Quantum Computing of Analogical Modeling of Language

Royal Skousen

Department of Linguistics
Brigham Young University
Provo, Utah 84602 USA
<royal_skousen@byu.edu>

Abstract

This paper discusses a general quantum algorithm that can be applied to any classical computer program. Each computational step is written using reversible operators, but the operators remain classical in that the qubits take on values of only zero and one. This classical restriction on the quantum states allows the copying of qubits, a necessary requirement for doing general classical computation. Parallel processing of the quantum algorithm proceeds because of the superpositioning of qubits, the only aspect of the algorithm that is strictly quantum mechanical. Measurement of the system collapses the superposition, leaving only one state that can be observed. In most instances, the loss of information as a result of measurement would be unacceptable. But the linguistically motivated theory of Analogical Modeling (AM) proposes that the probabilistic nature of language behavior can be accurately modeled in terms of the simultaneous analysis of all possible contexts (referred to as supracontexts) defined by a particular given context, providing one selects a single supracontext from those supracontexts that are homogeneous in behavior (namely, supracontexts that allow no increase in uncertainty). The amplitude for each homogeneous supracontext is proportional to its frequency of occurrence, with the result that the probability of selecting one particular supracontext to predict the behavior of the system is proportional to the square of its frequency.

The Quantum Mechanical Properties of Analogical Modeling

Analogical Modeling (AM) is a general theory for predicting behavior. It can also be considered a system of classification or categorization according to a particular set of outcomes. Predictions are directly based on a data set of exemplars. These exemplars give the outcome for various configurations of variables, which may be structured in different ways (such as strings or trees). A common method in AM is to define the variables so that there is no inherent structure or relationships between the variables (that is, each variable is defined independently of all the other variables). In this case, the variables can be considered a vector of features. In the data set, each feature vector is assigned an outcome vector. The data set is used to predict the outcome vector for a test set of various feature vectors for which no outcome vector has been assigned (or if one has been assigned, it is ignored). In AM the resulting predictions are not based on any learning stage for which the data set has been analyzed in advance in order to discover various kinds of potential relationships between the feature vectors and their associated outcome vectors. Neural nets, decision trees, and statistical analyses that determine the significance of the features in predicting the outcomes all rely on first learning something about the data set and then using that information to make predictions. AM, on the other hand, directly uses the data set to make a prediction for each specific feature vector in the test set.

Homogeneity versus Heterogeneity

The basic theory of AM was developed during 1979-1987 (see Skousen 1989 and Skousen 1992) and works from the hypothesis that in trying to predict the outcome (or behavior) for a vector of features, we consider all possible combinations of those features. Using a simple quadratic measure of uncertainty (not the traditional logarithmic one of information theory), we select those combinations of features that never permit any increase in the uncertainty. Those combinations that increase the uncertainty are referred to as heterogeneous and are eliminated from the analysis.

Another way to look at AM is to view each combination of features and its predicted outcome as a rule that maps from the feature combination to the outcome. The homogeneous combinations can be considered "true rules", the heterogeneous ones as "false rules". In other words, AM uses only the true rules; the false rules are ignored. Given that we have determined the true rules, the question then becomes: What are the chances of using a particular true rule to predict the outcome? The false rules, of course, are all assigned a probability of zero. Among the conceptual possibilities for assigning a probability to a true rule are the following: (1) each rule is equally probable; (2) the probability is proportional to the frequency of the rule in the data; (3) the

probability is proportional to the square of the frequency of the rule. Over time, it has become clear that the third choice is the simplest and most natural since it directly uses the same quadratic measure of uncertainty that is already needed to determine which rules are true (that is, which feature combinations are homogeneous in behavior). Moreover, the third choice has provided the most accurate results in predicting language behavior, including the appropriate degree of fuzziness that occurs at the boundaries of language behavior.

