we would notice that it behaves much like a repetition of a binary random experiment (*e.g.* coin flipping), hence the magnitude of χ. We have named this configuration the *Hopf–Chernoff configuration* suggesting both to its structure and, to some extent, its functionality.

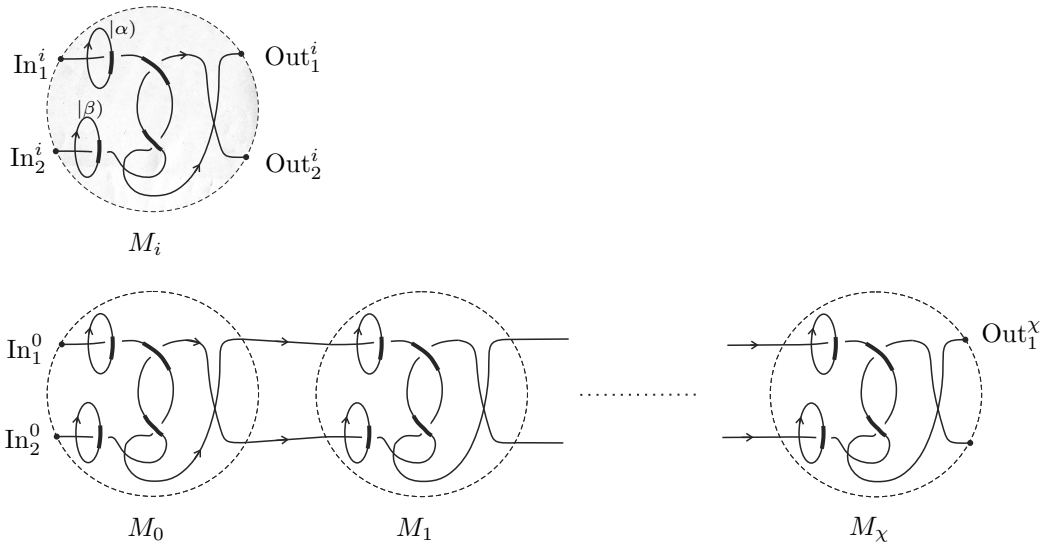


Figure 7. Hopf–Chernoff configuration(open version).

Theorem 4.1 (Hopf–Chernoff configuration)**.** *Consider the configuration of interactions in Figure 7 which underlies a deformed* IP *system with completeness* $c\delta$ *and soundness* $(c-\epsilon)\delta$*, where* $\epsilon > 0$*. There exists a pair of beliefs,* α *and* β*, independent of any other belief in the machine, such that for any given set of initial beliefs* In_j^0 *the machine decides any* $L \in \mathrm{IP}$ *using* $\chi = \mathcal{O}(\log(1/\epsilon))$ *submachines (and* 4χ *interactions). In particular, letting*

$$\begin{aligned} |\alpha) &= \left(\tfrac{1}{4} + \tfrac{1}{12}\epsilon\delta\right)|\mathrm{True}) + \left(\tfrac{3}{4} - \tfrac{1}{12}\epsilon\delta\right)|\mathrm{False})\,; \\ |\beta) &= \left(1 - \tfrac{1}{2}c\delta + \tfrac{1}{12}\epsilon\delta\right)|\mathrm{True}) + \left(\tfrac{1}{2}c\delta - \tfrac{1}{12}\epsilon\delta\right)|\mathrm{False})\,. \end{aligned} \tag{15}$$

yields

$$\mathrm{Out}_1^\chi \longrightarrow \begin{cases} \left[\tfrac{1}{2} + \tfrac{1}{12}\epsilon\delta\right]|\mathrm{True}) + \left[\tfrac{1}{2} - \tfrac{1}{12}\epsilon\delta\right]|\mathrm{False})\,, & x \in L; \\ \left[\tfrac{1}{2} - \tfrac{1}{12}\epsilon\delta\right]|\mathrm{True}) + \left[\tfrac{1}{2} + \tfrac{1}{12}\epsilon\delta\right]|\mathrm{False})\,, & x \notin L. \end{cases} \tag{16}$$

namely, $\mathrm{Out}_1^\chi = \left[\tfrac{1}{2} + \tfrac{1}{12}\epsilon\delta\right]|\mathrm{True}) + \left[\tfrac{1}{2} + \tfrac{1}{12}\epsilon\delta\right]|\mathrm{False})$.

Proof. The proof is deferred to the Appendix. □

The Hopf–Chernoff configuration is a recursive structure which is guaranteed to converge irrespective of its initial beliefs In_j^0. In fact it represents a two-dimensional homogeneous irreducible Markov chain whose rate of convergence is $\mathcal{O}(2^{-\chi})$. Its stationary distribution, which depends on whether $x \in L$ or $x \notin L$, is akin to (16). By virtue of its convergence properties we may just let it run forever (*i.e.* $\chi \to \infty$) knowing that it will eventually reach a stationary distribution not far from (16). For that reason we may as well substitute the open network in Figure 7 with its closed counterpart in Figure 8.

