Method for Performing Classical Bayesian Net Calculations Using a Quantum Computer

Robert R. Tucci

P.O. Box 226

Bedford, MA 01730

tucci@ar-tiste.com

May 22, 2004

CROSS REFERENCES TO RELATED APPLICATIONS

Not Applicable

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH AND DEVELOPMENT

Not Applicable

BACKGROUND OF THE INVENTION

(A)FIELD OF THE INVENTION

The invention relates to an array of quantum bits (qubits) commonly known as a quantum computer. More specifically, it relates to methods for translating an input data-set into a sequence of operations that can be used to manipulate said array. The invention also relates to classical probabilistic networks (classical Bayesian nets) and their quantum counterparts.

(B)DESCRIPTION OF RELATED ART

This invention deals with Quantum Computing. A quantum computer is an array of quantum bits (qubits) together with some hardware for manipulating these qubits. Quantum computers with several hundred qubits have not been built yet. However, once they are built, it is expected that they will perform certain calculations much faster that classical computers. A quantum computer follows a sequence of elementary operations. The operations are elementary in the sense that they act on only a few qubits (usually 1, 2 or 3) at a time. Henceforth, we will sometimes refer to

sequences as products and to operations as operators, instructions, steps or gates. Furthermore, we will abbreviate the phrase "sequence of elementary operations" by "SEO". SEOs are often represented as qubit circuits. In the quantum computing literature, the term "quantum algorithm" usually means a SEO for quantum computers for performing a desired calculation. Some quantum algorithms have become standard, such as those due to Deutsch-Jozsa, Shor and Grover. For a detailed discussion of quantum computing, see the books **Gru99:** J. Gruska, *Quantum Computing*, (Osborne McGraw-Hill, 1999), and **NieChu00:** M. Nielsen, I. Chuang, *Quantum Computation and Quantum Information*, (Cambridge University Press, 2000). Also, one can find on the Internet some excellent, free introductions to quantum computing.

This invention also deals with Classical Bayesian (CB) and Quantum Bayesian (QB) nets. (Most of the literature to date deals only with CB nets and refers to them simply as Bayesian nets (or networks), without the adjective "classical".)

A CB net comprises a graph (i.e., a diagram) and a matrix (with probabilities as entries) associated with each node of the graph. CB nets organize large amounts of probabilistic information and represent complicated probabilistic relationships. They do this in a way that is graphical, highly intuitive, and easily scalable. From the data contained within a CB net, one can derive many other conditional probabilities. Knowing such conditional probabilities is useful in applications of Decision Theory and Artificial Intelligence, wherein inferences are made based on uncertain knowledge. CB nets have been used in many areas, including pattern recognition, speech recognition, data mining, search engines, spam filters, expert systems, medical diagnosis, computer games with AI capabilities, etc. Companies which actively support the development and deployment of CB net technology include Microsoft, Intel, Google, etc. CB nets are also used in Defense (e.g., Star Wars missile discrimination). For a detailed discussion of CB nets, see the books **Jen01:** Finn V. Jensen, *Bayesian Networks and Decision Graphs* (Springer Verlag, 2001), and **Pea88:** Judea Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Mor-

gan Kaufmann, Palo Alto, 1988). Also, one can find on the Internet some excellent, free introductions to CB nets.

QB nets are a generalization of CB nets to quantum mechanics. A QB net comprises a graph and a matrix (with complex numbers called probability amplitudes or just amplitudes as entries) associated with each node of the graph. QB nets have been proposed as a graphical method for analyzing the behavior of quantum systems, in **QFogPat:** US Patent 5,787,236 by R.R. Tucci. QB net diagrams may be viewed as an alternative to qubit circuits. For an introduction to QB nets, see **QFogPat** and **Tuc99QIT:** R.R. Tucci, "Quantum Information Theory - A Quantum Bayesian Nets Perspective", ArXiv eprint quant-ph/9909039 .

The method proposed in this invention is based on an earlier method proposed by Fredkin-Toffoli (F-T) in **Tof80:** T. Toffoli, *Automata, Languages and Programming, 7th Coll.* (Springer Verlag, 1980) pg. 632, and in **FreTof82:** E. Fredkin, T. Toffoli, Int. Jour. of Th. Phys. (1982) Vol. 21, pg. 219. The F-T method is used in the field of (classical) reversible computing. F-T showed how, given any binary gate f (i.e., a function $f : \{0,1\}^r \to \{0,1\}^s$, for some integers r, s), one can construct another binary gate $\overline{f}$ such that $\overline{f}$ can be used to perform the same calculations as f, but in a reversible manner. We will call $\overline{f}$ a deterministic reversible extension (DRE) of f. Binary gates f and $\overline{f}$ can be represented as binary deterministic circuits. In this patent, we show how, given any CB net K^C, one can construct a QB net K^Q which is a "q-embedding" of K^C. ("q-" stands for "quantum-") By running K^Q on a quantum computer, one can calculate any conditional probability that one would be interested in calculating for the CB net K^C. Such conditional probabilities can be calculated with classical computers; the hope is that they can be calculated much faster with a quantum computer. Our method for constructing a q-embedding for a CB net is a generalization of the F-T method for constructing a DRE of a binary deterministic circuit. Thus, we generalize their method to the embedding of any classical stochastic circuit, not just binary deterministic ones.

Grover's algorithm is a quantum algorithm proposed in **GroPat:** US Patent 6,317,766, by Lov K. Grover. Our method of embedding a CB net K^C within a QB net K^Q can sometimes be used in combination with Grover's algorithm to great advantage. In certain cases, the target states that we wish to detect have probabilities that are too small to be measurable by running K^Q on a quantum computer. However, we will show that sometimes one can construct a new QB net, call it $K^{Q'}$, that magnifies to measurable values the target probabilities that were unmeasurable using K^Q alone. We will refer to $K^{Q'}$ as Grover's Microscope for K^Q, because $K^{Q'}$ is closely related to Grover's algorithm, and it magnifies some of the probabilities found with K^Q.

The independent claims of **GroPat** are 1 and 12. Claim 1 of **GroPat** refers to an "arrangement", presumably meaning quantum hardware such as a quantum computer. The claims of the present patent that pertain to Grover's algorithm require a classical computer that generates a sequence of operations; they do not require the actual application of said sequence of operations to a quantum computer. Claim 12 of **GroPat** refers to "A method for moving a quantum mechanical physical system which exists in a superposition of a plurality of states". Again, this seems to require quantum hardware to be manipulated according to the method. Even if it were judged that an embodiment of Claim 12 of **GroPat** did not require quantum hardware, the present patent claims a Grover-like algorithm only if used as a post-processor to a special, classical computer program of which a preferred embodiment called "Q-Embedder" is described below.

A quantum compiler is a computer program that one runs on classical computers. It can "compile" a unitary matrix; i.e., it can express a unitary matrix as a SEO that a quantum computer can follow. An early type of quantum compiler is discussed in **Bar95:** A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J.H. Smolin, H. Weinfurter, ArXiv eprint quant-ph/9503016. A different type of quantum compiler is discussed in **QbtrPat:** US Patent 6,456,994

B1, by R. R. Tucci, and in **Tuc99Qbtr:** R.R. Tucci, "A Rudimentary Quantum Compiler(2cnd ed.)", ArXiv eprint quant-ph/9902062.

To run a QB net on a quantum computer, as is proposed in this invention, we need to translate the QB net into an equivalent SEO. This can be done with the help of a quantum compiler. A possible method of accomplishing this is discussed in **QbtrPat** and in **Tuc98:** R.R. Tucci, "How to Compile a Quantum Bayesian Net", ArXiv eprint quant-ph/9805016 . Thus, the method of this invention promises to be fertile ground for the use of quantum compilers.

A precursor to this invention was first published in **TucV1:** R.R. Tucci, "Quantum Computer as an Inference Engine", ArXiv eprint quant-ph/0004028 Version 1, submitted on 6 Apr 2000. After **TucV1** was published, Tucci realized that the method of **TucV1** was flawed in some important respects. A new method, which is a substantial modification and correction of the method of **TucV1**, was published by Tucci in **TucV2:** R.R. Tucci, "Quantum Computer as a Probabilistic Inference Engine", ArXiv eprint quant-ph/0004028 Version 2, submitted on 19 Apr 2004. We emphasize that **TucV1** is now considered obsolete and flawed by Tucci. On the other hand, Tucci currently views **TucV2** as essentially correct and in agreement with this invention. **TucV1** and **TucV2** differ as follows. Contrary to **TucV2**, **TucV1** does not q-embed the root nodes of the CB net that it is trying to q-embed. More importantly, Section 6.3 of **TucV1** incorrectly claims that calculating a conditional probability for a CB net K^C using its q-embedding K^Q requires measurements of all internal nodes of K^Q. **TucV2** shows that only some of the external nodes of K^Q need to be measured.

JaePat: US Patent 6,675,154, by G.S. Jaeger, proposes the use of a quantum computer for performing Fuzzy Logic. (Fuzzy Logic is a field that started with the paper **Zad65:** Lotfi Zadeh, "Fuzzy Sets", Information and Control Vol 8 (1965) pgs. 338-353). Although it might at first appear that there is some overlap between the claims of **JaePat** and those of the present patent, careful reflection shows that this is

not the case. Indeed, the independent claims of **JaePat** are 1,2, 6 and 11. Claim 1 of **JaePat** requires the use of F1:"at least one fuzzy proposition". Claim 2 of **JaePat** requires the use of F2:"fuzzy logic operations". Claims 6 and 11 both require "fuzzy control", which presumably requires F1 or F2. The claims of the present patent do not require F1 or F2. It is also of interest to note that many scientists and engineers are highly critical of Fuzzy Logic, and consider Bayesian Nets a far better method for dealing with situations involving uncertain knowledge.