Linguistic Applications

AM has shown considerable success in explaining actual language behavior, beginning with Royal Skousen's description of the indefinite article in English (Skousen 1989:54-59 as well as Skousen 2003) and the past tense in Finnish (Skousen 1989:101-136), followed by Bruce Derwing and Royal Skousen's work on the past tense in English (Derwing and Skousen 1994), David Eddington on various problems in Spanish morphology (such as stress assignment in Eddington 2000a, diminutive formation in Eddington 2002a, gender assignment in Eddington 2002b, and nominalization in Eddington 2004:83-98), and Harald Baayen and his colleagues in the Netherlands on various aspects of Dutch morphology (see, for instance, Krott, Schreuder, and Baayen 2001 and Ernestus and Baayen 2003). Steve Chandler has provided a thorough comparison of AM with connectionist models of language as well as with a number of competing instance-based models. Chandler has shown how AM, a single-route approach to language description (that is, AM has a single conceptual mechanism), can readily handle various experimental results that were earlier claimed to be possible only in dual-route approaches to language; moreover, Chandler has found evidence from various psycholinguistic results that only AM seems capable of explaining (Chandler 2002). Similarly, Eddington 2000b has provided evidence that single-route exemplar-based approaches can accurately model speakers' abilities to predict both regular and irregular forms in language. For additional information about AM, see the Analogical Modeling website at <http://humanities.byu.edu/am/>.

One important aspect of AM is that we not restrict our analysis to just the important or crucial variables. We need to include "unimportant" variables in order to make our predictions robust. Consider, for example, the indefinite article *a*/*an* in English. Knowing that the following segment, whether consonant or vowel, "determines" the article (*a* for consonants, *an* for vowels), one could specify only the syllabicity of the following segment and thus predict *a*/*an* without error. Basically, we would be specifying a single rule analysis for the indefinite article. Yet in modeling the behavior of the indefinite article, AM specifies in addition the phonemic representation for that first segment in the following word as well as the phonemes and syllabicity for other segments in that word, supposedly unimportant variables. But by adding these other variables, AM is able to predict several behavioral properties of the indefinite article: (1) the one-way error tendency of adult speakers to replace *an* with *a* (but not *a* with *an*); (2) children's errors favoring the extension of *a*, but not *an*, such as "a upper", "a alligator", "a end", "a engine", "a egg", and "a other one"; (3) dialects for which *an* has been replaced by *a*, but not the other way around. In other words, the "unimportant" variables are crucial for predicting the fuzziness of actual language usage (for some discussion of these properties, see Skousen 2003). Finally, another important property is that AM can predict the indefinite article even when the first segment is obscured (that is, when one cannot tell whether that segment is a consonant or a vowel). In such cases, the other variables are used to guess the syllabicity of the obscured segment, thus allowing for the prediction. In other words, AM allows for robustness of prediction. If we assume a symbolic rule system with only one rule (one based on the syllabicity of the first segment), then no prediction is possible when that segment is obscured. For additional discussion of the robustness of AM with respect to the indefinite article, see Skousen 1989:58-59.

Specifying "unimportant" variables also allows for cases where the preferred analogy is not a nearest neighbor to the given context, but is found in a gang of homogeneous behavior at some distance from the given context. An important example of this occurs in predicting the past tense for the Finnish verb *sortaa* 'to oppress'. Standard rule analyses of Finnish as well as nearest neighbor approaches to language prediction argue that the past tense for this verb should be *sorsi*, whereas in fact it is *sorti*. Yet when AM is applied to predicting the past tense in Finnish, it is able to predict the correct *sorti*, mainly because AM indirectly discovers that the *o* vowel is the "crucial" variable in predicting the past tense for this verb. In previous analyses (typically based on the historically determined "crucial" variables), the *o* vowel was ignored. But AM, by specifying variables (both "important" and "unimportant") across the whole word, was able to make the correct prediction for this "exceptionally behaving" verb. For a complete discussion of how AM solves the problem of *sortaa*, see Skousen, Lonsdale, and Parkinson 2002:27-36.