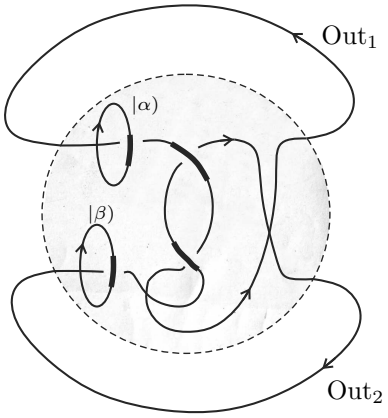


Figure 8. Hopf–Chernoff configuration (closed version).

5. PCP networks

In this section we consider networks of PCP verifiers (Arora, 1998). Consequently we replace the prover with an oracle Π who holds the proof encoding π. Thus, a PCP verifier queries locations in π by passing their addresses to the oracle. We assume all PCP verifiers to be *nonadaptive*.

We adopt the notation of (Moshkovitz, 2010) and denote by $\mathrm{PCP}_{c,s}[r(n), q(n)]_\Sigma$ the class of languages having a PCP verifier with completeness c and soundness s, which makes use of $r(n)$ random bits and $q(n)$ queries to a proof over an alphabet Σ, where $n \stackrel{\text{def}}{=} |x|$ is the word length. The original PCP theorem states $\mathrm{NP} \subseteq \mathrm{PCP}_{1,\frac{1}{2}}[\mathcal{O}(\log n), \mathcal{O}(1)]$ where $\Sigma = \{0, 1\}$ is conventionally omitted. The numbers $r(n)$ and $q(n)$ are referred to as the verifier's *randomness complexity* and *query complexity*.

5.1. **Deformed PCP verification.** A PCP network is a network whose interactions are governed by deformed PCP verifiers. To illustrate how such a deformation may be obtained it is useful to consider a particular PCP verifier. The Håstad 3-bit PCP verifier in (Håstad, 1997) makes a perfect example. It employs a linear test using only three bits obtained from the proof encoding (Long Code). Using the notation in Section 3, Π is now an oracle holding an encoding of the proof, which in this case is a Long Code representation of an assignment for a CSP instance (constrained satisfaction problem). In the original setting, where only one verifier is present, W provides the addresses of three bits to Π. Once these are received he is able to compute a certificate $\xi_t \in \{-1, 1\}$ according to which he decides to either accept ($\xi_t = 1$) or reject ($\xi_t = -1$).

Now let us describe a corresponding deformed version of this procedure. Suppose the two verifiers disagree, $v_t \neq w_t$, where $v_t, w_t \in \{|\mathrm{True}), |\mathrm{False})\}$. In this case the interaction proceeds as follows. The patient verifier W provides the addresses of three bits to Π. These bits are received by the agent verifier V which computes his own certificate ξ_t. He then may decide to flip one out of the three bits per the following rules:

- $(\xi_t = 1) \wedge (v_t = |\mathrm{True})) \longrightarrow V$ flips a bit with probability $1 - \delta$.
- $(\xi_t = 1) \wedge (v_t = |\mathrm{False})) \longrightarrow V$ flips a bit with probability δ.
- $(\xi_t = -1) \wedge (v_t = |\mathrm{True})) \longrightarrow V$ flips no bit.
- $(\xi_t = -1) \wedge (v_t = |\mathrm{False})) \longrightarrow V$ flips a bit.

The fabricated set of bits is then sent back to W who computes his own certificate. This protocol is an actualization of the communication channel in Figure 5.

5.2. **Query complexity and randomness complexity of a network.** Consider a PCP verifier whose completeness is possibly imperfect, $c < 1$, and whose soundness is $s = c - \epsilon$. By Theorem 4.1 a deformed version of such a PCP verifier running at every interaction in the Hopf–Chernoff network results in an output whose completeness and soundness parameters are,

respectively, $\frac{1}{2}+\frac{1}{12}\epsilon\delta$ and $\frac{1}{2}-\frac{1}{12}\epsilon\delta$. Such a machine uses $\mathcal{O}(\log(1/\epsilon))$ interactions which amounts to $\mathcal{O}(q(n)\log(1/\epsilon))$ queries.

At every interaction the deformed PCP protocol generates a random number which allows it to simulate the noisy channel in Figure 5. How many bits does this procedure consume throughout the entire network ? Fix a rational deformation parameter $\delta = D/N$ and suppose that random integers in the range $[1 \dots N]$ that are used throughout the network are written in advance on some "*random tape*" which can move only forward. In its turn, each interaction in the network reads one of these numbers (each of which is potentially encoded using $\log_2 N$ bits) and increments the "random head" to a position along the tape where the next integer is stored. This story line clearly shows that the number of "random bits" consumed when simulating the noisy channel depends on the total number of interactions, which in our case is $\mathcal{O}(\log(1/\epsilon))$.

From the above it follows that the total randomness complexity in the machine is $(r(n)+1)\mathcal{O}(\log(1/\epsilon))$ bits. The machine itself may therefore be viewed as a single PCP verifier whose completeness and soundness parameters are those underlying its output, and whose complexity and soundness are those just mentioned. As $\log(1/\epsilon)$ is a common factor for both overall completeness and soundness we may regard it as the number of sequential repetitions of a standard PCP verification. This interpretation of the Hopf–Chernoff configuration is formally expressed as

$$\mathrm{PCP}_{c,c-\epsilon}[r(n),\ q(n)] \overset{\text{Hopf–Chernoff}}{\longrightarrow} \mathrm{PCP}_{\frac{1}{2}+\mathcal{O}(\epsilon),\ \frac{1}{2}-\mathcal{O}(\epsilon)}[r(n)+\mathcal{O}(\log(1/\epsilon)),\ q(n)].$$

Note that this interpretation does not account for the number of bits actually received by a patient verifier. As previously mentioned the patient loses $H_2(\delta)$ bits of information due to the noisy channel induced by an agent. For that reason he literally inspects $q(n) - H_2(\delta) \geq q(n) - 1$ bits from the proof.