To summarize, the present invention has the following advantages over prior art:

- It merges ideas from various subjects (quantum computers, CB nets, QB nets, quantum compilers and Grover's algorithm) in a new way.
- It generalizes ideas of F-T used in classical reversible computing.
- It gives a method for performing CB net calculations on a quantum computer. Such calculations can be done on a classical computer. The hope is that they can be done much faster on a quantum computer.
- It uses ideas from **TucV2**, and avoids mistakes of **TucV1**. **TucV1** and **TucV2** were both written by the inventor of this patent.
- It shuns ideas from Fuzzy Logic in favor of Bayesian nets.

BRIEF SUMMARY OF THE INVENTION

A quantum computer is an array of quantum bits (qubits) together with some hardware for manipulating these qubits. A classical Bayesian (CB) net (or network) comprises a graph (i.e., a diagram) and a matrix (with probabilities as entries) associated with each node of the graph. A quantum Bayesian (QB) net comprises a graph and a matrix (with complex numbers called probability amplitudes or just amplitudes as entries) associated with each node of the graph.

A preferred embodiment of the invention comprises a classical computer that runs a special computer program. The program takes as input an initial data-set that contains probabilistic information and returns as output a sequence of elementary operations (SEO). The initial data-set helps determine a CB net K^C. A program called "Q-Embedder" q-embeds K^C within a QB net K^Q. ("q-" stands for "quantum-"). A program called "Qubiter" then translates K^Q into an equivalent SEO. Qubiter is an example of a type of program called a quantum compiler.

The SEO outputted by the classical computer can be used to manipulate an array of qubits in a quantum computer. Application of the SEO to the array, followed by a measurement of the array, yields the value of certain conditional probabilities that we wish to know.

A probability matrix is any matrix such that each column of the matrix constitutes a discrete probability distribution. Given a probability matrix P, this patent defines a unitary matrix U called a q-embedding of P. U carries all information contained in P. This patent describes a method for constructing one q-embedding U (out of many possible ones) for an arbitrary probability matrix P.

Given a CB net K^C, this patent defines a QB net K^Q called a q-embedding of K^C. By running K^Q on a quantum computer, one can calculate any conditional probability that one would be interested in calculating for the original CB net K^C. This patent describes a method for constructing one q-embedding K^Q (out of many possible ones) for an arbitrary CB net K^C. The method involves replacing each node matrix of K^C by a q-embedding. Doing this replacement of node matrices requires adding to the graph of K^C new nodes called marginalizer and ancilla nodes.

This patent also shows how to use a version of Grover's algorithm in combination with Q-Embedder and Qubiter.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a labelled graph and the four node matrices associated with the four nodes of the graph. The A_j are complex amplitudes. This figure illustrates a QB net. Suppose we replaced A_j everywhere in this figure by probabilities P_j. Then the figure would be an illustration of a CB net.

FIG. **2** shows how to construct a q-embedding of an arbitrary probability matrix $P(y|x)$. For definiteness, we assume in this figure that $x \in \{0, 1, 2\}$ and $y \in \{0, 1, 2, 3\}$. Shaded columns can be filled using the Gram-Schmidt algorithm.

FIG. **3** shows a CB net for 2-body scattering. The figure includes the net's graph and a table of its node probabilities. We use this net to illustrate how one can construct a q-embedding of an arbitrary CB net.

FIG. **4** shows a CB net (defined by a graph and a table of node probabilities) obtained by adding marginalizer nodes to the CB net of FIG.**3**.

FIG. **5** shows, for the scattering QB net, its graph.

FIG. **6** shows, for the scattering QB net, a table of its node amplitudes.

FIG. **7** shows, for the scattering QB net, the probability amplitude of its external nodes.

FIG. **8** shows, for the lung-disease-diagnosis CB net, its graph, and a table of its node probabilities.

FIG. **9** shows, for the lung-disease-diagnosis QB net, its graph.

FIG. **10** shows, for the lung-disease-diagnosis QB net, a table of its node amplitudes.

FIG. **11** shows, for the voting CB net, its graph and a table of its node probabilities.

FIG. **12** shows, for the voting QB net, its graph, a table of its node amplitudes and the amplitude of its external nodes.

FIG. **13** shows various vectors relevant to the algorithm that we call "Grover's Microscope".

FIG. **14** shows two boxes, each representing a text file. Together, these two text files fully specify a QB net.

FIG. **15** shows a block diagram of a classical computer feeding data to a quantum computer.

DETAILED DESCRIPTION OF THE INVENTION

(A)Theory Behind New Method

Henceforth, we use the following notation.

We will use the word "ditto" as follows. If we say "A (ditto, X) is smaller than B (ditto, Y)", we mean "A is smaller than B" and "X is smaller than Y".

The prefix "q-" will to stand for "quantum-" (as in "q-embedding") and the prefix "c-" will stand for "classical-".

Let $Bool = \{0, 1\}$. Let $Z_{a,b} = \{a, a+1, a+2, \ldots, b-1, b\}$ for arbitrary integers a and b such that $a \leq b$. For any finite set S, $|S|$ will denote the cardinality of S (i.e., the number of elements in S).

$\delta(x, y)$ will denote the Kronecker delta function; it equals one if $x = y$ and zero otherwise. For any statement St, we define the truth function $\theta(St)$ to equal 1 if St is true and 0 if St is false. For example, $\delta(x, y) = \theta(x = y)$.

$\oplus$ will denote addition mod 2. Suppose $\vec{x} = (x_0, x_1, x_2, \ldots) \in Bool^{\infty}$. We will call $x = \sum_{\alpha=0}^{\infty} x_\alpha 2^\alpha$ the decimal representation of $\vec{x}$ and denote it by $dec(\vec{x})$.

We will use the symbol $\sum_{\cdot}$ to denote a sum of whatever is on the right hand side of this symbol, where we sum over those indices with a dot underneath them.

For example, $\sum_{\underline{a}} f(\underline{a}) = \sum_a f(a)$

Suppose function f maps set S into the complex numbers. We will use $f(x)/(\sum_x \mathit{numerator})$ to represent $f(x)/(\sum_{x\in S} f(x))$. Thus, "*numerator*" stands for the numerator of the fraction.

Henceforth, we will either underline or put a caret over random variables. (ArXiv publications by Tucci (e.g., **TucV2**) indicate random variables by underlining them.) For example, $P(\hat{a} = a) = P_{\hat{a}}(a)$ will denote the probability that the random variable $\hat{a}$ assumes value a. $P(\hat{a} = a)$ will often be abbreviated by $P(a)$ when no confusion will arise. $S_{\hat{a}}$ will denote the set of values which the random variable $\hat{a}$ may assume, and $N_{\hat{a}}$ will denote the number of elements in $S_{\hat{a}}$. $pd(B|A)$ will stand for the set of probability distributions $P(\cdot|\cdot)$ such that $P(b|a) \geq 0$ and $\sum_{b'\in B} P(b'|a) = 1$ for all $a \in A$ and $b \in B$.

$H_1 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ is the one bit Hadamard matrix. $H_{N_B} = H_1^{\otimes N_B}$ (the n-fold tensor product of H_1) is the N_B bit Hadamard matrix.

Let $\vec{\kappa} = (\kappa_0, \kappa_1, \ldots, \kappa_{N_B-1})$ label N_B bits. Assume all κ_i are distinct. We will often use $N_S = 2^{N_B}$, where N_B stands for number of bits and N_S for number of states. If $|\phi\rangle_{\kappa_i} = |\phi(\kappa_i)\rangle$ is a ket for qubit κ_i, define $|\phi\rangle_{\vec{\kappa}} = |\phi(\vec{\kappa})\rangle = \prod_{i=0}^{N_B-1} |\phi(\kappa_i)\rangle$. For example, if

$$|0\rangle_{\kappa_i} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{1}$$

for all i, then

$$\begin{aligned} |0\rangle_{\vec{\kappa}} &= \prod_{i=0}^{N_B-1} |0\rangle_{\kappa_i} \\ &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= [1, 0, 0, \ldots, 0]^T . \end{aligned} \tag{2}$$

Likewise, if $\Omega(\kappa_i)$ is an operator acting on qubit κ_i, define $\Omega(\vec{\kappa}) = \prod_{i=0}^{N_B-1} \Omega(\kappa_i)$. For example, $H_1(\vec{\kappa}) = \prod_{i=0}^{N_B-1} H_1(\kappa_i)$ is the N_B bit Hadamard matrix.

Suppose ϕ is a normalized ($\phi^\dagger\phi = 1$) complex vector. Define the projection and reflection operators for ϕ by

$$\Pi_\phi = \phi\phi^\dagger \ , \quad R_\phi = 1 - 2\Pi_\phi \ . \tag{3}$$

Note that $\Pi_\phi^2 = \Pi_\phi$. If $x' = R_\phi x$, then x' is the reflection of x with respect to the plane perpendicular to ϕ. For example, $R_\phi\phi = -\phi$.

Next we will present a brief review of CB and QB nets. For more information about CB nets see **Jen01** or **Pea88** or the internet. For more information about QB nets, see **QFogPat** or **Tuc99QIT**. First, we will discuss QB nets. Then we will point out how CB nets differ from QB nets.

We call a *graph* (or a diagram) a collection of nodes with arrows connecting some pairs of these nodes. The arrows of the graph must satisfy certain constraints. We call a *labelled graph* a graph whose nodes are labelled. A *QB net* consists of two parts: a labelled graph with each node labelled by a random variable, and a collection of node matrices, one matrix for each node. These two parts must satisfy certain constraints.

An *internal arrow* is an arrow that has a starting (source) node and a different ending (destination) one. We will use only internal arrows. We define two types of nodes: an *internal or non-leaf node* is a node that has one or more internal arrows leaving it, and an *external or leaf node* is a node that has no internal arrows leaving it. It is also common to use the terms *root node* or *prior probability node* for a node that has no incoming arrows (if any arrows touch it, they are outgoing ones).

We restrict our attention to *acyclic* graphs; that is, graphs that do not contain cycles. (A *cycle* is a closed path of arrows with the arrows all pointing in the same sense.)

