On the interactive complexity of graph reliability

Jean-Marc COUVEIGNES * Juan Francisco DIAZ-FRIAS †
Michel de ROUGEMONT † Miklos SANTHA ‡

Abstract

We give an interactive protocol for $s-t$ RELIABILITY, the well known reliability problem on graphs. Our protocol shows that if $IP(f(n))$ denotes the class of languages whose interactive complexity is $O(f(n))$, that is the set of languages which can be accepted by an interactive proof system with $O(f(n))$ number of rounds, then $s-t$ RELIABILITY $\in IP(n)$. This complexity is significantly smaller than what one could get via reduction to QBF, the standard $PSPACE$-complete language. Another interesting aspect of our protocol is that it includes a general method to deal with rational numbers in interactive proof systems.

1 Introduction

The notion of an *interactive proof system* or *interactive protocol* was introduced by Goldwasser, Micali and Rackoff [6] and independently by Babai [1]. Intuitively, it is a way by which an infinitely powerful prover can convince using interaction a polynomially powerful probabilistic verifier about the membership of elements in some language, but only for those elements, which are indeed in the language. More formally, a language L belongs to the class IP, if there is a probabilistic polynomial time verifier $\mathcal{V}$, and a prover $\mathcal{P}$ such that for every $x \in L$, $\mathcal{P}$ can convince $\mathcal{V}$ to accept x with overwhelming probability, but for every $x \notin L$, no prover $\mathcal{P}'$ can convince $\mathcal{V}$ to accept x with more than negligible probability.

For several years the exact computational power of the class IP was an open problem. Then in 1990 two breakthrough papers gave a precise answer to this question. First Lund, Fortnow, Karloff and Nisan [7] have proved that every language in the class $\#P$ has an interactive proof system. Soon afterwards, Shamir [8] extended their technique to prove that IP coincides with the complexity class $PSPACE$.

One of the important parameters of an interactive proof system is the number of rounds, that is the number of messages exchanged between the prover and the verifier. Babai and Moran have proven [3] that the number of rounds in an unbounded interactive proof system can be reduced by a constant factor. Babai [1] has shown that bounded round interactive proof systems can be simulated by a protocol in just two round, where the first

*ENS, LIENS, 45, rue d'Ulm, 75005 Paris, France.
†CNRS, URA 410, Université Paris-Sud, LRI, 91405 Orsay, and Ecole Nationale Supérieure de Techniques Avancées, France. Research funded by DRET (Direction des Recherches et Etudes Techniques), Contracts No 89/1061 and 90/1538.
‡CNRS, URA 410, Université Paris-Sud, LRI, 91405 Orsay, France. Research supported by the ESPRIT Working Group 7097: RAND.

message is sent by the verifier. It is highly unlikely that it is possible to obtain a bounded round protocol for $\#P$-complete functions, since the existence of such a protocol would imply by a result of Boppana, Hastad and Zachos [4] that the polynomial time hierarchy collapses. Indeed, the first example of a proof system with an unbounded number of rounds was given for a $\#P$-complete problem in [7].

For a function $f(n)$, let us denote by $IP(f(n))$ the set of languages which can be accepted by an interactive proof system with $O(f(n))$ number of rounds. For languages in $IP(f(n))$, we will say that their *interactive complexity* is of $O(f(n))$. Lund, Fortnow, Karloff and Nisan actually have shown that PERMANENT $\in IP(n)$, where PERMANENT is the problem of computing the permanent of an $n \times n$ matrix. The result of Shamir can be restated as QBF $\in IP(n^2)$, where QBF is the $PSPACE$-complete language of quantified boolean formulas on n variables. ¿From the result of Shamir it is easy to deduce that #SAT $\in IP(n)$, where #SAT is the $\#P$-complete problem of computing the number of satisfying assignments for a CNF formula. This result was also obtained by Babai and Fortnow [2].

In this paper we will give an interactive protocol for the following graph reliability problem.

- $s - t$ **RELIABILITY**

  Input: An undirected graph $G = (V, E)$ with n vertices; $s, t \in V$; and for every edge e, a rational number $r(e)$ representing the probability that the edge e exists (does not fail).

  Output: The probability that there is a path from s to t consisting exclusively of edges that have not failed.

Our protocol will have a linear number of rounds as a function of the number of vertices, that is we will prove the following result about its interactive complexity:

Main Theorem: $s - t$ RELIABILITY $\in IP(n)$.

It is well known [9] that $s - t$ RELIABILITY is in the class $P^{\#P}$. Since $P^{\#P}$ is included in $PSPACE$, Shamir's result implies that there exists an interactive protocol for $s - t$ RELIABILITY. Nonetheless we think that the construction of a direct protocol for this problem is important for the following reasons:

1. The problem $s - t$ RELIABILITY is hard for $\#P$, but it is not in that class [9]. Also, it is not believed to be $PSPACE$-complete. As far as we know, the $\#P$-hard problems for which protocols were designed were either $\#P$-complete or $PSPACE$-complete. Therefore we believe that our protocol is interesting on its own.

2. The protocol establishes a far better bound on the number of rounds than one could get by using the generic reduction to QBF. Let us suppose that $s-t$ RELIABILITY can be solved in space $O(n)$ by a deterministic Turing Machine M (we don't know of any algorithm which would use less space than that). By the generic reduction to QBF, it is possible to associate in polynomial time to an input graph G with n vertices and a rational number q a boolean formula $\phi_{G,q}$ on $O(n^2)$ variables such that the probability of G being $s - t$ connected is q if and only if $\phi_{G,q}$ is in QBF.

By the result of Shamir, QBF on $O(n^2)$ variables is in $IP(n^4)$. Therefore our direct protocol establishes a far better bound for the interactive complexity.