5.3. **Perfect completeness (3–bit) PCP networks.** Ideally a PCP verifier would have a perfect completeness and minimal soundness (Moshkovitz, 2010). In this respect an interesting case is that of 3-bit query PCP verifiers where a tradeoff has been shown to exist between non-adaptiveness of the verifier and perfect completeness. An adaptive 3-bit PCP with perfect completeness and soundness above $\frac{1}{2}$ can be constructed (Guruswami, 1998). When the verifier is restricted to be nonadaptive, however, the best soundness parameter of 3-bit PCPs has been conjectured to be $\frac{5}{8}$ (Zwick, 1998). This conjecture implies that getting a $\frac{5}{8}$-approximation for 3-bit satisfiable CSP is NP-hard. The best known result for such a non-adaptive verifier has soundness parameter $\frac{20}{27}+\sigma$ for some arbitrarily small $\sigma > 0$ (Khot, 2006).

Consider the previously best known result assuming non-adaptiveness (Håstad, 1997):

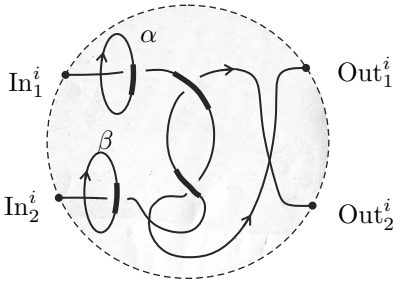
$$\mathrm{PCP}_{1,\frac{6}{8}+\sigma}[\log n,\ 3], \tag{17}$$

and let a deformed version of this verifier underlie any of the interactions in the Hopf–Chernoff configuration. By Theorem 4.1 the Hopf–Chernoff machine produces balanced completeness and soundness parameters in its output. However, by reading this machine differently we can achieve perfect completeness and a soundness parameter near the conjectured bound for a single verifier, specifically $\frac{5}{8}+\frac{1}{24}$.

5.3.1. *The Hopf-Chernoff algorithm.* The following algorithm is derived from the Hopf–Chernoff configuration in Figure 7. It merely describes a realization of such machine and the way its output may be read such that the desired performance is attained. It assumes perfect completeness of the underlying PCP verifier.

(i) Choose input beliefs In^0_j, $j = 1, 2$, arbitrarily from $\{|\mathrm{True}), |\mathrm{False})\}$. Assume $c = 1$ and fix δ.
(ii) Draw a belief α from a Bernoulli distribution with parameter $\frac{1}{2}$. Set $\beta = \neg\alpha$.
(iii) Do the following for $i = 1, \dots, \chi$, where $\chi = \mathcal{O}(1)$:
- Using the underlying PCP verifier, more precisely its deformed version according to the procedure described in Section 5.1, perform four interactions of the Hopf–Chernoff configuration. In detail, execute interactions consecutively for each of the

two verifiers whose beliefs propagate from In_j^i to Out_j^i according to the diagram below.



- Set $\mathrm{In}_j^i = \mathrm{Out}_j^{i-1}$, $j = 1, 2$.

(iv) If the patient belief in the output $\mathrm{Out}_1^\chi = \neg\alpha$ then return |True), otherwise return |False).

Theorem 5.1. *The Hopf–Chernoff algorithm approaches perfect completeness and soundness at most* $\frac{1}{3-2s}$ *as* $\delta \to 1$.

Proof. The proof is deferred to the Appendix. □

Corollary 5.2. *A Hopf–Chernoff algorithm with a non-adaptive* PCP *verifier* (17) *has perfect completeness, and soundness at most* $\frac{2}{3} = \frac{5}{8} + \frac{1}{24}$, *as* $\delta \to 1$.

Proof. This corollary is an application of Theorem 5.1 using the soundness parameter $s = \frac{6}{8}$. □

6. Low-dimensional topology and network equivalence

In (Carmi & Moskovich, 2014a) we defined *tangle machines* to be configurations of interactions such as those used in the definition of class TangIP. Diagrams of tangle machines contain thin and thick lines (thick lines are for agents), where two thin lines can cross in a *virtual crossing*, which is mere combinatorial information. Two tangle machine diagrams are considered *stably equivalent* if they are related by a finite sequence moves (local modifications in a tangle diagram) VR1, VR2, VR3, SV, R1, R2, R3, UC, and ST in Figure 9. We further add the move FS (*false stabilization*) to this collection of local moves, because in this note we do not distinguish between different interactions. In this note we consider only equivalence under *all* moves in Figure 9, and for simplicity we call the induced equivalence relation simply *equivalence*.

Remark 6.1. In a more general setting in which different interactions may represent different protocols, and maybe even represent proof systems with different soundness and completeness parameters, we would omit FS. But in this note FS is included. As we prove in (Carmi & Moskovich, 2014b), the resulting equivalence classes are in bijective correspondence with *w–tangles* as in (Bar-Natan & Dancso, 2013).