We assign a random variable to each node of the QB net. Suppose the random

variables assigned to the N nodes are $\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_N$. For each $j \in Z_{1,N}$, the random variable $\hat{x}_j$ will be assumed to take on values within a finite set S_j called *the set of possible states of* $\hat{x}_j$.

For example, consider the net of FIG.**1**. Nodes **11**, **12** and **13** are internal and node **14** is external. Node **11** is a root node. There are four nodes so $N = 4$. We will assume that the four nodes must lie in one of two states: either *no* or *si*. Thus, $S_1 = S_2 = S_3 = S_4 = \{no, si\}$.

If $\Gamma = \{k_1, k_2, \cdots, k_{|\Gamma|}\} \subset Z_{1,N}$, and $k_1 < k_2 < \cdots < k_{|\Gamma|}$, define $(x.)_\Gamma = (x_{k_1}, x_{k_2}, \cdots, x_{k_{|\Gamma|}})$ and $(\hat{x}.)_\Gamma = (\hat{x}_{k_1}, \hat{x}_{k_2}, \cdots, \hat{x}_{k_{|\Gamma|}})$. Sometimes, we also abbreviate $(x.)_{Z_{1,N}}$ (i.e., the vector that includes all the possible x_j components) by just $x.$, and $(\hat{x}.)_{Z_{1,N}}$ by just $\hat{x}.$.

For example, suppose $N = 4$. One has $Z_{1,4} = \{1, 2, 3, 4\}$. If $\Gamma = \{1, 3\}$, then $|\Gamma| = 2$. Furthermore, $(x.)_\Gamma = (x_1, x_3)$ and $(\hat{x}.)_\Gamma = (\hat{x}_1, \hat{x}_3)$. One defines $x. = (x.)_{Z_{1,4}} = (x_1, x_2, x_3, x_4)$ and $\hat{x}. = (\hat{x}.)_{Z_{1,4}} = (\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4)$.

Let Z_{ext} be the set of all $j \in Z_{1,N}$ such that $\hat{x}_j$ is an external node, and let Z_{int} be the set of all $j \in Z_{1,N}$ such that $\hat{x}_j$ is an internal node. Clearly, Z_{ext} and Z_{int} are disjoint and their union is $Z_{1,N}$.

For example, for FIG.**1**, $Z_{ext} = \{4\}$ and $Z_{int} = \{1, 2, 3\}$.

Each possible value $x.$ of $\hat{x}.$ defines a different *net story*. For any net story $x.$, we call $(x.)_{Z_{int}}$ the *internal state of the story* and $(x.)_{Z_{ext}}$ its *external state*.

For example, a possible story for the net of FIG.**1** is the case when $\hat{x}_1 = \hat{x}_2 = si$ and $\hat{x}_3 = \hat{x}_4 = no$. This net story may also be represented by $\hat{x}. = (si, si, no, no)$. Since we are assuming that each of the four nodes of FIG.**1** can assume two states, there are total of $2^4 = 16$ stories possible for the net of FIG.**1**. For story $\hat{x}. = (si, si, no, no)$, the internal state is $(x_1, x_2, x_3) = (si, si, no)$ and the external state is $x_4 = no$.

For each net story, we may assign an amplitude to each node. Define Γ_j to be the set of all k such that an arrow labelled x_k (i.e., an arrow whose source node is $\hat{x}_k$)

enters node $\hat{x}_j$. We say nodes $(\hat{x}.)_{\Gamma_j}$ are *parents* of node $\hat{x}_j$, and $\hat{x}_j$ is a *child* of nodes $(\hat{x}.)_{\Gamma_j}$. We assign a complex number $A_j[x_j|(x.)_{\Gamma_j}]$ to node $\hat{x}_j$. We call $A_j[x_j|(x.)_{\Gamma_j}]$ the *amplitude of node* $\hat{x}_j$ *within net story* $x.$.

For example, consider an arbitrary net story, call it (x_1, x_2, x_3, x_4), of FIG.**1**. No arrow enters node $\hat{x}_1$ so both Γ_1 and $(x.)_{\Gamma_1}$ are empty. Node $\hat{x}_2$ is entered by an arrow from node $\hat{x}_1$ so $\Gamma_2 = \{1\}$ and $(x.)_{\Gamma_2} = (x_1)$. Likewise, $\Gamma_3 = \{1\}$ and $(x.)_{\Gamma_3} = (x_1)$. Finally, $\Gamma_4 = \{2, 3\}$ and $(x.)_{\Gamma_4} = (x_2, x_3)$. We assign the complex number $A_1[x_1]$ to node $\hat{x}_1$, $A_2[x_2|x_1]$ to node $\hat{x}_2$, $A_3[x_3|x_1]$ to node $\hat{x}_3$, and $A_4[x_4|x_2, x_3]$ to node $\hat{x}_4$.

The *amplitude of net story* $x.$, call it $A(x.)$, is defined to be the product of all the node amplitudes $A_j[x_j|(x.)_{\Gamma_j}]$ for $j \in Z_{1,N}$. Thus,

$$A(x.) = \prod_{j \in Z_{1,N}} A_j[x_j|(x.)_{\Gamma_j}] \; . \tag{4}$$

For example, consider an arbitrary net story, call it (x_1, x_2, x_3, x_4), of FIG.**1**. One has that

$$A(x_1, x_2, x_3, x_4) = A_1[x_1]A_2[x_2|x_1]A_3[x_3|x_1]A_4[x_4|x_2, x_3] \; . \tag{5}$$

The function A_j with values $A_j[x_j|(x.)_{\Gamma_j}]$ determines a matrix that we will call the *node matrix of node* $\hat{x}_j$, and denote by Q_j. x_j is the matrix's *row index* and $(x.)_{\Gamma_j}$ is its *column index*.

For example, FIG.**1** gives the four node matrices Q_1, Q_2, Q_3, Q_4 associated with the four nodes of the graph shown there.

This concludes our brief review of QB nets. CB nets are the same a QB nets except that complex numbers (node amplitudes) $A_j[x_j|(x.)_{\Gamma_j}]$, are replaced by non-negative numbers (node probabilities) $P_j[x_j|(x.)_{\Gamma_j}]$. In analogy to Eq.(4), the *probability of net story* $x.$, call it $P(x.)$, is defined as

$$P(x.) = \prod_{j \in Z_{1,N}} P_j[x_j|(x.)_{\Gamma_j}] \ . \tag{6}$$

Whereas the node amplitudes of a QB net satisfy (usually)

$$\sum_{x_j \in S_j} \left| A_j[x_j|(x.)_{\Gamma_j}] \right|^2 = 1 \ , \tag{7}$$

the node probabilities of a CB net satisfy (usually)

$$\sum_{x_j \in S_j} P_j[x_j|(x.)_{\Gamma_j}] = 1 \ . \tag{8}$$

(We say "usually" because sometimes it might be convenient to use un-normalized P_j or A_j. Within the specification of this patent, we assume that node probabilities P_j and node amplitudes A_j are normalized. This should be interpreted as a preferred embodiment, not a necessity, of the invention.)

Refs.**QbtrPat** (see, for example, Eq.(20) of **QbtrPat**) and **Tuc98** show that given any QB net, one can find a (non-unique) unitary matrix, call it U_{net}, and an initial state vector, call it Ψ_0, so that U_{net} and Ψ_0 describe the state evolution for the situation captured by the QB net. One has

$$\Psi = U_{net} \Psi_0 \ , \tag{9}$$

where information about the root nodes of the QB net is encoded in the initial state vector Ψ_0, and information about the leaf nodes of the QB net is encoded in the final state vector Ψ.

Next we will define the q-embedding of a probability matrix and of a CB net.

A *probability matrix* $P(y|x)$ is a rectangular (not necessarily square) matrix with row index $y \in S_{\hat{y}}$ and column index $x \in S_{\hat{x}}$ such that $P(y|x) \geq 0$ for all x, y, and $\sum_y P(y|x) = 1$ for all x. The set of all probability matrices $P(y|x)$ where $x \in S_{\hat{x}}$ and $y \in S_{\hat{y}}$ will be denoted by $pd(S_{\hat{y}}|S_{\hat{x}})$ (pd = probability distribution). A probability matrix is assigned to each node of a CB net. A *probability matrix*

$P(y|x)$ *is deterministic* if for each column x, there exists a single row y, call it $f(x)$, such that $P(y|x) = \delta(f(x), y)$. Any map $f : S_{\hat{x}} \to S_{\hat{y}}$ uniquely specifies (and is uniquely specified) by the deterministic probability matrix P with matrix elements $P(y|x) = \delta(y, f(x))$ for all $x \in S_{\hat{x}}$ and $y \in S_{\hat{y}}$. We often talk about a map f and its associated probability matrix $P(y|x)$ as if they were the same thing.

A unitary matrix $A(y, \tilde{x}|x, \tilde{y})$ (with rows labelled by $y, \tilde{x}$ and columns by $x, \tilde{y}$) is a *q-embedding of probability matrix* $P(y|x)$ if

$$\sum_{\tilde{x}} |A(y, \tilde{x}|x, \tilde{y} = 0)|^2 = P(y|x) \tag{10}$$

for all possible values of y and x. We say $\tilde{y}$ is a *source index* and $\tilde{x}$ is a *sink index*. We also refer to $\tilde{x}$ and $\tilde{y}$ collectively as *ancilla indices*. If a q-embedding satisfies $A(y, \tilde{x}|x, \tilde{y}) \in Bool$ for all $y, \tilde{x}, x, \tilde{y}$, we say that it is a *deterministic q-embedding*. Examples of the q-embedding of a probability matrix will be given below.

