Weierstraß-Institut
für Angewandte Analysis und Stochastik
Leibniz-Institut im Forschungsverbund Berlin e. V.

Preprint ISSN 0946 – 8633

Probing nonlinear adiabatic paths with a universal integrator

Michael Hofmann[1,2], Gernot Schaller[2]

submitted: November 18, 2013

[1] Weierstrass Institute
Mohrenstr. 39
10117 Berlin
Germany
E-Mail: michael.hofmann@wias-berlin.de

[2] Technische Universität Berlin
Hardenbergstraße 36
10623 Berlin
Germany
E-Mail: gernot.schaller@tu-berlin.de

No. 1872
Berlin 2013

2010 *Physics and Astronomy Classification Scheme.* 03.67.Ac, 75.10.Nr, 75.10.Dg, 02.60.-x.

Key words and phrases. Quantum algorithms and protocols, Spin Hamiltonians, Numerical methods.

The authors have profited from discussions with C. Schröder, U. Kandler, R. Okuyama, T. Brandes, V. Mehrmann, R. Schützhold, and F. Renzoni. Financial support by the DFG (BRA-1528/7-1, SCHA 1646/2-1) is also gratefully acknowledged.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Abstract

We apply a flexible numerical integrator to the simulation of adiabatic quantum computation with nonlinear paths. We find that a nonlinear path may significantly improve the performance of adiabatic algorithms versus the conventional straight-line interpolations. The employed integrator is suitable for solving the time-dependent Schrödinger equation for any qubit Hamiltonian. Its flexible storage format significantly reduces cost for storage and matrix-vector multiplication in comparison to common sparse matrix schemes.

Simulating quantum systems requires enormous computational resources: Even for a few hundred particles there would be more variables to be stored than atoms exist in the universe [1]. To turn this problem into an advantage, quantum computers may be efficiently used for such simulations, since they are quantum systems themselves [2]. Moreover, quantum algorithms can solve distinct problems like number factoring with exponential speedup compared to classical computers [3].

In the conventional picture, quantum algorithms are implemented as a sequence of unitary operations [4], which implies fast switching of the generating Hamiltonian. In contrast, within the paradigm of adiabatic quantum computation [5], the Hamiltonian is modified slowly from a simple initial Hamiltonian with an easy-to-prepare ground state to a final Hamiltonian which encodes in its ground state the solution to some difficult problem. Most importantly, for a large class of problems, implementation of the final Hamiltonian is possible without knowing the solution of the problem explicitly. The adiabatic theorem implies – provided the evolution is slow enough – that the system will end up near the ground state of the final Hamiltonian, such that the solution to the problem can be obtained by measuring the system. The evolution time is related to the spectral properties of the time-dependent Hamiltonian and thus corresponds to the algorithmic complexity of an adiabatic quantum algorithm (AQA). The conventional circuit picture and the adiabatic approach are known to be polynomially equivalent [6, 7], but exact results for adiabatic algorithms are scarce [8]. It is therefore quite interesting that first numerical simulations of the Schrödinger equation revealed a seemingly polynomial complexity of the adiabatic algorithm for an NP-complete problem [5]. Since then, it has been a strongly debated question whether this scaling would persist for larger problem sizes [9, 10, 11, 12, 13, 14]. Recent findings suggest that the scaling complexity of the conventional straight-line adiabatic interpolation is typically exponential [15, 16]. It may however be conjectured that with modifications of the adiabatic algorithm, its scaling behavior can be considerably improved [17], such that the scaling behavior of adapted algorithms is still an open question.

Unfortunately, this question can currently not be settled from the experimental side: Though enormous progress has been made in the last decade, not more than a few quantum bits (qubits) have been entangled so far [18], which currently restricts the execution of quantum algorithms to proof-of-principle demonstrations. As experiments are still neither flexible nor scalable enough to investigate new theoretical models, the demand for classical computer simulations of quantum algorithms is growing. Such simulations are computationally expensive and usually must be

coded separately for each problem considered. Here, we use an efficient numerical integrator to solve the time-dependent Schrödinger equation for the high-dimensional but sparse Hamiltonians typical for qubit systems. An adopted storage format will reduce the memory required for storing the Hamiltonian in comparison to common sparse matrix schemes while keeping their advantage of fast matrix-vector multiplication. In particular its ability to follow flexible adiabatic paths renders our storage scheme suitable for such simulations.

The paper is structured as follows: In Sec. 1, we expose the prerequisites discussing the data storage scheme, adiabatic computation, and the particular NP-complete problem considered. Afterwards, we numerically compare the performance of different adiabatic quantum algorithms (AQAs) for straight-line interpolation in Sec. 2. Then, we turn to the investigation of non-linear paths in Sec. 3 and close with conclusions.

1 Theory

1.1 Sparse Quantum Hamiltonian (SQH)

A single-qubit state is a superposition of two fundamental states denoted by $|0\rangle$ and $|1\rangle$, which form the computational basis. As a convention, those states are the eigenstates of the Pauli matrix σ^z: $|0\rangle := \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right),\ |1\rangle := \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right)$. Similarly, the basis states for an n-qubit system can be constructed by the tensor product

$$|z\rangle = \bigotimes_{i=1}^{n} |z_i\rangle\,,\ z_i \in \{0,1\}. \tag{1}$$

Here z is the decimal representation $z = \sum_{i=1}^{n} z_i\, 2^{i-1}$ of the bitstring $z_n\, z_{n-1} \ldots z_2\, z_1$. An arbitrary n-qubit state is then given by the superposition

$$|\psi\rangle = \sum_{z=0}^{2^n-1} \alpha_z\, |z\rangle\,,\ \alpha_z \in \mathbb{C} \tag{2}$$

with normalization condition $\sum_z |\alpha_z|^2 = 1$. Obviously, the dimension of the Hilbert space, $N = 2^n$, is growing exponentially with the number of qubits which makes simulations of quantum systems hard.