Two machines related by the local moves in Figure 9 can be entirely recovered from one another, and any verification performed by one can be simulated by the other. For details, see (Carmi & Moskovich, 2014a). These moves are inspired by low-dimensional topology, as discussed in (Carmi & Moskovich, 2014a,b). In the rest of this section, we discuss the interpretation of R2 and of R3 (*Reidemeister Moves*) in terms of interactive proofs. All of the other local moves are trivial artifacts of our diagrammatic formalism, and do not appear to have philosophical content in the context of interactive proofs.

An patient which does not emerge from any interaction is said to be *initial*, and one that does not feed into any interaction is said to be *terminal*. Verifier states which are neither initial nor terminal are said to be *intermediate*.

That M and M' are equivalent, written $M' \sim M$, means that they share some similarities. For example they have identical initial and terminal beliefs which entails that both machines have the same computational power in terms of deciding a language. Nevertheless, their local behaviour may be different. In other words, the outcome of intermediate interactions in between the same initial and terminal statistics may be different in equivalent machines. This was one of the earlier examples wherein a machine and its equivalent counterpart represented two different prover strategies arriving at the same proof (see Figure 4). We may think that:

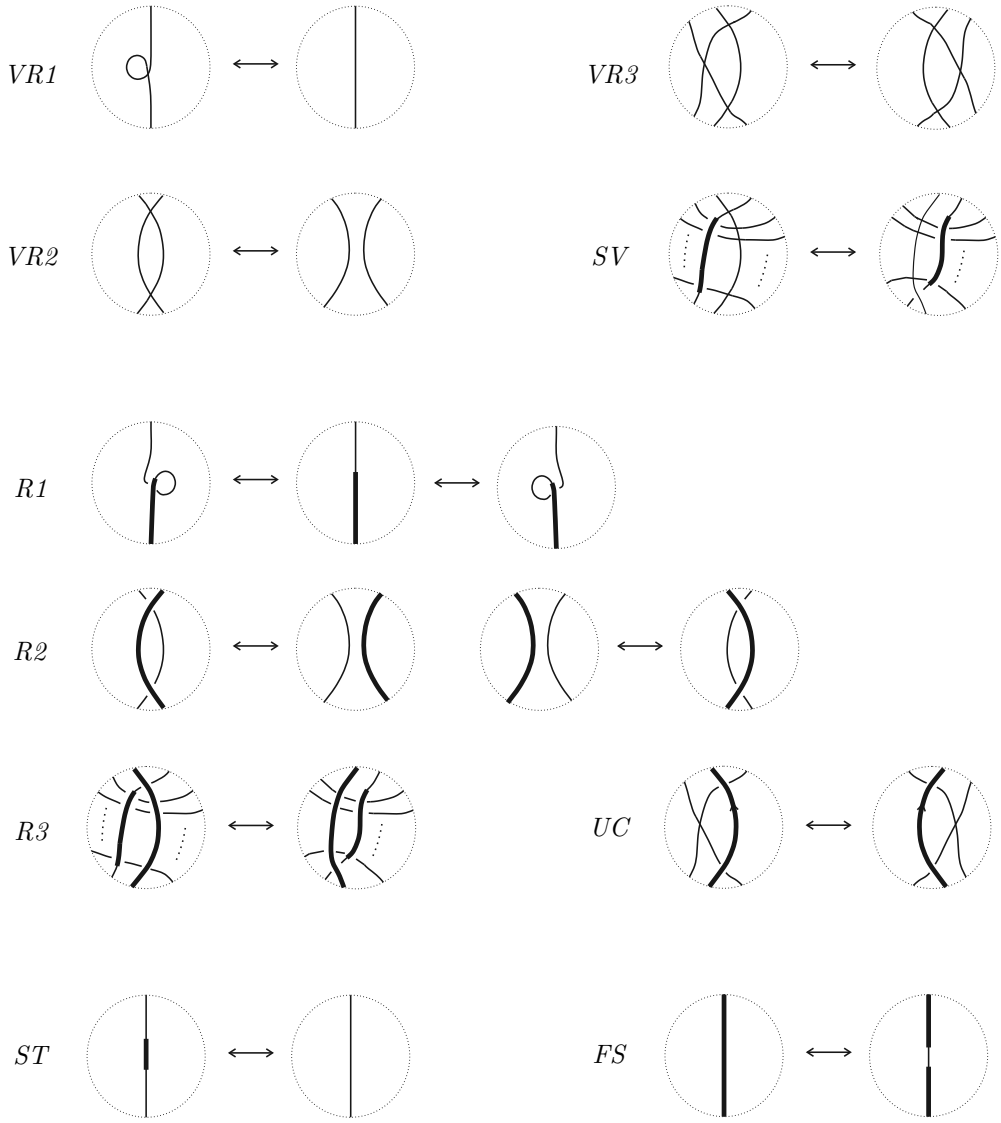
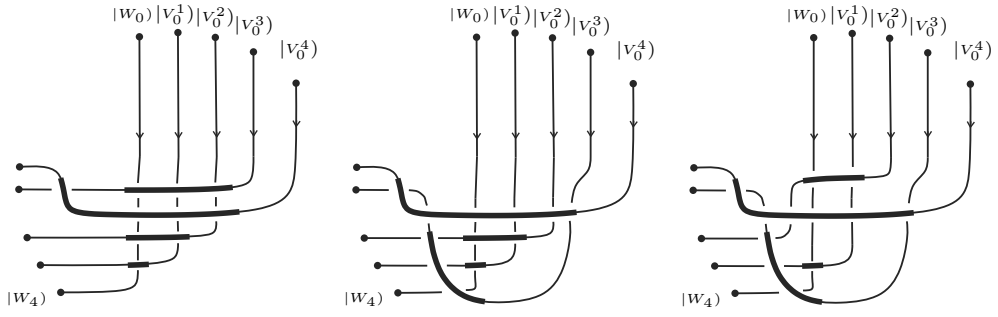