Given a QB net K^Q, let

$$P[(x.)_L] = \left| \sum_{(x.)_{not(L)}} A(x.) \right|^2 . \tag{11}$$

On the right hand side of Eq.(11), $A(x.)$ is the amplitude of story $(x.)$, $not(L) = Z_Q - L$, where Z_Q is the set of indices of all the nodes of K^Q, and L is the set of indices of all external (leaf) nodes of K^Q. In other words, $not(L)$ is the set of internal (non-leaf) nodes of K^Q. We say K^Q is a *q-embedding of CB net* K^C if $P[(x.)_L]$ defined by Eq.(11) satisfies

$$P[(x.)_{Z_C}] = \sum_{(x.)_{L_1}} P[(x.)_L] , \tag{12}$$

where $L_1 \subset L$, and Z_C is the set of indices of all nodes of K^C. Thus, the probability distribution associated with all nodes of K^C can be obtained from the probability distribution associated with the external nodes of K^Q. Examples of the q-embedding of a CB net will be given below.

Next we will prove that any probability matrix has a q-embedding. Suppose that we are given a probability matrix $P(y|x)$ where $x \in S_{\hat{x}}$ and $y \in S_{\hat{y}}$. Let $N_{\hat{x}} = |S_{\hat{x}}|$ and $N_{\hat{y}} = |S_{\hat{y}}|$. Let $\xi^{(x)}$ for $x \in S_{\hat{x}}$ be any orthonormal basis of the complex $N_{\hat{x}}$ dimensional vector space. The components of $\xi^{(x)}$ will be denoted by $\xi^{(x)}_{\tilde{x}}$, where $\tilde{x} \in S_{\hat{x}}$. If the $\xi^{(x)}$'s are the standard basis, then $\xi^{(x)}_{\tilde{x}} = \delta(x, \tilde{x})$. Define matrix A by

$$\begin{aligned} & A(y, \tilde{x}|x, \tilde{y}) = \\ & = \begin{cases} \sqrt{P(y|x)}\ \ \xi^{(x)}_{\tilde{x}} & \text{if } \tilde{y} = 0 \\ \text{obtained by Gram-Schmidt method} & \text{if } \tilde{y} \neq 0 \end{cases} . \end{aligned} \tag{13}$$

To understand the last equation, consider FIG.**2**. In that figure we have assumed for definiteness that $S_{\hat{x}} = \{0, 1, 2\}$ and $S_{\hat{y}} = \{0, 1, 2, 3\}$. The shaded (ditto, unshaded) columns have $\tilde{y} \neq 0$ (ditto, $\tilde{y} = 0$). It is easy to see that the unshaded columns are orthonormal because the vectors $\xi^{(x)}$ are orthonormal and $\sum_y P(y|x) = 1$. Since the unshaded columns are orthonormal, one can use the Gram-Schmidt method to fill the shaded columns so that all the columns of A are orthonormal and therefore A is unitary. The Gram Schmidt method is covered in most Linear Algebra books. See, for example, the book **NobDan88:** B. Noble and J.W. Daniels, *Applied Linear Algebra*, Third Edition (Prentice Hall, 1988). Note that by virtue of Eq.(13),

$$\begin{aligned} & \sum_{\tilde{x}} |A(y, \tilde{x}|x, \tilde{y} = 0)|^2 = \\ & = \sum_{\tilde{x}} P(y|x) \xi^{(x)*}_{\tilde{x}} \xi^{(x)}_{\tilde{x}} \\ & = P(y|x) \end{aligned} \tag{14}$$

so that the A defined by Eq.(13) does indeed satisfy Eq.(10).

Note that the matrix A defined by Eq.(13) will have real entries if the $\xi^{(x)}$ basis is chosen to lie in the real $N_{\hat{x}}$ dimensional vector space and the Gram-Schmidt process is carried out in that same space. Thus, one can always find a q-embedding A for a probability matrix such that A is not merely unitary, but also orthogonal. However,

if A is destined to become a node matrix in a QB net, it may be counterproductive to constrain A to be real, since this constraint may cause SEO decompositions of A to be longer.

Note that the matrix A defined by Eq.(13) has dimensions $N_{\hat{x}}N_{\hat{y}} \times N_{\hat{x}}N_{\hat{y}}$. It is sometimes possible to find a smaller q-embedding of an $N_{\hat{y}} \times N_{\hat{x}}$ probability matrix $P(y|x)$. For example, suppose

$$P(y|x_1, x_2) = \delta(y, x_1 \oplus x_2) \ , \tag{15}$$

for $y, x_1, x_2 \in Bool$. Then define

$$A(y, e|x_1, x_2) = \frac{(-1)^{x_1 e}}{\sqrt{2}} \delta(y, x_1 \oplus x_2) \ , \tag{16}$$

for $y, e, x_1, x_2 \in Bool$. It is easy to check that matrix A is unitary. Furthermore,

$$\sum_e |A(y, e|x_1, x_2)|^2 = \delta(y, x_1 \oplus x_2) \ . \tag{17}$$

Next we will show that any CB net has a q-embedding. So far we've shown how to construct a q-embedding for any probability matrix. Remember that each node of a CB net K^C has a probability matrix assigned to it. The main step in constructing a q-embedding of K^C is to replace each node matrix of K^C by a q-embedding of it.

Before describing our construction method, we need some definitions. We say a node $\hat{m}$ is a *marginalizer node* if it has a single input arrow and a single output arrow. Furthermore, the parent node of $\hat{m}$, call it $\hat{x}$, has states $x = (x_1, x_2, \ldots, x_n)$, where $x_i \in S_{\hat{x}_i}$ for each $i \in Z_{1,n}$. Furthermore, for some particular integer $i_0 \in Z_{1,n}$, the set of possible states of $\hat{m}$ is $S_{\hat{m}} = S_{\hat{x}_{i_0}}$, and the node matrix of $\hat{m}$ is $P[\hat{m} = m|\hat{x} = (x_1, x_2, \ldots, x_n)] = \delta(m, x_{i_0})$.

Let K^C be a CB net for which we want to obtain a q-embedding. Our construction has two steps:

(Step 1) Add marginalizer nodes.

More specifically, replace K^C by a modified CB net ${K^C}_{mod}$ obtained as follows. For each node $\hat{x}$ of K^C, add a marginalizer node between $\hat{x}$ and every child of $\hat{x}$. If $\hat{x}$ has no children, add a child to it.

As an example of this step, consider the net K^C ("two body scattering net") defined by FIG.**3**. FIG.**3** consists of two parts: a graph, and a table giving the probability matrices associated with each node of the graph.

Applying Step 1 to K^C defined by FIG.**3** yields ${K^C}_{mod}$ defined by FIG.**4**. Note that in FIG.**4**, black circles denote all the marginalizer nodes added in Step 1, whereas white circles denote the original nodes of K^C.

(Step 2) Replace node probability matrices by their q-embeddings. Add ancilla nodes.

More specifically, replace ${K^C}_{mod}$ by a QB net K^Q obtained as follows. For each node of ${K^C}_{mod}$, except for the marginalizer nodes that were added in the previous step, replace its node matrix by a new node matrix which is a q-embedding of the original node matrix. Add a new node for each ancilla index of each new node matrix. These new nodes will be called *ancilla nodes* (of either the source or sink kind) because they correspond to ancilla indices.

Applying Step 2 to net ${K^C}_{mod}$ for two body scattering yields K^Q defined by the graph shown in FIG.**5** and the table of node amplitudes shown in FIG.**6**. Note that in FIG.**5**, black circles denote all the marginalizer and ancilla nodes added in Steps 1 and 2, whereas white circles denote the original nodes of K^C.

K^Q looks much more complicated than K^C, but it really isn't, since most of its node matrices are delta functions which quickly disappear when summing over node states.

According to the table of FIG.**6**, the probability amplitude for the external (aka leaf) nodes is given by the equation of FIG.**7**, where we have summed over all internal (non-leaf) nodes. The equation of FIG.**7** shows that the net K^Q that

we constructed from the net K^C by following Steps 1 and 2 satisfies the definition Eq.(12) that we gave earlier for a q-embedding of K^C. The probability distribution of the states of the external nodes of the QB net K^Q contains all the probabilistic information of the original CB net K^C. Hurray!

The q-embedding of a CB net, as defined by Eq.(12), is not unique. For example, we could have defined the graph of FIG.**5** without the nodes $\hat{a}_3$ and $\hat{b}_3$. We chose to include such nodes for pedagogical reasons.

As another example of q-embedding a CB net, consider the CB net ("lung-disease-diagnosis net") defined by FIG.**8**. The figure includes the net's graph and a table of its node probabilities. After applying Steps 1 and 2 to the CB net of FIG.**8**, one obtains the QB net defined by the graph of FIG.**9** and the table of FIG.**10**.

Next we will first present a CB net, call it K^C, that describes voting. Then we will find a QB net K^Q that is a q-embedding of K^C. In certain cases, the target states that we wish to detect have probabilities that are too small to be measurable by running K^Q on a quantum computer. However, we will show that sometimes one can construct a new QB net, call it $K^{Q\prime}$, that magnifies to measurable values the target probabilities that were unmeasurable using K^Q alone. We will refer to $K^{Q\prime}$ as Grover's Microscope for K^Q, because $K^{Q\prime}$ is closely related to Grover's algorithm, and it magnifies some of the probabilities found with K^Q.

Suppose $y \in Bool$ and $\vec{x} = (x^0, x^1, \ldots, x^{N_B-1}) \in Bool^{N_B}$. Let $f : Bool^{N_B} \rightarrow Bool$.

We will say that f is *AND-like* if $f(\vec{x}) = \theta(\vec{x} = \vec{x}_{targ})$ for some target vector $\vec{x}_{targ} \in Bool^{N_B}$. An AND-like f maps all $\vec{x}$ into zero except for $\vec{x}_{targ}$ which it maps into one. Thus, $|f^{-1}(1)| = 1$. An example of an AND-like f is the multiple AND gate $f(\vec{x}) = x^0 \wedge x^1 \wedge \ldots \wedge x^{N_B-1}$, which can also be expressed as $f(\vec{x}) = \theta[\vec{x} = (1, 1, \ldots, 1)]$.