Solving the Exponential Explosion

Early on it was realized that AM involved an exponential explosion. For every feature variable that was added to an analysis, there was basically a doubling in both the hardware requirements (memory) as well as the running time for a standard sequentially programmed computer to determine which feature combinations are homogeneous. In the simplest case, when there are n features all independent of each other, the total number of possible combinations that must be considered is 2^n. In 2000 it was discovered that the problem of exponentiality could be theoretically solved if AM were

treated as a quantum mechanical problem for which the following would hold:

(1) all possible rules exist in a superposition;
(2) the system evolves so that
    (a) the amplitude of every heterogeneous rule equals zero, and
    (b) the amplitude of each homogeneous rule equals its relative frequency of occurrence;
(3) measurement or observation reduces the superposition to a single rule, with the probability of it being selected equal to its amplitude squared (that is, equal to its relative frequency squared).

At that time I was able to demonstrate in general how classical reversible programming combined with the superpositioning of quantum mechanics could be used to solve this outstanding problem in AM exponentiality (see Skousen 2000). In this paper I will refer to the quantum mechanical reinterpretation of AM as QAM (namely, Quantum Analogical Modeling). In discussing QAM, the following properties will be noted:

(1) There is only one essential quantum concept in QAM: namely, the *superposition* of all possible rules; technically, the contextual specification for each rule is referred to as a supracontext (that is, we have a superposition of all the supracontexts).

(2) The actual operators used to control the evolution of the superposition are *classical reversible operators*.

(3) The operators must be written so they are applicable to every supracontext in the superposition; there is no conditional application, only *universal application.*

(4) Since the operators are classical reversible operators, the *only possible states* are 0 and 1. We avoid the full range of possibilities defined by generalized qubits; we use qubits in QAM, but we make sure that these qubits effectively take only the states 0 and 1.

(5) The result of using only the states 0 and 1 means that *qubits can be copied* and that fanout is therefore possible. Copying qubits allows us to replicate any possible classical reversible operator; on the other hand, copying is not possible for generalized qubits.

(6) Heterogeneity is a specific property of supracontexts that can be identified and derived by classical reversible operators; in QAM the superposition evolves so that we are able to identify the heterogeneous supracontexts and *set their amplitude to zero.*

(7) For each homogeneous supracontext, the final amplitude is *proportional to the frequency of occurrence* for that supracontext.

(8) From a rule perspective, quantum mechanical measurement or observation *collapses the superposition* of supracontexts (all possible rules) to a single supracontext (only one of the rules), which is then used to predict the outcome.

(9) From the exemplar perspective, measurement in QAM occurs by *randomly selecting a pointer* to any one of the data items in any occurring homogeneous supracontext; empty supracontexts have no data items and therefore no pointers; the pointers between the data items in heterogeneous supracontexts are effectively eliminated (that is, rendered unobservable).

(10) This one-stage system of measurement in exemplar-based QAM *avoids the traditional two-stage measurement system* in quantum mechanics, yet both give exactly the same result: (a) the amplitude is equal to the relative frequency of occurrence for every homogeneous supracontext, but equal to zero for every heterogeneous supracontext; (b) the amplitude squared gives the probability of using a particular supracontext to predict the outcome.

(11) The *directional pointers* connecting each pair of data items are used to determine the uncertainty and consequently the heterogeneity of any given supracontext in the superposition. These same pointers are used to observe the system; since the number of pointers is equal to the square of the number of data items, observation automatically leads to the squaring of the amplitude.

(12) QAM is *a general quantum computational algorithm* that categorizes and predicts behavior; it is based on the notion that supracontexts can be classified according to a specific definition of heterogeneity. Classical reversible programming can be used to identify other properties for superpositions of supracontexts, but whether other properties will serve useful roles in predicting behavior (or anything else) is an open issue.