Another possibility to obtain an efficient interactive protocol would be to consider direct reduction to QBF. Nonetheless such a reductions seems to be technically quite complicated.

3. A novel aspect of our protocol is that we have to deal with rational numbers since an instance of $s - t$ RELIABILITY involves probability values. In all the known interactive protocols computations are done over the integers modulo p where p is some well chosen prime number. It is not at all obvious how rational numbers can be treated in such a finite field. We solve this problem by defining a homomorphism from an appropriately chosen subring of the rationals to the integers modulo p. We believe that this technique is new and that in can also be useful in other context as well.

4. Finally an IP protocol for the $s - t$ RELIABILITY problem has the following practical application : consider a graph with uncertainty defined by the function $r : E \rightarrow \{0, 1\}$, and some external agent who claims he can traverse the graph with probability greater than q, for some constant $q > 0$. How can we check he is right? Just consider him as a prover, and run the protocol. This idea is used in [5] to test the robustness of planning in robotics and obtain an efficient method to compare strategies controlling mobile robots. In general it is a method to study problems with uncertainty, and test the realizability of various hypotheses.

The $s - t$ RELIABILITY problem is closely related to the following graph enumeration problem:

- $s - t$ **CONNECTEDNESS**

  Input: An undirected graph $G = (V, E)$ with n vertices; $s, t \in V$.

  Output: $\text{sub}(G, s, t)$, the number of subgraphs of G in which there is a path from s to t.

Indeed, the connectivity problem is a special case of the reliability problem. If all the probabilities are $1/2$ then the number of connected subgraphs is just the probability of the graph being $s - t$ connected multiplied by $2^{\binom{n}{2}}$. Nonetheless a generalization of the enumeration problem to multi-graphs, that is graphs where the edges have integer weights, already captures all the combinatorial difficulty an interactive protocol has to address in the case of the reliability problem. Therefore we will first give an interactive protocol for this generalized enumeration problem which we call MULTI $s - t$ CONNECTEDNESS. Then we will define INTEGER $s - t$ RELIABILITY, a version of the reliability problem for multi-graphs. In opposition to that, we will call PROBABILISTIC $s - t$ RELIABILITY the original reliability problem. It will be easy to modify the protocol for MULTI $s - t$ CONNECTEDNESS to get a protocol for INTEGER $s - t$ RELIABILITY. Finally we will make a reduction from PROBABILISTIC $s - t$ RELIABILITY to INTEGER $s - t$ RELIABILITY so that the protocol for the later can be used for the former.

The rest of the paper is organized as follows: In Section 2 we describe the structure of our protocols. We also give here some basic definitions about multi-graphs. Section

3 contains the protocol for MULTI $s-t$ CONNECTEDNESS. Section 4.1 explain the construction of INTEGER $s-t$ RELIABILITY from PROBABILISTIC $s-t$ RELIABILITY. Section 4.2 has the protocol for INTEGER $s-t$ RELIABILITY, and Section 4.3 shows how to reduce PROBABILISTIC $s-t$ RELIABILITY to it.

2 The structure of the protocols

Throughout the paper every integer computation will be done in the field $\mathbf{Z}_p$ whose base set is $\{0,1,\ldots,p-1\}$, where p is an appropriately chosen prime number of exponential value in the size of the problem (the size of p is therefore polynomial). This prime can be given to the verifier by the prover who can also certify it in polynomial time with high probability. We denote by $\mathbf{Z}_p[x]$ the ring of univariate polynomials over $\mathbf{Z}_p$. The rational numbers will be denoted by $\mathbf{Q}$, and the natural numbers by $\mathbf{N}$.

The overall structure of our protocols is the same as the structure of all existing protocols for other $\#P$-complete problems, such as PERMANENT or #SAT. One way to look at it is to say that the verifier maintains during the protocol a list of statements, each about an instance of the problem. Initially this list contains the prover's original claim. At the end, the list contains a statement about a small instance which can be checked directly by the verifier. The protocol guarantees that if the original claim is true then the last statement will also be true. However, if the original claim was false, then whatever is done by the prover, the verifier will end up with overwhelming probability also with a false statement.

In the verifier's handling of its list we will distinguish two kinds of steps. When the list contains just one statement, then in an *extension* step $\mathcal{V}$ will create several instances. These instances will have the same size, and in some combinatorial sense, they will be smaller than the instance of the single statement. Then $\mathcal{V}$ asks the prover to give him the solution for all these new instances. $\mathcal{V}$ runs a simple consistency test on the original statement and the new ones, and if the test is passed, he replaces the original statement by the list of the answers of the verifier. Every extension step will satisfy, that whenever the single statement on the list of the verifier is false, one of the new statements of the prover will also be false.

When the list of $\mathcal{V}$ contains several statements, then first he creates in a *reduction* step an instance of the problem in which some numerical data are replaced by low degree polynomials in the same variable. We call this instance a *polynomial instance*, and the process of creating it *mixing.* The polynomial instance will encode the numerical instances on the list of the verifier. Its size will be the same as the size of the instances in the list, and it will not be harder in the particular combinatorial sense than the numerical instances. The solution for the polynomial instance will be a low degree polynomial, and $\mathcal{V}$ will ask $\mathcal{P}$ to give him the coefficients of this polynomial. Then he checks if the polynomial given by the prover does indeed encode the original instances. If this is the case, he creates a new numerical instance by replacing the variable in the polynomial instance by a randomly generated integer. He also computes the value of the polynomial at this integer point. His new list will consist of the single statement claiming that the solution of the new numerical instance is this value. For a reduction step it will be true, that if at least one of the original statements was false, then with overwhelming probability the new statement will also be false.