We will say that f is *OR-like* if $f(\vec{x}) = \theta(\vec{x} \neq \vec{x}_{targ})$ for some target vector $\vec{x}_{targ} \in Bool^{N_B}$. An OR-like f maps all $\vec{x}$ into one except for $\vec{x}_{targ}$ which it maps into zero. Thus, $|f^{-1}(0)| = 1$. An example of an OR-like f is the multiple OR gate

$f(\vec{x}) = x^0 \vee x^1 \vee \ldots \vee x^{N_B-1}$, which can also be expressed as $f(\vec{x}) = \theta[\vec{x} \neq (0, 0, \ldots, 0)]$.

We will say that f has a *single target* if it is either AND-like or OR-like. If f has more than one target (i.e., if $|f^{-1}(0)|$ and $|f^{-1}(1)|$ are both greater than one), then we will say that f has *multiple targets*.

Suppose $y \in Bool$ and $\vec{x} = (x^0, x^1, \ldots, x^{N_B-1}) \in Bool^{N_B}$. Consider the CB net ("voting net") defined by FIG.**11**.

Henceforth, we will abbreviate $P(y = 0|\vec{x}) = p_i$ and $P(y = 1|\vec{x}) = q_i$, where $i = dec(\vec{x}) \in Z_{0,N_S-1}$. Hence $p_i + q_i = 1$ for all $i \in Z_{0,N_S-1}$. In general, the probability matrix $P(y|\vec{x})$ has 2^{N_B} free parameters (namely, p_i for all $i \in Z_{0,N_S-1}$). This number of parameters is forbiddingly large for large N_B. To ease the task of specifying $P(y|\vec{x})$, it is common to impose additional constraints on $P(y|\vec{x})$. An interesting special type of $P(y|\vec{x})$ is *deterministic* $pd(Bool|Bool^{N_B})$ *matrices*; that is, those that can be expressed in the form

$$P(y|\vec{x}) = \delta(y, f(\vec{x})) \ , \tag{18}$$

where $f : Bool^{N_B} \to Bool$. In this case, the voting net can be used to pose the *satisfiability problem (SAT)*: given $y = 0$, find the most likely $\vec{x} \in Bool^{N_B}$; in other words, find those $\vec{x}$ for which $f(\vec{x}) = 0$. If f is *OR-like* then all p_i equal zero except for one p_i which equals one. For example, for $N_B = 2$, if f is an OR gate, then

$$P(y|\vec{x})_{OR} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \ , \tag{19}$$

where the row indices are $y = 0, 1$ and the column indices are $\vec{x} = 00, 01, 10, 11$ in that order. A slightly more general type of $P(y|\vec{x})$ is *quasi-deterministic* $pd(Bool|Bool^{N_B})$ *matrices*; that is, those that can be expressed in the form

$$P(y|\vec{x}) = \sum_{\vec{t}} \delta(y, f(\vec{t}))P(t^0|x^0)P(t^1|x^1)\ldots P(t^{N_B-1}|x^{N_B-1}) \ , \tag{20}$$

where $f : Bool^{N_B} \to Bool$ and we sum over all $\vec{t} = (t^0, t^1, \ldots, t^{N_B-1}) \in Bool^{N_B}$.

When $f(\vec{t}) = t^0 \vee t^1 \vee \ldots \vee t^{N_B-1}$, $P(y|\vec{x})$ is called a *noisy-OR*. **TucV2** discusses how to q-embed deterministic and quasi-deterministic $pd(Bool|Bool^{N_B})$ matrices, and how to express their q-embeddings as a SEO .

A q-embedding for the CB net defined by FIG.**11** is given by the QB net defined by FIG.**12**.

According to table **122** of FIG.**12**, the probability amplitude for the external (leaf) nodes is given by equation **123** of FIG.**12**.

To fully specify the QB net for voting, we need to extend $A(\vec{x}_2|\vec{x}_1 = 0)$ and $A(\vec{x}_3, y_2|\vec{x}_2, y_1 = 0)$ into unitary matrices by adding columns to them. This can always be accomplished by applying the Gram-Schmidt algorithm. But sometimes one can guess a matrix extension, and this makes application of the Gram-Schmidt method unnecessary. If $P(\vec{x})$ is uniform (i.e., $P(\vec{x}) = 1/N_S$ for all $\vec{x}$, which means there is no a priori information about $\vec{x}$), then $A(\vec{x}_2|\vec{x}_1 = 0) = 1/\sqrt{N_S}$. In this case, we can extend $A(\vec{x}_2|\vec{x}_1 = 0)$ into the N_B bit Hadamard matrix H_{N_B}:

$$[A(\vec{x}_2|\vec{x}_1)] = H_{N_B}/\sqrt{N_S} \; . \tag{21}$$

(This works because all entries of the first column of H_{N_B} are equal to 1.) As to extending $A(\vec{x}_3, y_2|\vec{x}_2, y_1 = 0)$, this can be done as follows. Define

$$\Delta_p = diag(\sqrt{p_0}, \sqrt{p_1}, \ldots, \sqrt{p_{N_S-1}}) \; , \tag{22}$$

and

$$\Delta_q = diag(\sqrt{q_0}, \sqrt{q_1}, \ldots, \sqrt{q_{N_S-1}}) \; . \tag{23}$$

A possible way of extending $A(\vec{x}_3, y_2|\vec{x}_2, y_1 = 0)$ into a unitary matrix is

$$[A(\vec{x}_3, y_2|\vec{x}_2, y_1)] = \begin{pmatrix} \Delta_p & -\Delta_q \\ \Delta_q & \Delta_p \end{pmatrix} \; . \tag{24}$$

Unitary matrices of this kind are called D-matrices in **QbtrPat**. **QbtrPat** shows how to decompose any D-matrix into a SEO.

Next we will discuss Grover's Microscope for the voting QB net defined by FIG.**12**. For simplicity, we will assume that $P(\vec{x})$ is uniform.

Let $\vec{\kappa} = (\kappa_0, \kappa_1, \ldots, \kappa_{N_B-1})$ label N_B bits and let τ label another bit. Assume that τ and all the κ_i are distinct. Define

$$\phi_p = (\sqrt{p_0}, \sqrt{p_1}, \ldots, \sqrt{p_{N_S-1}})^T \ , \tag{25}$$

$$\phi_q = (\sqrt{q_0}, \sqrt{q_1}, \ldots, \sqrt{q_{N_S-1}})^T \ , \tag{26}$$

and

$$|\Psi\rangle = \Psi = \frac{1}{\sqrt{N_S}} \begin{pmatrix} \phi_p \\ \phi_q \end{pmatrix} \ . \tag{27}$$

Since $p_i + q_i = 1$ for all i, $\phi_p^T \phi_p + \phi_q^T \phi_q = N_S$. According to equation **123** of FIG.**12**, when $P(\vec{x})$ is uniform, the voting QB net fully specifies a unitary matrix U_{net} such that

$$|\Psi\rangle = U_{net} |0\rangle_{\vec{\kappa}} |0\rangle_{\tau} \ . \tag{28}$$

(The last equation is an example of Eq.(9).)

Define orthonormal vectors e_0 and e_1 by

$$e_0 = \begin{pmatrix} \phi_p/|\phi_p| \\ 0 \end{pmatrix} \ , \quad e_1 = \begin{pmatrix} 0 \\ \phi_q/|\phi_q| \end{pmatrix} \ . \tag{29}$$

If $P(y|\vec{x})$ is deterministic with OR-like f, then all components of e_0 are zero except for the one at the target state j_{targ}.

Ψ can be expressed in terms of e_0, e_1 as

$$\Psi = \frac{1}{\sqrt{N_S}}(|\phi_p|e_0 + |\phi_q|e_1) \ . \tag{30}$$

It is convenient to define a vector $\Psi_\perp$ orthogonal to Ψ:

$$\Psi_\perp = \frac{1}{\sqrt{N_S}}(|\phi_q|e_0 - |\phi_p|e_1) \ . \tag{31}$$

If $P(y|\vec{x})$ is deterministic with OR-like f, then $|\phi_p| = 1$ and $|\phi_q| = \sqrt{N_S - 1}$ so, for large N_S, $\Psi \approx e_1$ and $\Psi_\perp \approx e_0$. For an arbitrary angle α, let

$$\Psi'_\perp = \frac{1}{\sqrt{N_S}}\left[(c_{\frac{\alpha}{2}}|\phi_q| + s_{\frac{\alpha}{2}}|\phi_p|)e_0 + (s_{\frac{\alpha}{2}}|\phi_q| - c_{\frac{\alpha}{2}}|\phi_p|)e_1\right] \ , \tag{32}$$

where $s_A = \sin A$ and $c_A = \cos A$ for any angle A. Note that the angle between $\Psi'_\perp$ and $\Psi_\perp$ is $\alpha/2$. Also, the angle between e_1 and Ψ is $\theta/2$.

FIG.**13** portrays various vectors that arise in explaining Grover's Microscope. Note that $\Psi'_\perp = e_0$ when $\alpha = \theta$.

Since we plan to stay within the two dimensional vector space with orthonormal basis e_0, e_1, it is convenient to switch matrix representations. Within $span(e_0, e_1)$, e_0, e_1 can be represented more simply by:

$$e_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \ , \quad e_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \ . \tag{33}$$

If e_0, e_1 are represented in this way, then

$$\Psi = \frac{1}{\sqrt{N_S}} \begin{pmatrix} |\phi_p| \\ |\phi_q| \end{pmatrix} \ , \tag{34}$$

$$\Psi_\perp = \frac{1}{\sqrt{N_S}} \begin{pmatrix} |\phi_q| \\ -|\phi_p| \end{pmatrix} \ , \tag{35}$$

and

$$\Psi'_\perp = W\Psi \ , \tag{36}$$

where

$$W = \begin{pmatrix} c_{\frac{\alpha}{2}} & -s_{\frac{\alpha}{2}} \\ s_{\frac{\alpha}{2}} & c_{\frac{\alpha}{2}} \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} . \tag{37}$$

The matrix $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ is a clockwise rotation by $\pi/2$ in space $span(e_0, e_1)$. Thus, W equals a clockwise rotation by $\pi/2$ followed by a counter-clockwise rotation by $\alpha/2$.