Nielsen and Chuang (2000:39) warn that in measuring the superposition of 2^n cases, we can observe the information contained in only one of those cases:

> A significant caveat is that even though a quantum computer can simulate many quantum systems far more efficiently than a classical computer, this does not mean that the fast simulation will allow the desired information about the quantum system to be obtained. When measured, a kn qubit simulation will collapse into a definite state, giving only kn bits of information; the c^n bits of 'hidden information' in the wavefunction is not entirely accessible.

Analogical Modeling, it turns out, proposes such a collapse into a single definite state, yet that loss of information actually predicts how language behaves – and at the appropriate level of uncertainty.

A General Quantum Computing Algorithm for Analogical Modeling

Representations

In this first section, I list the symbols needed for representing AM and QAM. With respect to AM, I generally use the following indexing system:

- i, n from $i = 1$ to n, where n is the number of feature variables
- $\imath, \omega$ from $\imath = 1$ to ω, where ω is the number of outcome variables
- j, m from $j = 1$ to m, where m is the number of data items
- k, ℓ from $k = 1$ to ℓ, where ℓ is the number of bits needed for a variable specification

In addition, I is used to index the superposition of n independent feature variables, from $I = 1$ to 2^n. Indexing is used as a symbolic convention for representing the various bits and qubits. In other words, the indexes themselves (i, $\imath$, j, and k) are not actual variables; for instance, the statement "from $j = 1$ to m do A" simply means 'do A to each of the m data items'.

For QAM we have the following general notational system:

- $\mathbf{X}$ a qubit vector of length L
- X_i individual qubit representation, where X_i is the ith qubit in the qubit vector $\mathbf{X}$; thus $\mathbf{X} = \mathrm{X}_1\,\mathrm{X}_2\,\mathrm{X}_3 \cdots \mathrm{X}_L$

Qubit vectors are a sequence of qubits, with an assumed length of L (which can be different for various vectors). Qubit vectors are given in bold, such as $\mathbf{X}$. We use subscripts to stand as an index for the individual qubits in a qubit vector $\mathbf{X}$. For instance, if $\mathbf{X}$ had $L = 8$ qubits, we can represent each qubit as a nonbold form of $\mathbf{X}$ with an added subscript: that is, this particular qubit vector $\mathbf{X}$ can be represented as $\mathrm{X}_1\,\mathrm{X}_2\,\mathrm{X}_3\,\mathrm{X}_4\,\mathrm{X}_5\,\mathrm{X}_6\,\mathrm{X}_7\,\mathrm{X}_8$. Here each qubit X_i can theoretically take on all possible quantum state values. But in QAM, qubit vectors will be restricted to sequences of zeros and ones, such as $\mathbf{X} = 01011001$. Each individual qubit can be referred to by means of a subscript, such as $\mathrm{X}_1 = 0$ and $\mathrm{X}_8 = 1$.

- $\mathbf{X}^{[j]}$ the jth qubit vector $\mathbf{X}$

Sometimes we will refer to different qubit vectors that are all of the same type $\mathbf{X}$. In order to refer to such qubits individually, we will use superscripts in square brackets, such as $\mathbf{X}^{[j]}$ to stand for the jth qubit vector $\mathbf{X}$.

- $\mathbf{X}'$ the final evolved state of a qubit vector $\mathbf{X}$
- X'_i the final evolved state of the ith qubit in the qubit vector $\mathbf{X}$

The qubit $\mathbf{X}$ will evolve towards a resulting state that will then be copied (usually only one of its qubits) to some other qubit Y. We typically refer to the resulting state of $\mathbf{X}$ as $\mathbf{X}'$, an evolved qubit vector. When copying, we often copy only X_L', the evolved state of the last (or Lth) qubit in the qubit vector $\mathbf{X}$. After the copying, $\mathbf{X}'$ is typically returned to its original qubit vector $\mathbf{X}$ for subsequent processing of other qubits; this is accomplished by applying the operators in reverse order.