Figure 9. Local moves for machines, valid for any orientations of the strands. The FS move is valid when two interactions with the same parameters are considered the same (*e.g.* in this note), but in contexts where different interactions should be distinguished, we drop the FS move.

Equivalent tangle machines represent different ways of proving a theorem.

As an example, here are a few equivalent prover strategies for the machine in Figure 6 all which are obtained by application of R2 and R3 moves.



Topology, in particular the Reidemeister moves, dictate the spread of knowledge throughout the machine. This brings to mind a familiar concept.

6.1. **Zero-Knowledge tangle machines.** Here is an interesting question concerning the above. Does a machine M have an equivalent counterpart $M' \sim M$ wherein the proof emerges in one or more of its intermediate verifier states? What that means is that the proof could potentially be obtained by a submachine of M' which is a machine smaller than M. As M may be fairly complicated, there is no guarantee that this question is easy to answer in general.

If within M hides a smaller machine which can prove the same, then it means that M contains some redundancy. This 'redundancy' is in fact a set of interactions which dictate the process of getting the proof but which contribute no other knowledge apart from this. If someone where to give us the right combination of Reidemeister moves we could potentially extract a smaller machine that contains the same knowledge in terms of proving the theorem but which does not contain any of the redundant interactions. This concept is the machine analog of zero-knowledge proof. The precise definition is the following.

Definition 6.2 (Zero-Knowledge tangle machine)**.** *A tangle machine M is said to be* zero-knowledge *if the following is satisfied.*

(i) There are no intermediate interactions in M that decides L.
(ii) There exists a machine $M' \sim M$ which decides L at one of its intermediate interactions.

The notion of zero-knowledge proof (Goldwasser, 1989) essentially restricts the knowledge that may be gained by an arbitrary verifier in a course of interacting with a prescribed prover on some common input (x). The precise definition makes use of a simulator which is an arbitrary feasible algorithm that is able to reproduce the transcript of such an interaction without ever interacting with the prover. A good tutorial on the subject including a wealth of interesting discussions is (Goldreich, 2010).

In the following paragraphs we would show an analog of this concept from the perspective of tangle machines.

6.2. **R3: Knowledge extraction.** Consider the machines $M' \sim M$ in Figure 10. Both have the same initial and terminal belief statistics. The explicit structure of the machines is mostly irrelevant except that they both contain a submachine S with an arbitrary internal structure which is represented by a blank disk. In M, the verifier Z is an agent to all initial states of S and the resulting beliefs from this interaction are characterized by $|X_i)$, $i = 1, 2, \ldots,$. In M' the same verifier is an agent to the terminal states of S and the resulting beliefs from this interaction are characterized by $|Y_i)$. The machine M decides L in its terminal statistics, and by machine equivalence so does M'. Furthermore, some of S's terminal statistics, $|Y_i)$, also decide L.

That M is zero-knowledge implies the proof should not appear somewhere within it. Assuming none of the $\mathrm{In}'s$ decide L, and neither do any intermediate belief states of S, this requirement implies that none of the $|X_i) = (\mathrm{In}_i)^{|Z)}$ decide L. On the other hand, the machine M' shows us that an interaction between the terminal states of S, some of which decide L, with the agent Z are able to produce the proof. That is, some of $\mathrm{Out}_i = |Y_i)^{|Z)}$ decide L. These requirements completely characterize the belief distribution $|Z)$. Perhaps unsurprisingly, it turns out that $|Z) = \frac{1}{2}|\mathrm{True}) + \frac{1}{2}|\mathrm{False})$.

Ideally we would require that S reproduces the proof as if it was produced by M itself. This requirement translates into

$$|Y_i) = \mathrm{Out}_i, \tag{18}$$

for any Out_i that decides L. As either sides of (18) normally depend on the deformation parameter δ it may be used to find δ such that M is zero-knowledge. Below we give an example of this.