The idea of mixing several numerical instances into one polynomial instance whose solution is a low degree polynomial was probably the main ingredient which led to interactive protocols for $\#P$-complete and $PSPACE$-complete problems. The reduction step and its probabilistic analysis are now well known and standard procedures. In this aspect our protocols will be standard, and therefore we will not elaborate in detail on this part. On the other hand, the extension step requires new ideas for graph problems, and in the description of our protocols we will put the emphasis on it.

We now give some definitions which will be needed in the protocols. First we generalize the notion of graphs to multi-graphs, which will be graphs with weights on the edges. In polynomial multi-graphs the weights will be polynomials. Then we also extend the definition of a path to this new concept. Finally we define the necessary operations for the mixing of multi-graphs.

Definition: A *multi-graph* (*m-graph*) $\mathcal{G}$ is a pair (V, l). The finite set V is the set of *vertices.* The subsets of cardinality two of the vertices are called *edge slots,* the set of edge slots will be denoted by $V^{(2)}$. The *multiplicity function* $l : V^{(2)} \to \mathbf{Z}_p$ defines for each edge slot $\{u, v\}$, its multiplicity. The set $E = \{e \in V^{(2)} : l(e) \neq 0\}$ is the set of *edges* of $\mathcal{G}$. We will write $\mathcal{G} = (V, E, l)$ rather than $\mathcal{G} = (V, l)$, when we want to refer explicitly to the set of edges of $\mathcal{G}$. In a *polynomial m-graph* the multiplicity function maps the edge slots into $\mathbf{Z}_p[x]$. The set of edges in this case consists of those edge slots whose multiplicity is not the zero polynomial.

Definition: An $s - t$ *multi-path* ($s - t$ *m-path*) in an m-graph or polynomial m-graph $\mathcal{G} = (V, E, l)$ is a sequence of vertices $p = < v_0, v_1, \ldots, v_{k-1}, v_k >$ for some integer k such that $s = v_0$, $t = v_k$, all the vertices in the sequence are distinct, and for every $1 \leq i \leq k$, we have $\{v_{i-1}, v_i\} \in E$.

Definition: Let $\mathcal{G} = (V, l)$ be an m-graph, and let $g(x)$ be a polynomial over $\mathbf{Z}_p$. Then $g(x)\mathcal{G} = (V, l_g)$ is a polynomial m-graph where $l_g(e)$ is the polynomial $l(e)g(x)$ for every edge slot e. Let $\mathcal{G} = (V, l)$ and $\mathcal{H} = (V, l')$ be m-graphs, or polynomial m-graphs. The sum of $\mathcal{G}$ and $\mathcal{H}$ is $\mathcal{G} + \mathcal{H} = (V, l + l')$, where $(l + l')(e) = l(e) + l'(e)$ for every edge slot e.

Every graph $G = (V, E)$ can be conceived as an m-graph $\mathcal{G} = (V, E, l)$, where the edges have unit multiplicity. Also, for technical reasons we will associate to every m-graph or polynomial m-graph a graph whose edges are the edges of the m-graph. The operations for converting a graph to an m-graph and vice versa are formally:

Definition: Given an m-graph or a polynomial m-graph $\mathcal{G} = (V, E, l)$, the *associated graph* $\mathcal{G}^*$ is defined as $G = (V, E)$. Given a graph $G = (V, E)$, the *associated m-graph* G^* is the m-graph (V, l) where for every edge slot $\{u, v\}$, $l(\{u, v\}) = 1$ if $\{u, v\} \in E$, and 0 otherwise.

3 A protocol for MULTI $s - t$ CONNECTEDNESS

Definition: The m-graph $\mathcal{G}' = (V, l')$ is a *multi-subgraph* (*m-subgraph*) of the m-graph $\mathcal{G} = (V, l)$ if for every edge slot e, we have $l'(e) \leq l(e)$.

The generalized connectedness problem for which we will give first an interactive protocol is the following:

- **MULTI $s-t$ CONNECTEDNESS**

  Input: $(\mathcal{G}, s, t)$, where $\mathcal{G} = (V, l)$ is a multi-graph with n vertices, and $s, t \in V$.

  Output: $\mathrm{sub}(\mathcal{G}, s, t)$, the number of multi-subgraphs of $\mathcal{G}$ which contain a multi-path from s to t.

Clearly, $\mathrm{sub}(G, s, t) = \mathrm{sub}(G^*, s, t)$ for every graph G, therefore we have indeed generalized the original connectedness problem. Before explaining our protocol we state a proposition about the number of $s-t$ connected m-subgraphs, which gives a way of defining the number of polynomial m-subgraphs in a polynomial m-graph.
Definition: Let $G = (V, E)$ be a graph, s and t two vertices, and l a multiplicity function from $V^{(2)}$ to $\mathbf{Z}_p$ or $\mathbf{Z}_p[x]$. We define the *value* of G with respect to l as $\mathrm{val}(G, l) = \prod_{e \in E} l(e)$. Also, let the function $\mathrm{con}(G, s, t)$ be 1 if there is a $s-t$ path in G, and 0 otherwise.