- $\mathbf{A}$ ... $\mathbf{T}$ substantive qubit vectors
- $\mathbf{U}$ ... $\mathbf{Z}$ auxiliary qubit vectors

Capital letters near the end of the alphabet will be reserved for the auxiliary qubits that are used to derive various results from substantive qubits or from other derived qubits that typically remain unchanged during the particular evolution under consideration. Capital letters near the end of the Greek alphabet may also be used to represent auxiliary qubits (especially for ones involving outcomes).

- $\mathbf{0}$ a qubit vector of zeros
- $\mathbf{1}$ a qubit vector of ones
- 0 a zero qubit
- 1 a one qubit

We use bold $\mathbf{0}$ and $\mathbf{1}$ to refer to qubit vectors of all zeros and all ones, but nonbold 0 and 1 to refer to individual qubits of zero or one. The initial state of a qubit vector $\mathbf{X}$ is generally set at specific values. If all the qubits are zeros, we represent that vector as $\mathbf{X} = \mathbf{0}$; if all the qubits are ones, we have $\mathbf{X} = \mathbf{1}$.

- $\mathbb{X}$ the superpositioned qubit vector $\mathbb{X}$
- $\mathbb{X}_i$ the ith qubit in the superpositioned qubit vector $\mathbb{X}$
- $\mathbb{X}^{[j]}$ the jth superpositioned qubit vector $\mathbb{X}$

When a qubit vector $\mathbf{X}$ is in superposition, we represent it as $\mathbb{X}$. Typographical outlining will be used to stand for the simultaneous multiple representation of superpositioning. We use a subscript i to represent the ith qubit in the superpositioned qubit vector $\mathbb{X}$ and the bracketed superscript j to stand for the jth superpositioned qubit vector $\mathbb{X}$.

- $\mathbb{0}$ a superpositioned qubit vector of zeros
- $\mathbb{1}$ a superpositioned qubit vector of ones

If superpositioning is involved, we may have a superpositioned qubit vector of all zeros ($\mathbb{0}$) or all ones ($\mathbb{1}$).

Fundamental Operators

Given this terminology, we can define the fundamental operators on qubits, qubit vectors, superpositioned qubits, and superpositioned qubit vectors:

for a qubit vector

NOT(**X**)
CNOT(**A**, **X**)
CCNOT(**A**, **B**, **X**)

for an individual qubit

NOT(X_i)
CNOT(A_j, X_i)
CCNOT(A_j, B_k, X_i)

for a superpositioned qubit vector

NOT($\mathbb{X}$)
CNOT($\mathbb{A}$, $\mathbb{X}$)
CCNOT($\mathbb{A}$, $\mathbb{B}$, $\mathbb{X}$)

for an individual qubit in a superpositioned qubit vector

NOT($\mathbb{X}_i$)
CNOT($\mathbb{A}_j$, $\mathbb{X}_i$)
CCNOT($\mathbb{A}_j$, $\mathbb{B}_k$, $\mathbb{X}_i$)

NOT(X_i) reverses the state of qubit X_i; CNOT(A_j, X_i) reverses the state of qubit X_i providing A_j equals 1; and CCNOT(A_j, B_k, X_i) reverses the state of X_i providing both A_j and B_k equal 1. In the same way, application of one of these operators to the qubit vector **X** means that every qubit X_i in **X** is reversed (under the appropriate controls). Similarly, application to the superpositioned qubit vector $\mathbb{X}$ means that every qubit in every superposition of **X** is reversed (again under the appropriate controls). On the other hand, application to $\mathbb{X}_i$ means that the operator applies to the qubit X_i (but only that qubit), yet in every superposition of $\mathbb{X}_i$.

By applying certain sequences of operators, we can create various derived operators. In general, we will use all caps to represent the name of such derived operators, just as we do with NOT, CNOT, and CCNOT. Within parentheses after the name of the operator we will specify the operands for the operator – namely, those (superpositioned) qubits and (superpositioned) qubit vectors that this operator will operate on. We might have something like OPERATOR (**P**, **Q**, **Z**). By reversing the order of its fundamental operators, we can reverse the effect of the derived operator and return to the original states for the appropriate qubits. We will use the superscript −1 with the name of the derived operator to indicate its reversal, such as OPERATOR^{-1}(**P**, **Q**, **Z**).