Define the following reflection operators

$$R_0 = 1 - 2\Pi_{|0\rangle_{\vec{\kappa}}}\Pi_{|0\rangle_\tau} , \tag{38}$$

$$R_\Psi = U_{net} R_0 U_{net}^\dagger , \tag{39}$$

$$R_{\Psi'_\perp} = W R_\Psi W^\dagger . \tag{40}$$

It follows that

$$-R_\Psi R_{\Psi'_\perp} = c_\alpha \Psi\Psi^T - s_\alpha \Psi\Psi_\perp^T + s_\alpha \Psi_\perp \Psi^T + c_\alpha \Psi_\perp \Psi_\perp^T . \tag{41}$$

Thus, $-R_\Psi R_{\Psi'_\perp}$ rotates vectors in $span(e_0, e_1)$, clockwise by an angle α.

Grover's Microscope can be summarized by the following equation

$$(-R_\Psi R_{\Psi'_\perp})^r \Psi \approx e_0 , \tag{42}$$

for some integer r to be determined, where "$\approx$" means approximation at large N_S. What this means is that our system starts in state Ψ and is rotated consecutively r times, each time by a small angle α, until it arrives at the state e_0. If $P(y|\vec{x})$ is deterministic with OR-like f, then measuring state e_0 yields the target state j_{targ}.

The optimum number r of iterations is

$$r\alpha \approx \frac{\pi}{2}(1+2k) \tag{43}$$

for some integer k. Note that $\cos(\theta/2) = \langle\Psi|e_1\rangle = |\phi_q|/\sqrt{N_S}$. Hence, in general, θ depends on $|\phi_p|$ (or on $|\phi_q| = \sqrt{N_S - |\phi_p|^2}$). If $P(y|\vec{x})$ is deterministic with OR-like f, then $|\phi_p| = 1$ and $|\phi_q| = \sqrt{N_S - 1}$. In this case, it is convenient to choose $\alpha = \theta$, so that $\Psi'_\perp = e_0$. Then the optimum number r of iterations for Grover's original algorithm and for Grover's Microscope are equal. If we don't know ahead of time the value of $|\phi_p|$, then setting $\theta = \alpha$ will make both r and α depend on the unknown $|\phi_p|$, although the product $r\alpha$ will still be independent of it.

Let

$$\begin{aligned} U_{Gscope} &= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \\ &= -e_1 e_0^T + e_0 e_1^T \\ &= -\Psi\Psi_\perp^T + \Psi_\perp \Psi^T \ . \end{aligned} \tag{44}$$

Note that

$$U_{Gscope}\Psi = \Psi_\perp \ . \tag{45}$$

From the point of view of quantum compiling, Grover's Microscope approximates the $\pi/2$ rotation U_{Gscope} by the r-fold product of $-R_\Psi R_{\Psi'_\perp}$, where we assume that $-R_\Psi R_{\Psi'_\perp}$ can be shown to have a SEO of low (polynomial in N_B) complexity. (If such a low complexity SEO cannot be found, then it is pointless to divide U_{Gscope} into r iterations of $-R_\Psi R_{\Psi'_\perp}$, and we might be better off compiling U_{Gscope} all at once.)

(B)Computer Implementation of Theory

In Section (A), we described a mathematical algorithm for q-embedding any CB net within a QB net. Next we describe a particular implementation of this algorithm, a computer program called Q-Embedder that can be run on a classical computer.

To understand the input and output data of Q-Embedder, one must first understand the convention Q-Embedder uses for specifying CB and QB nets. Q-Embedder uses two text files to specify a QB net. An example is shown in FIG.**14**. In this figure, boxes **140** and **145** each represents a text file.

From text file **140** we learn that the QB net has 3 nodes called A, B and X. We also learn the possible states of each node. For example, node A has two possible states, $a1$ and $a2$. The hash symbol in line **141** indicates that a new node will follow. Line **142** names the node A being considered. Lines **143** list the two possible states, $a1, a2$, of A.

From text file **145**, we learn that nodes A, B, X are connected by two arrows: (1) from A to X, (2) from B to X. We also learn the node matrix for each of the nodes. For example, we learn that node A is a root node, and the amplitudes of its two states $a1$ and $a2$ are, respectively, $0.707 + 0i$ and $0 + 0.707i$. Node X has four parent states: $(B, A) = (b1, a1)$, $(b2, a1)$, $(b1, a2)$ and $(b2, a2)$. For the parent state $(B, A) = (b1, a1)$, the amplitudes of the two states $x1$ and $x2$ of X are, respectively, $1 + 0i$ and $0 + 0i$. The hash symbol in line **146** indicates that a new parent state will follow. Line **147** names the node X being considered. Lines **148** give the parent state $(B, A) = (b1, a1)$. Lines **149** give the amplitude of the states $x1, x2$.

There are many equivalent ways of specifying a QB net. In earlier examples, we specified a QB net by giving a graph (diagram) and a table specifying the amplitudes for each node. On the other hand, Q-Embedder specifies QB nets by means of two text files exemplified by FIG.**14**.

To specify a CB net instead of a QB net, Q-Embedder also uses two text files, almost identical to those exemplified by FIG.**14**. The only difference is that wherever QB net files list two real numbers separated by white space to represent a complex number (a node amplitude), CB net files list a single real number, from the interval [0,1], to represent a probability.

Now that we understand how Q-Embedder specifies CB nets and QB nets, it

is easy to describe the input and output data for Q-Embedder. Q-Embedder takes as input two text files that specify a CB net K^C, and it returns as output two text files that specify a QB net K^Q that is a q-embedding of K^C. For example, if the two input text files specify the CB net defined by FIG.**3**, then the two output text files will specify that QB net defined by FIG.**5** and FIG.**6**.

We will not present source code for Q-Embedder in this patent. Those skilled in the art of programming will find it a straightforward exercise to write a computer program like Q-Embedder that performs Steps 1 and 2. These steps were carefully described and illustrated with two detailed examples, two body scattering and lung disease diagnosis.

Next we will discuss how to combine Q-Embedder, Qubiter, and a quantum computer.

QbtrPat proposes a computer program for translating a QB net into an equivalent SEO. **QbtrPat** gives source code for a computer program called Qubiter-1.0 that can accomplish such translations partially, for two node QB nets. Then **QbtrPat** gives careful instructions on how to augment Qubiter-1.0 so that it can translate any QB net. Assume henceforth a computer program called Qubiter that can translate any QB net into a SEO.

Q-Embedder can be used in tandem with Qubiter. In such a configuration, Q-Embedder takes as input 2 text that specify a CB net, and it returns as output 2 text files that specify a QB net. Then Qubiter takes as input the 2 output files of Q-Embedder, and it returns as output an equivalent SEO.

Note that it may suffice to find a SEO that is only approximately (within a certain precision) equivalent instead of exactly equivalent to the QB net. This may be true if, for example, the probabilities associated with the CB net that was q-embedded were not specified too precisely to begin with.

A classical computer running Q-Embedder and Qubiter in tandem can feed the SEO produced by Qubiter to a quantum computer.

FIG.**15** is a block diagram of a classical computer feeding data to a quantum computer. Box **150** represents a classical computer. It comprises sub-boxes **151, 152, 153**. Box **151** represents input devices, such as a mouse or a keyboard. Box **152** represents the CPU, internal and external memory units. Box **152** does calculations and stores information. Box **153** represents output devices, such as a printer or a display screen. The graph (e.g., FIG.**3**) of a CB net, or the graph (e.g., FIG.**5**) of a QB net, can be rendered on the display screen. Box **155** represents a quantum computer, comprising an array of quantum bits and some hardware for manipulating the state of those bits. For more information about the organization of a present day classical computer, see **CPP:** J. Adams, S. Leestma, L. Nyhoff, "C++, an Introduction to Computing",(Prentice Hall, 1995) pages 19-20.

Next we describe how to calculate probabilities with a quantum computer. Consider the example of the CB net K^C given by FIG.**3** and its q-embedding, the QB net K^Q given by FIG.**5** and FIG.**6**. From the equation of FIG.**7**, it is clear that by running K^Q on a quantum computer, we can calculate any conditional probability that one would want to calculate for K^C. For example, suppose we wanted to calculate $P_{\hat{a},\hat{d}|\hat{x}}$. Run K^Q on the quantum computer several times, each time measuring nodes $\hat{a}_5, \hat{d}_3$ and $\hat{x}_{5d}$ and not measuring all other external nodes. The resulting measurements will be distributed according to the probability distribution $P_{\hat{a},\hat{d},\hat{x}}$. Nature will automatically take the magnitude squared of the amplitude $A(a_5, b_5, c_3, d_3, x_{5c}, x_{5d})$ and sum the result over the states of the un-measured external nodes. The laws of quantum mechanics guarantee it. Proceed in the same way to calculate $P_{\hat{x}}$. Run K^Q on the quantum computer several times, each time measuring node $\hat{x}_{5d}$ and not measuring all other external nodes. Finally divide $P_{\hat{a},\hat{d},\hat{x}}$ by $P_{\hat{x}}$ on a classical (or quantum) computer. This procedure works if we assign an integer number of qubits to each external node of K^Q, and if different external nodes are assigned different qubits. This way, when we say that we measured or did not measure an external node, we mean that we measured or did not measure the qubits assigned to that node. To implement

this idea, it is convenient to extend the set of possible states of each node of K^C so that the cardinality of the extended set equals a power of two. For example, for the CB net of FIG.**3**, let $N_{\hat{a}} = |S_{\hat{a}}|$. Then let

$$\overline{N}_{\hat{a}} = min\{2^n : n \in Z_{0,\infty}, N_{\hat{a}} \leq 2^n\} \ . \tag{46}$$

We extend $S_{\hat{a}}$ to a larger set $\overline{S}_{\hat{a}}$ which contains $S_{\hat{a}}$ and has $|\overline{S}_{\hat{a}}| = \overline{N}_{\hat{a}}$. We also define $P(a) = 0$ for $a \in \overline{S}_{\hat{a}} - S_{\hat{a}}$. In an analogous way, we extend $S_{\hat{b}}, S_{\hat{x}}, S_{\hat{c}}$ and $S_{\hat{d}}$ so that each has a cardinality which is a power of two. We also extend the functions $P(b)$, $P(x|a,b)$, $P(c|x)$ and $P(d|x)$ so that they take the same values on the old elements of the domain and vanish on the new ones.