6.2.1. *Example.* Consider the two machines in Figure 11. Assume they both employ a deformed IP system whose completeness and soundness are δ and $\frac{1}{2}\delta$. Let $|Z) = \frac{1}{2}|\mathrm{False}) + \frac{1}{2}|\mathrm{True})$ and let us first see what the value of δ is for the machine to decide L. The output $|X_2)$ of either machines is given by

$$|X_2) \longrightarrow \begin{cases} \left[(1-\delta)^2 + \frac{1}{2}\delta\right]|\mathrm{False}) + \left[\delta(1-\delta) + \frac{1}{2}\delta\right]|\mathrm{True}), & x \in L; \\ \left[(1-\frac{1}{2}\delta)^2 + \frac{1}{4}\delta\right]|\mathrm{False}) + \left[\frac{1}{2}\delta(1-\frac{1}{2}\delta) + \frac{1}{4}\delta\right]|\mathrm{True}), & x \notin L. \end{cases} \tag{19}$$

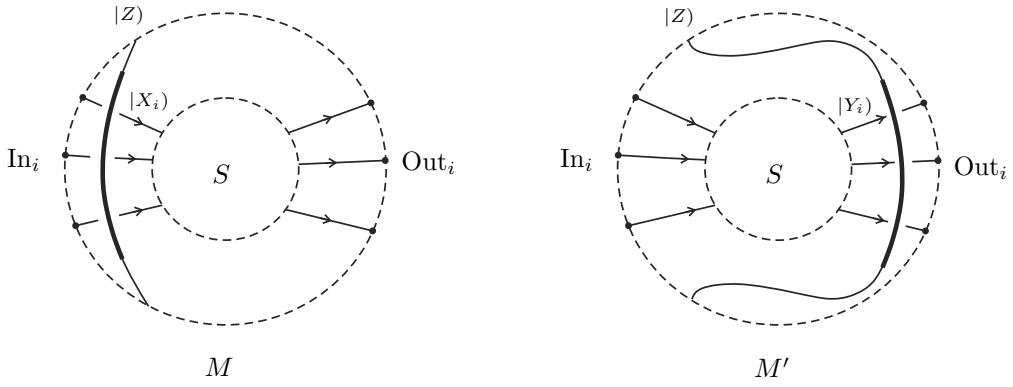


Figure 10. Two equivalent machines. The machine M is zero-knowledge. The proper submachine S determines L.

from which we conclude that $\delta > \frac{1}{2}$. The machine on the right in this figure is zero-knowledge because $|X_1)$ does not decide L:

$$|X_1) \longrightarrow \begin{cases} \left[1-\frac{1}{2}\delta\right]|\text{False}) + \frac{1}{2}\delta\,|\text{True})\,, & x \in L; \\ \left[1-\frac{1}{4}\delta\right]|\text{False}) + \frac{1}{4}\delta\,|\text{True})\,, & x \notin L. \end{cases} \tag{20}$$

and on the other hand the submachine inside the small disk on the left decides L:

$$\left|\bar{X}_1\right) \longrightarrow \begin{cases} [1-\delta]\,|\text{False}) + \delta\,|\text{True})\,, & x \in L; \\ \left[1-\frac{1}{2}\delta\right]|\text{False}) + \frac{1}{2}\delta\,|\text{True})\,, & x \notin L. \end{cases} \tag{21}$$

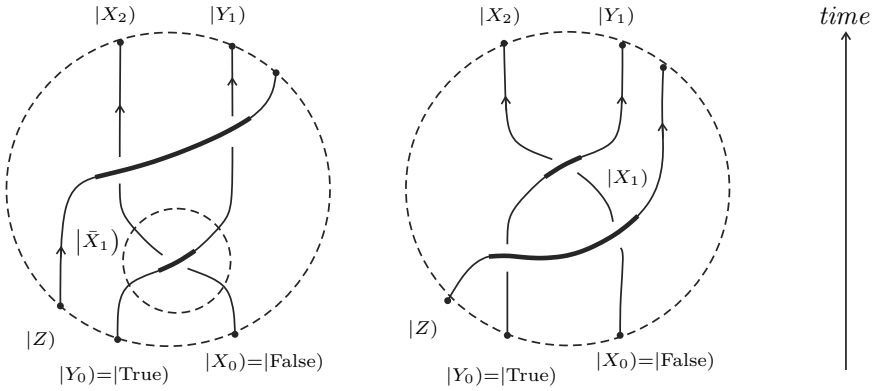


Figure 11. Example of a zero-knowledge machine.

If we further restrict the value of δ so as to satisfy (18), namely,

$$\left|\bar{X}_1\right) = |X)_2 \longrightarrow \delta = \delta(1-\delta) + \frac{1}{2}\delta, \tag{22}$$

we obtain $\delta = \frac{1}{2}$ which obviously contradicts the basic requirement of deciding L. If we slightly relax this condition to allow a small discrepancy between the underlying distributions then we may take $\delta = \frac{1}{2} + \kappa(x)$ where $\kappa(x)$ is a statistical distance which potentially depends on x.

7. Some open issues

We list a number of standing problems that arise from this work.

Elegant approaches to deform an IP system. In Sections 3 and 5.1 we have shown how a deformed system may be reconstructed using a regular IP system via the process of deformation. Deformation can be seen as a way to introduce uncertainty into an interaction between a prover and a verifier. We use an additional verifier, an agent, in order to establish a one-way noisy communication between the prover and the (patient) verifier. One could think of more natural ways to introduce uncertainty into such an interaction. Perhaps this could be accomplished in the

level of the proof itself. To illustrate this concept, imagine a PCP verifier with access to a proof encoding which has been fabricated by an agent verifier. The proof encoding may potentially be contaminated by noise, or maybe fragments from the original proof are encoded alongside pieces of irrelevant material. We suspect that a proper encoding strategy and a respective decoding (patient) verifier could reproduce the effect of deformation.