A General Summary

We may generalize the procedure used in QAM as follows. Given a set $\mathcal{S}$, we first assign instances to the members of $\mathcal{S}$ according to some qualifying characterization (in the case of QAM, it is contextual inclusion). The members of $\mathcal{S}$ form a superposition. While in superposition, we determine which members of $\mathcal{S}$ have a particular property $\mathcal{P}$. This property must be determined simultaneously but independently for each member in $\mathcal{S}$ and by using only classical reversible operators with qubits restricted to the states zero and one, thus allowing for copying of qubits. Each member of $\mathcal{S}$ with the property $\mathcal{P}$ is assigned an amplitude proportional to the frequency of the instances assigned to that member of $\mathcal{S}$, while all other members of $\mathcal{S}$ (those without the property $\mathcal{P}$) are assigned an amplitude of zero. Nonoccurring members of $\mathcal{S}$ (those for which no instances are assigned) will also have an amplitude of zero. Observation of the superposition proceeds in two stages. We first reduce the set $\mathcal{S}$ to a single member. The probability of selecting a member with the property $\mathcal{P}$ is proportional to the square of the frequency of the instances assigned to that member. Then, in order to predict an actual instance of that member, we randomly select one of the instances assigned to the member. In QAM, the set $\mathcal{S}$ is a power set and is in superposition; the members of the set are the supracontexts. The property $\mathcal{P}$ is homogeneity. For each member in $\mathcal{S}$, the property of homogeneity for each member in $\mathcal{S}$ can be determined in linear time (and with linear resources) – that is, without the exponential processing inherent in solving this problem using classical programming.

The General Algorithm

In the following, I provide the the general algorithm for QAM. Generally, *m* represents the number of data items in the data set, *n* the number of feature variables, and ω the number of outcome variables. The two derived operators ONES and INCLUSION are defined at the end of this summary. Here I do not show the derivation of the constant qubit vectors: the differences **D**, the feature variables **F**, and the outcome variables **Ω**; I will assume that they have already been provided for. (For a full presentation of how to set up these database vectors plus an example worked out in detail, see Skousen 2005.)

THE QUBITS AND THEIR INITIAL SETTINGS

controlling supracontextual superposition

$\mathbb{S}$: $\{0,1\}^n$

initial states for the substantive qubit vectors, with the number of qubits

$\mathbb{C} = \mathbb{0}$ *m* containment
$\mathbb{P} = \mathbb{1}$ *n* nonplurality of feature variables

$\mathbb{Q} = \mathbb{1}$ ω nonplurality of outcome variables
$\mathbb{M} = \mathbb{1}$ 1 plurality of subcontexts
$\mathbb{N} = \mathbb{1}$ 1 plurality of outcomes
$\mathbb{H} = \mathbb{1}$ 1 homogeneity
$\mathbb{A} = \mathbb{0}$ m amplitude

initial states for the auxiliary qubit vectors, with the number of qubits

$\mathbb{U} = \mathbb{0}$ n
$\mathbb{V} = \mathbb{0}$ n
$\mathbb{W} = \mathbb{1}$ n
$\mathbb{Z} = \mathbb{0}$ $n + 1$
$\mathbb{X} = \mathbb{1}$ m
$\mathbb{Y} = \mathbb{0}$ $m + 1$
$\Phi = \mathbb{0}$ ω
$\Psi = \mathbb{0}$ ω
$\mathbb{Y} = \mathbb{0}$ ω + 1