Suppose samples $a_1, a_2, \ldots a_\nu$ belong to a finite set $S_{\hat{a}}$, and suppose that they are distributed according to a probability distribution $P_{\hat{a}}$. What number ν of samples a_i is necessary to estimate $P_{\hat{a}}$ within a given precision? This question is directly relevant to our method for estimating probabilities by running a QB net on a quantum computer. We will not give a detailed answer to this question here. For an answer, the reader can consult any book on the mathematical theory of Statistics. An imprecise rule of thumb is that if the support of $P_{\hat{a}}$ has ν_0 elements, then ν should be at least as large as ν_0; i.e., one needs at least "one data point per bin" to estimate $P_{\hat{a}}$ with any decent accuracy.

We've explained how to estimate a conditional probability for a CB net by running a QB net ν times on a quantum computer. If we wanted to find $P(y|x^0, x^1)$ for the voting CB net, then the number of runs ν required to estimate $P(y|x^0, x^1)$ with moderate accuracy would not be too onerous, because the domain of $P(y|x^0, x^1)$ is $Bool^3$, which contains only 8 points. But what if we wanted to estimate $P(y|\vec{x})$? For large N_B, the domain of $P(y|\vec{x})$ is very large (2^{N_B+1} points). If the support of $P(y|\vec{x})$ occupies a large fraction of this domain, then the number of runs ν required to estimate $P(y|\vec{x})$ with moderate accuracy is forbiddingly large. However, there are some cases in which "Grover's Microscope" can come to the rescue, by allowing us to

amplify certain salient features of $P(y|\vec{x})$ so that they become measurable in only a few runs.

So far, we have described some exemplary preferred embodiments of this invention. Those skilled in the art will be able to come up with many modifications to the given embodiments without departing from the present invention. Thus, the inventor wishes that the scope of this invention be determined by the appended claims and their legal equivalents, rather than by the given embodiments.

I claim:

1. A method of operating a classical computer having display, storage and calculation means, to calculate a q-net data-set based on a c-net data-set with the purpose of inducing a quantum computer to calculate a desired probability that depends on said c-net data-set, said method comprising the steps of:

displaying on said display means a c-graph comprising a plurality of N c-nodes, and a plurality of directed c-lines connecting certain pairs of said c-nodes,

storing said c-net data-set in said storage means, wherein said c-net data-set comprises:

(a) c-graph information comprising a c-node label for each of said N c-nodes, and also comprising, for each said directed c-line, said c-node label for the source c-node and for the destination c-node of the directed c-line,

(b) c-state information comprising, for each $j \in \{1, 2, \cdots N\}$, a finite set S_j containing labels for the states that the j'th c-node $\hat{x}_j$ may assume, and

(c) c-probability information comprising, for each $j \in \{1, 2, \cdots N\}$, a representation of a non-negative real number $P_j[x_j | x_{k_1}, x_{k_2}, \cdots, x_{k_{|\Gamma_j|}}]$ for each vector $(x_j, (x.)_{\Gamma_j}) = (x_j, x_{k_1}, x_{k_2}, \cdots, x_{k_{|\Gamma_j|}})$ such that $x_j \in S_j$, $x_{k_1} \in S_{k_1}$, $x_{k_2} \in S_{k_2}$, $\cdots$, and $x_{k_{|\Gamma_j|}} \in S_{k_{|\Gamma_j|}}$, wherein $(\hat{x}_{k_1}, \hat{x}_{k_2}, \cdots, \hat{x}_{k_{|\Gamma_j|}})$ are the $|\Gamma_j|$ c-nodes connected to $\hat{x}_j$ by directed c-lines entering $\hat{x}_j$, wherein said $|\Gamma_j|$ is an integer greater or equal to zero,

composing said q-net data-set with said calculation means and using said c-net data-set, wherein said q-net data-set comprises

(a′) q-graph information comprising a q-node label for each q-node of a plurality of N' q-nodes, and also comprising a plurality of directed q-lines, wherein a directed q-line comprises an ordered pair of said

q-node labels, wherein one member of the label pair labels the source q-node and the other member labels the destination q-node of the directed q-line,

(b′) q-state information comprising, for each $j \in \{1, 2, \cdots N'\}$, a finite set S'_j containing labels for the states that the j'th q-node $\hat{y}_j$ may assume, and

(c′) q-amplitude information comprising, for each $j \in \{1, 2, \cdots N'\}$, a representation of a complex number $A_j[y_j|y_{k_1}, y_{k_2}, \cdots, y_{k_{|\Gamma'_j|}}]$ for each vector $(y_j, (y.)_{\Gamma'_j}) = (y_j, y_{k_1}, y_{k_2}, \cdots, y_{k_{|\Gamma'_j|}})$ such that $y_j \in S'_j$, $y_{k_1} \in S'_{k_1}$, $y_{k_2} \in S'_{k_2}$, $\cdots$, and $y_{k_{|\Gamma'_j|}} \in S'_{k_{|\Gamma'_j|}}$, wherein $(\hat{y}_{k_1}, \hat{y}_{k_2}, \cdots, \hat{y}_{k_{|\Gamma'_j|}})$ are the $|\Gamma'_j|$ nodes connected to $\hat{y}_j$ by directed lines entering $\hat{y}_j$, wherein said $|\Gamma'_j|$ is an integer greater or equal to zero,

wherein if

$$P(x.) = \prod_{j=1}^{N} P_j[x_j|(x.)_{\Gamma_j}]/(\sum_{(x.)} numerator),$$

and

$$A(y.) = \prod_{j=1}^{N'} A_j[y_j|(y.)_{\Gamma'_j}]/(\sum_{(y.)} |numerator|^2),$$

and L is the set of all j such that $\hat{y}_j$ is a leaf node of said q-net data-set, and $not(L) = \{1, 2, \ldots N'\} - L$, and

$$A_L[(y.)_L] = \sum_{(y.)_{not(L)}} A(y.),$$

then, for all $x_1 \in S_1$, $x_2 \in S_2, \ldots$, and $x_N \in S_N$, $P(x.)$ is a sum of some numbers from the set

$$\{|A_L[(y.)_L]|^2 : \text{ for all possible values of } (y.)_L\}.$$

2. The method of claim 1, comprising the additional step of:

calculating with said classical computer, a data-set that specifies a unitary matrix U_{net}, wherein U_{net} describes the state evolution for the situation captured by said q-net data-set.

3. The method of claim 2, comprising the additional steps of:

   calculating with said classical computer, a data-set that specifies a unitary matrix T, wherein if $\Psi = U_{net}|0\rangle$, and if $\Psi' = T\Psi$, and if all the components Ψ_j of Ψ have an absolute value that is small compared to one, then some component Ψ'_{j_0} of Ψ' has an absolute value that is close to one.

   calculating with said classical computer, a sequence of operations, wherein said sequence of operations and said T both would, if applied to an array of qubits, produce equivalent transformations of the array.

4. The method of claim 3, wherein said sequence of operations comprises elementary operations on qubits.

5. The method of claim 3, wherein said sequence of operations is a sequence of elementary operations on qubits.

6. The method of claim 2, comprising the additional step of:

   calculating with said classical computer, a sequence of operations, wherein said sequence of operations and said U_{net} both would, if applied to an array of qubits, produce equivalent transformations of the array.

7. The method of claim 6, wherein said sequence of operations comprises elementary operations on qubits.

8. The method of claim 6, wherein said sequence of operations is a sequence of elementary operations on qubits.

9. The method of claim 6, further utilizing a quantum computer, comprising the additional step of:

   manipulating said quantum computer largely according to said sequence of operations.