Proposition 1 *Let $\mathcal{G} = (V, l)$ be an m-graph, s and t two vertices. Then*

$$\mathrm{sub}(\mathcal{G}, s, t) = \sum_{G' \text{ is a subgraph of } \mathcal{G}^*} \mathrm{con}(G', s, t)\mathrm{val}(G', l).$$

Proof: Let G' be a subgraph of $\mathcal{G}^*$. In fact, $\mathrm{val}(G', l)$ is the number of m-subgraphs of $\mathcal{G}$ whose associated graph is G'. In addition, if there is a $s-t$ path in G', then there is a $s-t$ m-path in all of the m-subgraphs of $\mathcal{G}$ whose associated graph is G', and the result follows. $\square$
Definition: The *number* of polynomial m-subgraphs in a polynomial m-graph $\mathcal{G} = (V, l)$ is

$$\mathrm{sub}(\mathcal{G}, s, t) = \sum_{G' \text{ is a subgraph of } \mathcal{G}^*} \mathrm{con}(G', s, t)\mathrm{val}(G', l).$$

The hard part of our protocol will be the extension step. Its basic idea is that the value of $\mathrm{sub}(\mathcal{G}, s, t)$, can be divided into two terms. We can pick up any edge e, and can consider on the one hand all the $s-t$ connected m-subgraphs which do not contain e, and on the other hand all the $s-t$ connected m-subgraphs which do contain e. The computation of the first term is simply another instance of MULTI $s-t$ CONNECTEDNESS, which is given by the following proposition.
Definition: Let $\mathcal{G} = (V, l)$ be an m-graph and e an edge of $\mathcal{G}$. We define the m-graph $\mathcal{G}_{\overline{e}}$ as $(V, l_{\overline{e}})$, where $l_{\overline{e}}(e) = 0$, and $l_{\overline{e}}(e') = l(e')$ for every other edge slot e'.

Proposition 2 *Let $\mathcal{G} = (V, l)$ be an m-graph and let e be an edge. The number of $s-t$ connected m-subgraphs which do not contain e is* $\mathrm{sub}(\mathcal{G}_{\overline{e}})$.

Proof: Immediate. $\square$

The calculation of the number of $s-t$ connected m-subgraphs which contain a given edge e is more complicated. The main technical contribution of this protocol is to reduce this computation also to another instance of MULTI $s-t$ CONNECTEDNESS. The crucial notion for achieving this will be the contraction of an m-graph by an edge.
Definition: Let $\mathcal{G} = (V, E, l)$ be an m-graph, and let $e = \{u, v\} \in E$ be an edge, with

$v \notin \{s,t\}$. We define $\mathcal{G}_{e,v}$, the *contraction* of $\mathcal{G}$ by e with respect to the vertex v, as the m-graph $(V, l_{e,v})$ where for an edge slot $\{x,y\}$,

$$l_{e,v}(\{x,y\}) = \begin{cases} l(\{x,y\}) & \text{if } \{x,y\} \cap \{u,v\} = \emptyset, \\ 0 & \text{if } v \in \{x,y\}, \\ l(\{x,u\}) + l(\{x,v\}) + l(\{x,u\})l(\{x,v\}) & \text{if } y = u \text{ and } x \neq v. \end{cases}$$

The edge e is called the *contraction edge*, and v is its *isolation point.* When G is a graph, its contraction $G_{e,v}$ is defined as $((G^*)_{e,v})^*$.

For the ease of notation, when it is clear from the context that which vertex is the isolation point of the contraction edge, we will use $\mathcal{G}_e$ instead of $\mathcal{G}_{e,v}$, and l_e instead of $l_{e,v}$. We proceed similarly in the case of graphs. The following proposition relates the number of $s-t$ connected m-subgraphs in $G_{e,v}$ to the number of $s-t$ connected m-subgraphs in $\mathcal{G}$ which contain e.

Proposition 3 *Let $\mathcal{G} = (V,E,l)$ be an m-graph and let $e = \{u,v\} \in E$ be an edge with isolation point v. The number of $s-t$ connected m-subgraphs of $\mathcal{G}$ which contain e is $l(e)\mathrm{sub}(\mathcal{G}_e)$.*

Proof: It is given in the Appendix. □

From Proposition 2 and Proposition 3 we get immediately the following proposition which will constitute the basis of the consistency test of the verifier:

Proposition 4 *Let $\mathcal{G} = (V,E,l)$ be an m-graph and let $e = \{u,v\} \in E$ be an edge with isolation point v. Then we have:*

$$\mathrm{sub}(\mathcal{G}, s, t) = \mathrm{sub}(\mathcal{G}_{\overline{e}}, s, t) + l(e)\mathrm{sub}(\mathcal{G}_e, s, t).$$

If the above proposition were used as the consistency test, it would yield a protocol of interactive complexity $O(n^2)$. An easy iteration of this proposition can give a consistency test which yields a protocol of linear interactive complexity. The idea is to contract the m-graph by all the edges which are incident to some vertex v. This was first proposed by Yuri Matiyasevich.

Definition: Let $\mathcal{G} = (V,E,l)$ be an m-graph and v a vertex. Let $e_1 = \{u_1, v\}, \ldots, e_d = \{u_d, v\}$ the edges incident to v, enumerated in an arbitrary order, where $d = d(v)$ is the degree of v.

$$\mathcal{G}_{v,0} = (\ldots(\mathcal{G}_{\overline{e_1}})_{\overline{e_2}}\ldots)_{\overline{e_d}},$$

$$\mathcal{G}_{v,i} = ((\ldots(\mathcal{G}_{\overline{e_1}})_{\overline{e_2}}\ldots)_{\overline{e_{i-1}}})_{e_i}.$$

for $i = 1, \ldots, d$, i.e. the graph where $e_1, e_2, \ldots e_{i-1}$ are removed, and e_i is contracted on v.

If we iterate Proposition 4, we get :

Proposition 5 *Let $\mathcal{G} = (V,E,l)$ be an m-graph and v a vertex of degree d. Then we have*

$$\mathrm{sub}(\mathcal{G}, s, t) = \mathrm{sub}(\mathcal{G}_{v,0}, s, t) + \sum_{i=1}^{d} l(e_i)\mathrm{sub}(\mathcal{G}_{v,i}, s, t).$$

The instances $\mathcal{G}_{v,0}, \ldots, \mathcal{G}_{v,d}$ are smaller than $\mathcal{G}$ in the sense that the vertex v is isolated in them and therefore it can be disregarded. We describe now the protocol.