THE EVOLUTION OF THE SYSTEM

determining which data items are contained in $\mathbb{S}$

from j = 1 to m do

INCLUSION($\mathbb{S}$, $\mathbf{D}^{[j]}$, $\mathbb{W}$, $\mathbb{Z}$)
CNOT($\mathbb{Z}_n$, $\mathbb{C}$)
INCLUSION^{-1}($\mathbb{S}$, $\mathbf{D}^{[j]}$, $\mathbb{W}$, $\mathbb{Z}$)

determining the plurality of the feature variables in $\mathbb{S}$

from i = 1 to n do

CCNOT($\mathbb{C}$, $\mathbf{F}^{[i]}$, $\mathbb{X}$)
ONES($\mathbb{X}$, $\mathbb{Y}$)
CNOT($\mathbb{Y}_m$, $\mathbb{U}_i$)
ONES^{-1}($\mathbb{X}$, $\mathbb{Y}$)
CCNOT($\mathbb{C}$, $\mathbf{F}^{[i]}$, $\mathbb{X}$)
NOT($\mathbf{F}^{[i]}$)
CCNOT($\mathbb{C}$, $\mathbf{F}^{[i]}$, $\mathbb{X}$)
ONES($\mathbb{X}$, $\mathbb{Y}$)
CNOT($\mathbb{Y}_m$, $\mathbb{V}_i$)
ONES^{-1}($\mathbb{X}$, $\mathbb{Y}$)
CCNOT($\mathbb{C}$, $\mathbf{F}^{[i]}$, $\mathbb{X}$)
NOT($\mathbf{F}^{[i]}$)
NOT($\mathbb{U}_i$)
NOT($\mathbb{V}_i$)
CCNOT($\mathbb{U}_i$, $\mathbb{V}_i$, $\mathbb{P}_i$)

determining the plurality of the outcome variables in $\mathbb{S}$

from ι = 1 to ω do

CCNOT($\mathbb{C}$, $\mathbf{\Omega}^{[\iota]}$, $\mathbb{X}$)
ONES($\mathbb{X}$, $\mathbb{Y}$)
CNOT($\mathbb{Y}_m$, Φ_ι)
ONES^{-1}($\mathbb{X}$, $\mathbb{Y}$)
CCNOT($\mathbb{C}$, $\mathbf{\Omega}^{[\iota]}$, $\mathbb{X}$)
NOT($\mathbf{\Omega}^{[\iota]}$)
CCNOT($\mathbb{C}$, $\mathbf{\Omega}^{[\iota]}$, $\mathbb{X}$)
ONES($\mathbb{X}$, $\mathbb{Y}$)
CNOT($\mathbb{Y}_m$, Ψ_ι)
ONES^{-1}($\mathbb{X}$, $\mathbb{Y}$)
CCNOT($\mathbb{C}$, $\mathbf{\Omega}^{[\iota]}$, $\mathbb{X}$)
NOT($\mathbf{\Omega}^{[\iota]}$)
NOT(Φ_ι)
NOT(Ψ_ι)
CCNOT(Φ_ι, Ψ_ι, $\mathbb{Q}_\iota$)

determining the plurality of the subcontexts in $\mathbb{S}$

ONES($\mathbb{P}$, $\mathbb{Z}$)
CNOT($\mathbb{Z}_n$, $\mathbb{M}$)
ONES^{-1}($\mathbb{P}$, $\mathbb{Z}$)

determining the plurality of the outcomes in $\mathbb{S}$

ONES($\mathbb{Q}$, $\mathbb{Y}$)
CNOT($\mathbb{Y}_\omega$, $\mathbb{N}$)
ONES^{-1}($\mathbb{Q}$, $\mathbb{Y}$)

determining the homogeneity in $\mathbb{S}$

CCNOT($\mathbb{M}$, $\mathbb{N}$, $\mathbb{H}$)

determining the amplitude in $\mathbb{S}$

from j = 1 to m do
CCNOT($\mathbb{C}_j$, $\mathbb{H}$, $\mathbb{A}_j$)

OBSERVATION (using pointers)

Randomly select a pointer to any data item in the (superpositioned) amplitude $\mathbb{A}$; the predicted behavior is the outcome associated with that data item.