10. A method of operating a classical computer to calculate a q-net data-set based on a c-net data-set with the purpose of inducing a quantum computer to calculate a desired probability that depends on said c-net data-set, said method comprising the steps of:

storing said c-net data-set in said classical computer, wherein said c-net data-set comprises:

(a) c-graph information comprising a c-node label for each c-node of a plurality of N c-nodes, and also comprising a plurality of directed c-lines, wherein a directed c-line comprises an ordered pair of said c-node labels, wherein one member of the label pair labels the source c-node and the other member labels the destination c-node of the directed c-line,

(b) c-state information comprising, for each $j \in \{1, 2, \cdots N\}$, a finite set S_j containing labels for the states that the j'th c-node $\hat{x}_j$ may assume, and

(c) c-probability information comprising, for each $j \in \{1, 2, \cdots N\}$, a representation of a non-negative real number $P_j[x_j | x_{k_1}, x_{k_2}, \cdots, x_{k_{|\Gamma_j|}}]$ for each vector $(x_j, (x.)_{\Gamma_j}) = (x_j, x_{k_1}, x_{k_2}, \cdots, x_{k_{|\Gamma_j|}})$ such that $x_j \in S_j$, $x_{k_1} \in S_{k_1}$, $x_{k_2} \in S_{k_2}$, $\cdots$, and $x_{k_{|\Gamma_j|}} \in S_{k_{|\Gamma_j|}}$, wherein $(\hat{x}_{k_1}, \hat{x}_{k_2}, \cdots, \hat{x}_{k_{|\Gamma_j|}})$ are the $|\Gamma_j|$ c-nodes connected to $\hat{x}_j$ by directed c-lines entering $\hat{x}_j$, wherein said $|\Gamma_j|$ is an integer greater or equal to zero,

composing said q-net data-set with said classical computer and using said c-net data-set, wherein said q-net data-set comprises

(a′) q-graph information comprising a q-node label for each q-node of a plurality of N' q-nodes, and also comprising a plurality of directed q-lines, wherein a directed q-line comprises an ordered pair of said q-node labels, wherein one member of the label pair labels the source

q-node and the other member labels the destination q-node of the directed q-line,

(b′) q-state information comprising, for each $j \in \{1, 2, \cdots N'\}$, a finite set S'_j containing labels for the states that the j'th q-node $\hat{y}_j$ may assume, and

(c′) q-amplitude information comprising, for each $j \in \{1, 2, \cdots N'\}$, a representation of a complex number $A_j[y_j|y_{k_1}, y_{k_2}, \cdots, y_{k_{|\Gamma'_j|}}]$ for each vector $(y_j, (y.)_{\Gamma'_j}) = (y_j, y_{k_1}, y_{k_2}, \cdots, y_{k_{|\Gamma'_j|}})$ such that $y_j \in S'_j$, $y_{k_1} \in S'_{k_1}$, $y_{k_2} \in S'_{k_2}$, $\cdots$, and $y_{k_{|\Gamma'_j|}} \in S'_{k_{|\Gamma'_j|}}$, wherein $(\hat{y}_{k_1}, \hat{y}_{k_2}, \cdots, \hat{y}_{k_{|\Gamma'_j|}})$ are the $|\Gamma'_j|$ nodes connected to $\hat{y}_j$ by directed lines entering $\hat{y}_j$, wherein said $|\Gamma'_j|$ is an integer greater or equal to zero,

wherein if

$$P(x.) = \prod_{j=1}^{N} P_j[x_j|(x.)_{\Gamma_j}]/(\sum_{(x.)} numerator),$$

and

$$A(y.) = \prod_{j=1}^{N'} A_j[y_j|(y.)_{\Gamma'_j}]/(\sum_{(y.)} |numerator|^2),$$

and L is the set of all j such that $\hat{y}_j$ is a leaf node of said q-net data-set, and $not(L) = \{1, 2, \ldots N'\} - L$, and

$$A_L[(y.)_L] = \sum_{(y.)_{not(L)}} A(y.),$$

then, for all $x_1 \in S_1$, $x_2 \in S_2, \ldots$, and $x_N \in S_N$, $P(x.)$ is a sum of some numbers from the set

$$\{|A_L[(y.)_L]|^2 : \text{ for all possible values of } (y.)_L\}.$$

11. The method of claim 10, wherein said classical computer has a display screen, comprising the additional step of:

displaying on said display screen a diagram of said c-graph information.

12. The method of claim 10, comprising the additional step of:

    calculating with said classical computer, a data-set that specifies a unitary matrix U_{net}, wherein U_{net} describes the state evolution for the situation captured by said q-net data-set.

13. The method of claim 12, comprising the additional steps of:

    calculating with said classical computer, a data-set that specifies a unitary matrix T, wherein if $\Psi = U_{net}|0\rangle$, and if $\Psi' = T\Psi$, and if all the components Ψ_j of Ψ have an absolute value that is small compared to one, then some component Ψ'_{j_0} of Ψ' has an absolute value that is close to one.

    calculating with said classical computer, a sequence of operations, wherein said sequence of operations and said T both would, if applied to an array of qubits, produce equivalent transformations of the array.

14. The method of claim 13, wherein said sequence of operations comprises elementary operations on qubits.

15. The method of claim 13, wherein said sequence of operations is a sequence of elementary operations on qubits.

16. The method of claim 12, comprising the additional step of:

    calculating with said classical computer, a sequence of operations, wherein said sequence of operations and said U_{net} both would, if applied to an array of qubits, produce equivalent transformations of the array.

17. The method of claim 16, wherein said sequence of operations comprises elementary operations on qubits.

18. The method of claim 16, wherein said sequence of operations is a sequence of elementary operations on qubits.

19. The method of claim 16, further utilizing a quantum computer, comprising the additional step of:

manipulating said quantum computer largely according to said sequence of operations.

20. A method of operating a classical computer to calculate a q-net data-set based on a c-net data-set with the purpose of inducing a quantum computer to calculate a desired probability that depends on said c-net data-set, said method comprising the steps of:

storing said c-net data-set in said classical computer, wherein said c-net data-set comprises:

(a) c-graph information comprising a c-node label for each c-node of a plurality of N c-nodes, and for each $j \in \{1, 2, \cdots, N\}$, a subset Γ_j of $\{1, 2, \cdots, N\}$, and

(b) c-state information comprising, for each $j \in \{1, 2, \cdots N\}$, a finite set S_j containing labels for the states that the j'th c-node $\hat{x}_j$ may assume, and

(c) c-probability information comprising, for each $j \in \{1, 2, \cdots N\}$, a representation of a non-negative real number $P_j[x_j|(x_\cdot)_{\Gamma_j}]$ for each vector $(x_j, (x_\cdot)_{\Gamma_j}) = (x_j, x_{k_1}, x_{k_2}, \cdots, x_{k_{|\Gamma_j|}})$ such that $x_j \in S_j$, $x_{k_1} \in S_{k_1}$, $x_{k_2} \in S_{k_2}$, $\cdots$, and $x_{k_{|\Gamma_j|}} \in S_{k_{|\Gamma_j|}}$, wherein said $|\Gamma_j|$ is an integer greater or equal to zero,

composing said q-net data-set with said classical computer and using said c-net data-set, wherein said q-net data-set comprises

(a′) q-graph information comprising a q-node label for each q-node of a plurality of N' q-nodes, and for each $j \in \{1, 2, \cdots, N'\}$, a subset Γ'_j of $\{1, 2, \cdots, N'\}$, and

(b′) q-state information comprising, for each $j \in \{1, 2, \cdots N'\}$, a finite set S'_j containing labels for the states that the j'th q-node $\hat{y}_j$ may assume, and

(c′) q-amplitude information comprising, for each $j \in \{1, 2, \cdots N'\}$, a representation of a complex number $A_j[y_j|(y.)_{\Gamma'_j}]$for each vector $(y_j, (y.)_{\Gamma'_j}) = (y_j, y_{k_1}, y_{k_2}, \cdots, y_{k_{|\Gamma'_j|}})$ such that $y_j \in S'_j$, $y_{k_1} \in S'_{k_1}$, $y_{k_2} \in S'_{k_2}$, $\cdots$, and $y_{k_{|\Gamma'_j|}} \in S'_{k_{|\Gamma'_j|}}$, wherein said $|\Gamma'_j|$ is an integer greater or equal to zero,

wherein if

$$P(x.) = \prod_{j=1}^{N} P_j[x_j|(x.)_{\Gamma_j}]/(\sum_{(x.)} numerator),$$

and

$$A(y.) = \prod_{j=1}^{N'} A_j[y_j|(y.)_{\Gamma'_j}]/(\sum_{(y.)} |numerator|^2),$$

and $not(L) = \cup_{j=1}^{N'} \Gamma'_j$, and $L = \{1, 2, \ldots N'\} - not(L)$, and

$$A_L[(y.)_L] = \sum_{(y.)_{not(L)}} A(y.),$$

then, for all $x_1 \in S_1$, $x_2 \in S_2, \ldots$, and $x_N \in S_N$, $P(x.)$ is a sum of some numbers from the set

$$\{|A_L[(y.)_L]|^2 : \text{ for all possible values of } (y.)_L\}.$$

21. The method of claim 20, comprising the additional step of:

calculating with said classical computer, a data-set that specifies a unitary matrix U_{net}, wherein U_{net} describes the state evolution for the situation captured by said q-net data-set.

22. The method of claim 21, comprising the additional steps of:

calculating with said classical computer, a data-set that specifies a unitary matrix T, wherein if $\Psi = U_{net}|0\rangle$, and if $\Psi' = T\Psi$, and if all the components

Ψ_j of Ψ have an absolute value that is small compared to one, then some component Ψ'_{j_0} of Ψ' has an absolute value that is close to one.

calculating with said classical computer, a sequence of operations, wherein said sequence of operations and said T both would, if applied to an array of qubits, produce equivalent transformations of the array.

23. The method of claim 22, wherein said sequence of operations comprises elementary operations on qubits.

24. The method of claim 22, wherein said sequence of operations is a sequence of elementary operations on qubits.

25. The method of claim 21, comprising the additional step of:

calculating with said classical computer, a sequence of operations, wherein said sequence of operations and said U_{net} both would, if applied to an array of qubits, produce equivalent transformations of the array.

26. The method of claim 25, wherein said sequence of operations comprises elementary operations on qubits.

27. The method of claim 25, wherein said sequence of operations is a sequence of elementary operations on qubits.

28. The method of claim 25, further utilizing a quantum computer, comprising the additional step of:

manipulating said quantum computer largely according to said sequence of operations.

ABSTRACT

The invention comprises a classical computer that runs a special computer program. The program takes as input an initial data-set that comprises probabilistic information and returns as output a sequence of elementary operations (SEO). The initial data-set helps determine a classical Bayesian (CB) net. A program called "Q-Embedder" embeds the CB net within a quantum Bayesian (QB) net. A program called "Qubiter" (a quantum compiler) then translates the QB net into an equivalent SEO. The SEO outputted by the classical computer can be used to manipulate an array of qubits in a quantum computer. Application of the SEO to the array, followed by a measurement of the array, yields the value of certain conditional probabilities that we wish to know. The main goal of the invention is to provide a method for performing classical Bayesian net calculations on a quantum computer. Such calculations can be done on a classical computer; the hope is that they can be done much faster on a quantum computer.