Extension step: At the beginning the list of the verifier contains a statement of the form $((\mathcal{G}, s, t), q)$. If the only non isolated vertices of $\mathcal{G}$ are s and t, he checks directly if $\mathrm{sub}(\mathcal{G}, s, t) = q$. Otherwise he chooses a non isolated vertex v different from s and t. Let d be the degree of v. The verifier asks the prover for $\mathrm{sub}(\mathcal{G}_{v,i}, s, t)$ and gets back q_i for $i = 0, \ldots, d$. He checks if

$$q = q_0 + \sum_{i=1}^{d} l(e_i) q_i.$$

If the equality does not hold he halts the protocol. Otherwise he replaces his list by $< ((\mathcal{G}_{v,0}, s, t), q_0), \ldots, ((\mathcal{G}_{v,d}, s, t), q_d) >$.

Reduction step: Let $< ((\mathcal{G}_{v,0}, s, t), q_0), \ldots, ((\mathcal{G}_{v,d}, s, t), q_d) >$ be the list of the verifier. For $0 \leq i \leq d$ let us define the polynomials

$$\delta_i(x) = \prod_{0 \leq j \leq d, j \neq i} \frac{(x-j)}{(i-j)}.$$

The polynomial m-graph created by $\mathcal{V}$ is $(\mathcal{G}(x), s, t)$, where $\mathcal{G}(x) = \sum_{i=0}^{d} \delta_i(x) \mathcal{G}_{v,i}$. Then he asks for the coefficients of the polynomial $\mathrm{sub}((\mathcal{G}(x), s, t))$ whose degree is of $O(n^3)$. From the answer of $\mathcal{P}$ he creates the polynomial $h(x)$ and checks if for every i, $h(i) = q_i$. If this is not true, $\mathcal{V}$ rejects. Otherwise, he randomly generates an element $a \in \mathbf{Z}_p$, and creates the list $< ((\mathcal{G}(a), s, t), h(a)) >$.

Theorem 1 MULTI $s-t$ CONNECTEDNESS $\in IP(n)$.

Proof: The correctness of the consistency test of the verifier follows from Proposition 5. The probabilistic analysis of the protocol is standard. In every reduction step a new vertex becomes isolated, therefore the number of rounds is indeed $O(n)$. $\square$

4 $s-t$ RELIABILITY

4.1 INTEGER versus PROBABILISTIC $s-t$ RELIABILITY

The draw a parallel between MULTI $s-t$ CONNECTEDNESS and the reliability problem, first we will define the the notion of a probabilistic graph. Like an m-graph, it will be a graph with a function on the edges, except that in this case the values of the function will be probabilities, that is rational numbers between 0 and 1. We also state the $s-t$ RELIABILITY problem in terms of probabilistic graphs and therefore will call it from now on PROBABILISTIC $s-t$ RELIABILITY. In the definition we make precise the representation of rational numbers.

Definition: A *probabilistic graph* (*p-graph*) $\mathcal{G}$ is a pair (V, r). As for m-graphs, V is the set of *vertices*, and $V^{(2)}$ is the set of *edge slots*. The *probability function* $r : V^{(2)} \rightarrow \mathbf{Q} \cap [0,1]$ defines for each edge slot its probability of existence. For every edge slot e, the rational number $r(e)$ is represented as a/b with $a, b \in \mathbf{N}$ and $\gcd(a,b) = 1$. The set $E = \{e \in V^{(2)} : r(e) \neq 0\}$ is the set of *edges* of $\mathcal{G}$.

The notions of *p-path, associated graph* and *associated p-graph* are defined as in the case of m-graphs. On the contrary, we will change the definition of the value of a probabilistic graph.
Definition: Let $G = (V, E)$ be a graph, and $r : V^{(2)} \to \mathbf{Q} \cap [0,1]$ a function. We define $\mathrm{pval}(G, r)$, the *probabilistic value* of G with respect to r as

$$\mathrm{pval}(G, r) = \prod_{e \in E} r(e) \prod_{e \notin E} (1 - r(e)).$$

We state now the original $s-t$ RELIABILITY problem in terms of probabilistic graphs.

- **PROBABILISTIC $s - t$ RELIABILITY**

  Input: $(\mathcal{G}, s, t)$, where $\mathcal{G} = (V, r)$ is a probabilistic graph with n vertices, and $s, t \in V$.

  Output: $\mathrm{rely}(\mathcal{G}, s, t)$, the probability that there exists a p-path from s to t.

Since the probability that there exists a p-path from s to t in $\mathcal{G}$ is clearly equal to the sum of the probabilistic values of all $s-t$ connected subgraphs of $\mathcal{G}$, we have the following proposition.