DERIVED OPERATORS

ONES($\mathbb{X}$, $\mathbb{Y} = \mathbb{0}$)

NOT($\mathbb{Y}_0$)
from k = 1 to L do
CCNOT($\mathbb{X}_k$, $\mathbb{Y}_{k-1}$, $\mathbb{Y}_k$)

[with $\mathbb{X}$ of length L and $\mathbb{Y}$ of length L + 1]

INCLUSION($\mathbb{S}$, $\mathbf{D}$, $\mathbb{W} = \mathbb{1}$, $\mathbb{Z} = \mathbb{0}$)

CCNOT($\mathbb{S}$, $\mathbf{D}$, $\mathbb{W} = \mathbb{1}$)
ONES($\mathbb{W}$, $\mathbb{Z} = \mathbb{0}$)

Acknowledgments

I wish to thank my colleagues Deryle Lonsdale and Don Chapman in the Analogical Modeling Research Group for providing useful critiques of this paper.

References

Chandler, Steve (2002). "Skousen's Analogical Approach as an Exemplar-Based Model of Categorization", in Skousen, Lonsdale, and Parkinson (2002: 51-105).

Derwing, Bruce, and Royal Skousen (1994). "Productivity and the English Past Tense: Testing Skousen's Analogy Model", *The Reality of Linguistic Rules,* edited by Susan D. Lima, Roberta L. Corrigan, and Gregory K. Iverson (Amsterdam: John Benjamins), 193-218.

Eddington, David (2000a). "Spanish Stress Assignment within Analogical Modeling of Language", *Language* 76:92-109.

Eddington, David (2000b). "Analogy and the Dual-Route Model of Morphology", *Lingua* 110:281-298.

Eddington, David (2002a). "Spanish Diminutive Formation without Rules or Constraints", *Linguistics* 40:395-419.

Eddington, David (2002b). "Spanish Gender Assignment in an Analogical Framework", *Journal of Quantitative Linguistics* 9:49-75.

Eddington, David (2004). *Spanish Phonology and Morphology* (Amsterdam: John Benjamins).

Ernestus, Mirjam, and R. Harald Baayen (2003). "Predicting the Unpredictable: Interpreting Neutralized Segments in Dutch", *Language* 79:5-38.

Krott, Andrea, Robert Schreuder, and R. Harald Baayen (2001). "Analogy in Morphology: Modeling the Choice of Linking Morphemes in Dutch", *Linguistics* 39:51-93.

Nielsen, Michael A., and Isaac L. Chuang (2000). *Quantum Computation and Quantum Information* (Cambridge: Cambridge University Press).

Skousen, Royal (1989). *Analogical Modeling of Language* (Dordrecht, The Netherlands: Kluwer).

Skousen, Royal (1992). *Analogy and Structure* (Dordrecht, The Netherlands: Kluwer).

Skousen, Royal (2000). "Analogical Modeling and Quantum Computing" <http://www.arXiv.org> [published in Skousen, Lonsdale, and Parkinson 2002: 319-346].

Skousen, Royal, Deryle Lonsdale, and Dilworth B. Parkinson, editors (2002). *Analogical Modeling: An Exemplar-Based Approach to Language* (Amsterdam: John Benjamins).

Skousen, Royal (2003). "Analogical Modeling: Exemplars, Rules, and Quantum Computing", *Proceedings of the Twenty-Ninth Annual Meeting of the Berkeley Linguistics Society*, edited by Pawel Nowak, Corey Yoquelet, and David Mortensen (Berkeley, California: Berkeley Linguistics Society), 425-439 [paper also available at <http://humanities.byu.edu/am/>].

Skousen, Royal (2005). "Quantum Analogical Modeling: A General Quantum Computing Algorithm for Predicting Language Behavior" <http://www.arXiv.org>.