Expressive power of networks. We are unsure whether MIP $\subseteq$ TangIP. An affirmative answer would suggest that the effect of coordinated provers such as those underlying MIP can be reproduced by multiple verifiers and a single prover.

Low-dimensional topology. To what extent can low-dimensional topology assist in designing networked proof strategies? We have seen that equivalent networks have the same expressive power yet they differ in their intermediate interactions. They may be viewed as different ways to prove the same things. What are the factors controlling the existence of zero-knowledge configurations?

Appendix A. Proof of Theorem 4.1

Let us begin by writing down the relations between the outputs Out_j and inputs In_j of this machine. Note that

$$\mathrm{Out}_1 = \left(\mathrm{In}_1^{|\alpha)}\right)^{\left(\mathrm{In}_2^{|\beta)}\right)}, \quad \mathrm{Out}_2 = \left(\mathrm{In}_2^{|\beta)}\right)^{\left(\mathrm{In}_1^{|\alpha)}\right)}. \tag{23}$$

Explicitly writing (23) using the formal parameter h yields

$$\begin{bmatrix}\mathrm{Out}_1^i\\ \mathrm{Out}_2^i\end{bmatrix} = \underbrace{\begin{bmatrix}(1-h)^2 & h(1-h)\\ h(1-h) & (1-h)^2\end{bmatrix}}_{\stackrel{\mathrm{def}}{=}A(h)} \begin{bmatrix}\mathrm{In}_1^i\\ \mathrm{In}_2^i\end{bmatrix} + \underbrace{\begin{bmatrix}h(1-h) & h^2\\ h^2 & h(1-h)\end{bmatrix}}_{\stackrel{\mathrm{def}}{=}B(h)} \begin{bmatrix}|\alpha)\\ |\beta)\end{bmatrix}. \tag{24}$$

Letting $\mathrm{In}_j^{i+1} = \mathrm{Out}_j^i$, $j = 1, 2$, equation (24) underlies a linear dynamical system. It is easy to verify that the eigenvalues of the transition matrix $A(h)$ all are within the unit circle, i.e. $|\lambda(A(h))| < 1$ for any $h > 0$. That means that the system (24) reaches a steady-state as $i \to \infty$. The steady-state can be obtained as follows. Rewrite (24) as

$$\begin{bmatrix}\mathrm{Out}_1\\ \mathrm{Out}_2\end{bmatrix} = A(h) \begin{bmatrix}\mathrm{Out}_1\\ \mathrm{Out}_2\end{bmatrix} + B(h) \begin{bmatrix}|\alpha)\\ |\beta)\end{bmatrix}, \tag{25}$$

and solve for Out_1 and Out_2. Thus,

$$\begin{bmatrix}\mathrm{Out}_1\\ \mathrm{Out}_2\end{bmatrix} = (I - A(h))^{-1} B(h) \begin{bmatrix}|\alpha)\\ |\beta)\end{bmatrix} = \frac{1}{3-2h} \begin{bmatrix}2(1-h)\,|\alpha) + |\beta)\\ |\alpha) + 2(1-h)\,|\beta)\end{bmatrix}. \tag{26}$$

Define

$$\begin{aligned} |\alpha) &\stackrel{\mathrm{def}}{=} a\,|\mathrm{True}) + (1-a)\,|\mathrm{False})\,;\\ |\beta) &\stackrel{\mathrm{def}}{=} b\,|\mathrm{True}) + (1-b)\,|\mathrm{False})\,. \end{aligned} \tag{27}$$

For the network to decide L we require the steady-state of Out_1 to satisfy

$$\mathrm{Out}_1 \longrightarrow \begin{cases} (\frac{1}{2}+\sigma)\,|\mathrm{True}) + (\frac{1}{2}-\sigma)\,|\mathrm{False})\,, & x \in L;\\ (\frac{1}{2}-\sigma)\,|\mathrm{True}) + (\frac{1}{2}+\sigma)\,|\mathrm{False})\,, & x \notin L. \end{cases} \tag{28}$$

for some $\sigma > 0$. Using both (26) and (27) this requirement translates into the following set of equations

$$\begin{aligned} (3-2c\delta)\left(\tfrac{1}{2}+\sigma\right) &= 2(1-c\delta)a + b, && x \in L;\\ (3-2(c-\epsilon)\delta)\left(\tfrac{1}{2}-\sigma\right) &= 2(1-(c-\epsilon)\delta)a + b, && x \notin L. \end{aligned} \tag{29}$$

where the fact that $h = c\delta$ for $x \in L$ and $h = (c-\epsilon)\delta$ for $x \notin L$ has been used. Solving (29) for the coefficients a and b while assuming $\sigma = \frac{1}{12}\epsilon\delta$ yields (15). The underlying output probabilities in (16) are given by (28).

To complete the argument we need to show that the network converges within the stated number of iterations. It is sufficient to consider the case where the output probabilities (28)

are attained to within the order $\mathcal{O}(\sigma) = \mathcal{O}(\epsilon)$. Growth rate of the system (24) is linear in $|\lambda_1(A(h))|^\chi$ where $\lambda_1(A(h))$ denotes the largest eigenvalue of $A(h)$. Simple calculation shows that $\lambda_1(A(h)) = 1 - h$ which yields $\chi = \mathcal{O}(\log(1/\epsilon))$.