Proposition 6 *Let $\mathcal{G} = (V, r)$ be a p-graph, s and t two vertices. Then*

$$\mathrm{rely}(\mathcal{G}, s, t) = \sum_{G' \text{ is a subgraph of } \mathcal{G}^*} \mathrm{con}(G', s, t)\mathrm{pval}(G', r).$$

It is not easy to give a direct protocol for PROBABILISTIC $s - t$ RELIABILITY since we have to deal with the additional problem due to the probabilities which are rational numbers and which can not be directly interpreted in a field $\mathbf{Z}_p$. The first idea is to work instead of $\mathbf{Z}_p$ directly in the field of the rational numbers. But this is not a satisfying solution since during the reduction step the verifier should then generate random rational numbers which is clearly impossible. Instead we will define INTEGER $s - t$ RELIABILITY, which is the same problem as PROBABILISTIC $s-t$ RELIABILITY but defined on m-graphs. Naturally we can not define probabilities when the multiplicities of the edges are integer numbers but Proposition 6 actually gives a way to extend the function *rely* to that case. Then we will obtain an interactive protocol for PROBABILISTIC $s - t$ RELIABILITY in two steps. First we will give a protocol for INTEGER $s - t$ RELIABILITY. This will not be hard since the combinatorial structure of this problem is very similar to the structure of MULTI $s - t$ CONNECTEDNESS and therefore the protocol for the later will be easily modifiable to give a protocol for this problem. Then we will make a polynomial time many one reduction from PROBABILISTIC $s-t$ RELIABILITY to INTEGER $s - t$ RELIABILITY. This implies that the protocol for INTEGER $s - t$ RELIABILITY can also be used for PROBABILISTIC $s - t$ RELIABILITY.
Definition: Let $G = (V, E)$ be a graph, and $l : V^{(2)} \to \mathbf{Z}_p$ a function. We define the *probabilistic value* of G with respect to l as $\mathrm{pval}(G, l) = \prod_{e \in E} l(e) \prod_{e \notin E} (1 - l(e))$.
The definition of INTEGER $s - t$ RELIABILITY is the following:

- **INTEGER $s - t$ RELIABILITY**

  Input: $(\mathcal{G}, s, t)$, where $\mathcal{G} = (V, l)$ is an m-graph with n vertices, and $s, t \in V$.

Output: $\text{rely}(\mathcal{G}, s, t)$, where by definition

$$\text{rely}(\mathcal{G}, s, t) = \sum_{G' \text{ is a subgraph of } \mathcal{G}^*} \text{con}(G', s, t)\text{pval}(G', l).$$

4.2 A protocol for INTEGER $s-t$ RELIABILITY

The protocol for INTEGER $s-t$ RELIABILITY follows strongly the protocol for MULTI $s-t$ CONNECTEDNESS. The only important difference is that we change the definition of the contraction of the multi-graph in order to reflect the difference between the functions *val* and *pval*.

Definition: Let $\mathcal{G} = (V, E, l)$ be an m-graph, and let $e = \{u, v\} \in E$ be an edge, with $v \notin \{s, t\}$, and let d be the degree of v. We define $\mathcal{G}^p_{e,v}$, the *probabilistic contraction* of $\mathcal{G}$ by e with respect to the vertex v, as the m-graph $(V, r^p_{e,v})$ where

$$r^p_{e,v}(\{x, y\}) = \begin{cases} r(\{x, y\}) & \text{if } \{x, y\} \cap \{u, v\} = \emptyset, \\ 0 & \text{if } v \in \{x, y\}, \\ r(\{x, u\}) + r(\{x, v\}) - r(\{x, u\})r(\{x, v\}) & \text{if } y = u \text{ and } x \neq v. \end{cases}$$

The m-graph $\mathcal{G}^p_{\overline{e}}$ is defined as $\mathcal{G}_{\overline{e}}$. For $i = 0, \ldots, d$, the m-graphs $\mathcal{G}^p_{v,i}$ are defined as $\mathcal{G}_{v,i}$, except that we take probabilistic contraction.

The following two propositions are respectively analogous to Propositions 4 and 5.

Proposition 7 *Let $\mathcal{G} = (V, E, l)$ be an m-graph and let $e = \{u, v\} \in E$ be an edge with isolation point v. Then we have:*

$$\text{rely}(\mathcal{G}, s, t) = (1 - r(e))\text{rely}(\mathcal{G}^p_{\overline{e}}, s, t) + r(e)\text{rely}(\mathcal{G}^p_e, s, t).$$

Proposition 8 *Let $\mathcal{G} = (V, E, l)$ be an m-graph, v a vertex of degree d, and $e_1 = \{u_1, v\}, \ldots, e_d = \{u_d, v\}$ the edges incident to v. Then we have*

$$\text{rely}(\mathcal{G}, s, t) = \prod_{i=1}^{d}(1 - l(e_i))\text{rely}(\mathcal{G}^p_{v,0}) + \sum_{i=1}^{d} l(e_i) \prod_{j=1}^{i-1}(1 - l(e_j))\text{rely}(\mathcal{G}^p_{v,i}).$$

Theorem 2 INTEGER $s-t$ RELIABILITY $\in IP(n)$.

Proof: The protocol is the same as for MULTI $s-t$ CONNECTEDNESS except that we change the consistency test according to Proposition 8. $\Box$

4.3 The reduction

We can now turn to the question of reducing PROBABILISTIC $s-t$ RELIABILITY to INTEGER $s-t$ RELIABILITY. The main idea is to define a homomorphism from a subring of the rationals containing all the input probabilities to a field $\mathbf{Z}_p$. Let p be a prime number. We define the set $\mathbf{Q}_p$ as

$$\mathbf{Q}_p = \{a/b : a \in \mathbf{Z}, b \in \mathbf{N}, \gcd(b, p) = 1\}.$$

Clearly $\mathbf{Q}_p$ is a ring. We also define the function $f_p : \mathbf{Q}_p \to \mathbf{Z}_p$ as

$$f_p(a/b) = ab^{-1} \bmod p.$$

Lemma 1 *The function f_p is a ring homomorphism.*

We call f_p the canonical homomorphism. This function is not injective since it maps an infinite domain into a finite one. However, it is injective on a subset of $\mathbf{Q}_p$ whose elements have small numerator and denominator compared to p. This property is crucial for the reduction, and is expressed in the following lemma.