Appendix B. Proof of Theorem 5.1

Completeness. Assume $x \in L$ and note that as $\delta \to 1$, so does $h = c\delta \to 1$. In the limit where $h = 1$ note that by (13) the Hopf–Chernoff swaps the beliefs α and β such that always $\mathrm{Out}_1 = \beta$ and $\mathrm{Out}_2 = \alpha$. As $\beta = \neg\alpha$, step (iv) of the algorithm concludes with |True).

Soundness. The soundness of the algorithm bounds the probability that $\mathrm{Out}_1 = \neg\alpha$ in case where $x \notin L$. This probability is given by

$$\begin{aligned}\Pr(\mathrm{Out}_1 = \neg\alpha) = \Pr(\mathrm{Out}_1 = |\mathrm{True}) \mid \alpha = |\mathrm{False})) \Pr(\alpha = |\mathrm{False})) \\ + \Pr(\mathrm{Out}_1 = |\mathrm{False}) \mid \alpha = |\mathrm{True})) \Pr(\alpha = |\mathrm{True})).\end{aligned} \tag{30}$$

Although not truly essential, the algorithm assumes $\Pr(\alpha = |\mathrm{True})) = \frac{1}{2}$. The conditional probabilities above can be bounded using (13) as follows. Take $h = s\delta \to s$ and assume that $|\alpha) = |\mathrm{False})$, $|\beta) = |\mathrm{True})$. In this case (13) implies

$$\Pr(\mathrm{Out}_1 = |\mathrm{True}) \mid \alpha = |\mathrm{False})) \leq \frac{1}{3 - 2s}. \tag{31}$$

On the other hand, letting $|\alpha) = |\mathrm{True})$, $|\beta) = |\mathrm{False})$, the same equation reads

$$\Pr(\mathrm{Out}_1 = |\mathrm{False}) \mid \alpha = |\mathrm{True})) \leq \frac{1}{3 - 2s}. \tag{32}$$

This follows from the fact that the algorithm decides $x \in L$ if $\mathrm{Out}_1 = \neg\alpha$ irrespective of the beliefs themselves. For that reason both these equations are merely the same one, though with different values for α and Out_1. Both describing a failure of the algorithm to decide $x \notin L$. The theorem now follows from (30), (31) and (32).

References

Arora, S., & Barak, B. 2009 Computational complexity: A modern approach. Cambridge University Press.

Arora, S. & Safra, S. 1998 Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM 45(1)*, 70-122.

Bar-Natan, D., & Dancso, S. 2013 Finite type invariants of w-knotted objects: From Alexander to Kashiwara and Vergne. `arXiv:1309.7155`

Ben-Or, M., & Goldwasser, S., & Kilian, J. & Wigderson, A. 1988 Multi prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the 20th ACM Symposium on Theory of Computing*, 113–121.

Buliga, M. 2011 Computing with space: A tangle formalism for chora and difference. `arXiv:1103.6007`

Buliga, M., & Kauffman, L. 2013 GLC actors, artificial chemical connectomes, topological issues and knots. `arXiv:1312.4333`

Carmi, A.Y., & Moskovich, D. 2014 Tangle machines I: Concept. Unpublished. `http://arxiv.org/abs/1404.2862`

Carmi, A.Y., & Moskovich, D. 2014 Tangle machines II: Invariants. Unpublished. `http://arxiv.org/abs/1404.2863`

Goldreich, O. 2010 A short tutorial of zero-knowledge. Unpublished. `http://www.wisdom.weizmann.ac.il/~oded/zk-tut02.html`

Goldwasser, S., & Micali, S., & Rackoff, C. 1989 The knowledge complexity of interactive proof systems. *SIAM Journal on Computing 18(1)*, 186–208.

Guruswami, V., & Lewin, D., & Trevisan, L. 1998 A tight characterization of NP with 3 query PCPs. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*.

Håstad, J. 1997 Some optimal inapproximability results. *Journal of the ACM 48(4)*, 798–859.

Kauffman, L.H. 1994 Knot automata. In *Twenty-Fourth International Symposium on Multiple-Valued Logic, Conference Proceedings*, 328–333.

Khot, S., & Saket, R. 2006 A 3-query non-adaptive PCP with perfect completeness. In *Proceedings of the 21st IEEE Conference on Computational Complexity*, 159–169.

Meredith, L.G., & Snyder, D.F. 2010 Knots as processes: A new kind of invariant. `http://arxiv.org/abs/1009.2107`

Moshkovitz, D., & Raz, R. 2010 Two-query PCP with subconstant error. *Journal of the ACM 57(5)*.

Surowiecki, J. 2005 The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business, economies, societies, and nations. Random House LLC.

Vicary, J. 2012 Higher quantum theory. `arXiv:1207.4563`

Zwick, U. 1998 Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 201–210.

Division of Mathematical Sciences, Nanyang Technological University, 21 Nanyang Link Singapore 637371

E-mail address: `dmoskovich@ntu.edu.sg`

Faculty of Engineering Sciences, Ben-Gurion University of the Negev, Israel

E-mail address: `avishycarmi@gmail.com`