Lemma 2 *Let $0 \leq a_1, a_2 < \sqrt{p}$ and $0 < b_1, b_2 < \sqrt{p}$. If we have*

$$f(a_1/b_1) = f(a_2/b_2),$$

then

$$a_1/b_1 = a_2/b_2.$$

Proof: The hypothesis implies that

$$a_1 b_2 \bmod p = a_2 b_1 \bmod p.$$

Since $0 \leq a_1 b_2, a_2 b_1 < p$, we also have

$$a_1 b_2 = a_2 b_1,$$

which implies the result. $\square$

We are now ready to explain the reduction of PROBABILISTIC $s-t$ RELIABILITY to INTEGER $s-t$ RELIABILITY. Let $\mathcal{G} = (V, E, r)$ be a p-graph, with $|V| = n$, and s, t two vertices. Let $N = \max\{\text{denominator}(r(e)) : e \in E\}$, where $\text{denominator}(a/b) = b$ for $a, b \in \mathbf{N}$. Let $p > N^{2\binom{n}{2}}$ be a prime number. Let f_p be the canonical homomorphism from $\mathbf{Q}_p$ to $\mathbf{Z}_p$. We will define an m-graph denoted with some abuse of notation $f_p(\mathcal{G})$ as follows: $f_p(\mathcal{G}) = (V, l)$, where $l(e) = f_p(r(e))$.

Theorem 3 *Let $0 \leq a < \sqrt{p}$ and $0 < b < \sqrt{p}$. Then we have*

$$\text{rely}(\mathcal{G}, s, t) = a/b \Leftrightarrow \text{rely}(f_p(\mathcal{G}), s, t) = f_p(a/b).$$

Proof: First we show that

$$\text{rely}(\mathcal{G}, s, t) = a/b \Leftrightarrow f_p(\text{rely}(\mathcal{G}, s, t)) = f_p(a/b).$$

Let $\text{rely}(\mathcal{G}, s, t) = a'/b'$, with $\gcd(a', b') = 1$. Clearly $a' \leq b'$ since a'/b' is a probability. We will show that $0 < b' < \sqrt{p}$ and then apply Lemma 2. Let $c = \prod_{e \in E} \text{denominator}(r(e))$. For every subgraph G' of $\mathcal{G}^*$, the denominator of $\text{val}(G', r)$ is a divisor of c. Since $\text{rely}(\mathcal{G}, s, t)$ is the sum of $\text{val}(G', r)$ for all $s-t$ connected G', b' is also a divisor of c. Using that and the definition of N, we have $b' \leq c \leq N^{\binom{n}{2}} < \sqrt{p}$.

For the rest it is sufficient to show that

$$f_p(\text{rely}(\mathcal{G}, s, t)) = \text{rely}(f_p(\mathcal{G}), s, t).$$

But this is actually an easy consequence of the definition of $f_p(\mathcal{G})$ and Lemma 1. $\square$

Theorem 4 PROBABILISTIC $s-t$ RELIABILITY $\in IP(n)$.

Proof: Let $\mathcal{G} = (V, E, r)$ be a p-graph, with $|V| = n$, and s, t two vertices. Let $N = \max\{\text{denominator}(r(e)) : e \in E\}$. Let the prover's statement be "rely$(\mathcal{G}, s, t) = a/b$", with $\gcd(a, b) = 1$. The two parties agree on a prime p such that $N^{2\binom{n}{2}} < p < 2N^{2\binom{n}{2}}$. Observe that size of such a prime is polynomial in the size of $\mathcal{G}$. The verifier checks if $0 \leq a < \sqrt{p}$ and $0 < b < \sqrt{p}$. If this is not satisfied, he rejects. Otherwise the two parties run the protocol for INTEGER $s - t$ RELIABILITY on $f_p(\mathcal{G})$ and $f_p(a/b)$. The correctness of this protocol follows from Theorems 2 and 3. $\square$

Acknowledgements

We would like to thank Yuri Matiyasevich and Adi Shamir for several helpful conversations on the subject.

References

[1] L. Babai (1985), Trading group theory for randomness, *Proceedings of 17th ACM STOC*, 421-429.

[2] L. Babai and L. Fortnow (1990), A characterization of $\#P$ by arithmetic straight line programs, *Proceedings of 31st IEEE FOCS*, 26-34.

[3] L. Babai and S. Moran (1988), Arthur-Merlin games: A randomized proof system and a hierarchy of complexity classes, *Journal of Computer and System Sciences* 36, 254-276.

[4] R. Boppana, J. Hastad and S. Zachos (1987), Does $co - NP$ have short interactive proofs?, *Information Processing Letters* 25, 127-132.

[5] J. Diaz-Frias and M. de Rougemont (1992), A theory of robust planning, *Proceedings of IEEE International Conference on Robotics and Automation*, 2453-2459.

[6] S. Goldwasser, S. Micali and C. Rackoff (1989), The knowledge complexity of interactive proof systems, *SIAM Journal of Computing*, 18:1, 186-208.

[7] C. Lund, L. Fortnow, H. Karloff and N. Nisan (1990), Algebraic methods for interactive proof systems, *Proceedings of 31st IEEE FOCS*, 2-10.

[8] A. Shamir (1990), $IP = PSPACE$, *Proceedings of 31st IEEE FOCS*, 11-15.

[9] L. Valiant (1979), The complexity of enumeration and reliability problems, *SIAM Journal of Computing*, 8:3, 410-421.

Appendix - Proof of Proposition 3

We define $S^* = \{G : G \text{ is a subgraph of } \mathcal{G}^* \text{ containing } e\}$. Also by definition, let $(S_e)^* = \{G : G \text{ is a subgraph of } (\mathcal{G}_e)^*\}$, and $(S^*)_e = \{G : G \text{ is a subgraph of } (\mathcal{G}^*)_e\}$. By Proposition 1 it will be sufficient to prove the following equation:

$\sum_{G\in S^*} \text{con}(G)\text{val}(G,l) = l(e)\sum_{G\in(S_e)^*} \text{con}(G)\text{val}(G,l_e)$.(1)Our first observation is that in equation 4.3 we can consider the subgraphs of $(\mathcal{G}^*)_e$ instead of the subgraphs of $(\mathcal{G}_e)^*$. We have indeed

$$\sum_{G\in(S_e)^*} \text{con}(G)\text{val}(G,l_e) = \sum_{G\in(S^*)_e} \text{con}(G)\text{val}(G,l_e), \tag{2}$$

since $(S_e)^* \subseteq (S^*)_e$, and every $G \in (S^*)_e \setminus (S_e)^*$ contains an edge e' for which by definition $l_e(e') = 0$. Equation 4.3 therefore will follow from the following one:

$$\sum_{G\in S^*} \text{con}(G)\text{val}(G,l) = l(e)\sum_{G\in(S^*)_e} \text{con}(G)\text{val}(G,l_e). \tag{3}$$

The rest of the proof is devoted to the proof of Equation 3.

We define an equivalence relation $\equiv_e$ on S^* as follows: For two graphs $G_1, G_2 \in S^*$, we say that $G_1 \equiv_e G_2$ if $(G_1)_e = (G_2)_e$. We would like to characterize the equivalence classes of $\equiv_e$. For a graph $G = (V, E_0)$ in S^*, and for a vertex x such that $x \notin \{u, v\}$, let E_0^x be the set $\{\{x,u\},\{x,v\}\} \cap E_0$. The cardinality of such a set is between 0 and 2.

Lemma 3 *Let $G_1 = (V, E_1)$, and $G_2 = (V, E_2)$ two graphs in S^*. Then $G_1 \equiv_e G_2$ if and only if the following two conditions are satisfied:*

1. *for every edge slot $\{x, y\}$ such that $\{x,y\} \cap \{u,v\} = \emptyset$ we have*
$$\{x,y\} \in E_1 \iff \{x,y\} \in E_2,$$
2. *for every vertex $x \notin \{u, v\}$, we have*
$$|E_1^x| > 0 \iff |E_2^x| > 0.$$

Proof: Let $G = (V, E_0)$ be a graph in S^*, and set $(G)_e = (V, e_0')$. Then it follows from the definition of contraction that the following two conditions are satisfied:

1. for every edge slot $\{x, y\}$ such that $\{x,y\} \cap \{u,v\} = \emptyset$ we have
$$\{x,y\} \in E_0 \iff \{x,y\} \in e_0',$$
2. for every vertex $x \notin \{u, v\}$, we have
$$|E_0^x| > 0 \iff \{x,u\} \in e_0'. \quad \square$$

The next lemma relates the $s-t$ connectedness of graphs in S^* to the $s-t$ connectedness of their contraction.

Lemma 4 *Let G be a graph in S^*. Then G is $s-t$ connected if and only if G_e is $s-t$ connected.*

Proof: If there is no path in G between s and t then there is not either in G_e. If there is an $s-t$ path in G which does not go through v then the same path exists in G_e. If there is a path in G which contains v then it is of the form $< s, v_1, \ldots, v_{k-1}, v, v_k, \ldots, t >$. In this case either $< s, v_1, \ldots, v_{k-1}, v_k, \ldots, t >$ or $< s, v_1, \ldots, v_{k-1}, u, v_k, \ldots, t >$ is a path in G_e. Indeed, the former one exists when $v_{k-1} = u$ and the latter one when $v_{k-1} \neq u$. $\square$

Next, we show that the there is a bijection between the equivalence classes in S^* and the elements of $(S^*)_e$.

Lemma 5 *We have*

$$|S^*/\equiv_e| = |(S^*)_e|\,.$$

Proof: We define an application $f_e : S^* \to (S^*)_e$ by $f_e(G) = G_e$. We show that f_e is surjective. Let $G' = (V, e'_0) \in (S^*)_e$. It follows from Lemma 1 that for every vertex x such that $\{x, u\} \in e'_0$, in $\mathcal{G}^*$ at least one of the edges $\{x, u\}, \{x, v\}$ exists. Let $e(x)$ denote such an edge slot. We define $G = (V, E_0)$ as follows:

$$E_0 = \{\{u, v\}\} \cup \{e' \in e'_0 : u \notin e'\} \cup \{e(x) : \{x, u\} \in e'_0\}.$$

Then we have $f_e(G) = G'$. $\square$

For a graph $G' \in (S^*)_e$ let $< G' >$ denote the equivalence class which contains the graphs whose contraction is G'.

Lemma 6 *For every $G' \in (S^*)_e$, we have*

$$\sum_{G \in <G'>} \mathrm{val}(G, l) = l(e)\mathrm{val}(G', l_e)$$

Proof Let $G' = (V, e') \in (S^*)_e$, and let $X = \{x_1, \ldots, x_k\}$ be the set of vertices adjacent to u in G'. Then

$$\mathrm{val}(G', l_e) = \prod_{e' \in e', u \notin e'} l_e(e') \prod_{1 \le i \le k} l_e(\{x_i, u\})$$

and

$$\sum_{G \in <G'>} \mathrm{val}(G, l) = \prod_{e' \in e', u \notin e', v \notin e'} l(e') \prod_{1 \le i \le k} (l(\{x_i, u\}) + l(\{x_i, v\}) + l(\{x_i, u\})l(\{x_i, v\}).$$

By the definition of the function l_e these two products are equal. $\square$

Equation 3 and therefore Proposition 3 is a direct consequence Lemmas 4, 5 and 6. $\square$