A Hamiltonian acting on n qubits can be described by the Pauli matrices $\sigma^x, \sigma^y, \sigma^z$: Together with the identity $\sigma^0 := \mathbb{1}$ they span the space of all 2×2-matrices. Using the n-fold Kronecker product of those matrices yields $N^2 = 2^{2n}$ generalized Pauli matrices (GPMs) $\mathcal{S}_i = \bigotimes_{q=1}^{n} \sigma^{\alpha_{i,q}}$ as a basis for all $N \times N$-matrices. Trace-orthogonality ensures that any Hamiltonian can be decomposed into GPMs,

$$\mathcal{H} = \sum_{i=0}^{N^2-1} m_i\, \mathcal{S}_i \quad \text{with} \quad m_i = \frac{1}{N}\mathrm{Tr}\{\mathcal{H}\,\mathcal{S}_i\} \in \mathbb{R}. \tag{3}$$

Let σ_i^α be the short notation of the tensor product

$$\underbrace{\mathbb{1} \otimes \ldots \otimes \mathbb{1}}_{i-1} \otimes \sigma^\alpha \otimes \underbrace{\mathbb{1} \otimes \ldots \otimes \mathbb{1}}_{n-i}, \qquad \alpha \in \{x, y, z\}, \tag{4}$$

acting only on the i-th qubit. A GPM of order j (acting non-trivially on j qubits) is then written as $\sigma_{i_1}^{\alpha_1}\sigma_{i_2}^{\alpha_2}\ldots\sigma_{i_j}^{\alpha_j}$. By counting only non-vanishing terms $m_i \neq 0$, Eq. (3) has the more convenient form

$$\mathcal{H} = m^{(0)}\mathbb{1} + \sum_{j=1}^{p}\sum_{l=1}^{k_j} m_l^{(j)} \prod_{q=1}^{j} \sigma_{i_{l,q}^{(j)}}^{\alpha_{l,q}^{(j)}} , \tag{5}$$

where p denotes the order of the Hamiltonian, k_j the number of j-local terms, $m_l^{(j)}$ the corresponding real prefactor, and $m^{(0)}$ the energy shift.

We now introduce a *Sparse Quantum Hamiltonian (SQH) format*: For a complete description of the Hamiltonian's structure, we do not need to store the full GPMs but only the parameters used in Eq. (5), including the positions i and types α of the (single) Pauli matrices. Consequently, storage of a Hamiltonian is efficient if its order is independent of the number of qubits, $p \ll n$, and becomes even more favorable when the number of terms for every order j is small, e.g., $k_j \sim \mathcal{O}(n)$. An example of such a system is the quantum Ising model in a transverse field,

$$\mathcal{H} = -\Omega(1-s)\sum_{i}^{n}\sigma_i^x - \Omega s \sum_{i}^{n}\sigma_i^z\sigma_{i+1}^z , \tag{6}$$

where Ω represents an energy scale and s a control parameter and where periodic boundary conditions are assumed $\sigma_{n+1}^z = \sigma_1^z$. This Hamiltonian would only require $\mathcal{O}(n)$ elements to store in SQH format.

A matrix-vector product $\mathcal{H}\left|\psi\right\rangle$ is according to Eq. (5) reduced to a sum of GPMs acting on a basis state, $\sigma_{i_1}^{\alpha_1}\ldots\sigma_{i_j}^{\alpha_j}\left|z\right\rangle$. Due to the tensor structure of z (c.f. Eq. (1)), such a multi-qubit operation is broken down to a multiplication of successive single-qubit operations $\sigma_i^\alpha\left|z_i\right\rangle$. Assuming the number of terms in $\mathcal{H}$ is $\mathcal{O}(n)$, the effort for computing $\mathcal{H}\left|\psi\right\rangle$ scales as $N \times \mathcal{O}(n)$ instead of N^2 when the Hamiltonian was stored conventionally.

The universal applicability of the SQH representation allows to write program code (e.g., an integrator) independent of the used Hamiltonian as long as it has the SQH structure given in Eq. (5). Also time-dependent Hamiltonians can be implemented by time-dependent coefficients $m_l^{(j)}(t)$.

1.2 Adiabatic Quantum Computation

Quantum computation by adiabatic evolution has been suggested as a promising approach for solving NP-complete problems [5]. The idea is simple: A final Hamiltonian $\mathcal{H}_f$ is constructed which encodes the solution for a computational problem in its ground state, e.g., as an energy penalty function. We stress here that for a number of hard problems this can be done without knowing the solution. Starting from an easy to construct ground state of an initial Hamiltonian $\mathcal{H}_i$, the system is transformed to $\mathcal{H}_f$ after the runtime T. A common approach is to use a linear interpolation:

$$\mathcal{H}(s) = (1-s)\mathcal{H}_i + s\mathcal{H}_f \tag{7}$$

with constant velocity $s(t) = \frac{t}{T}$ and $0 \leq t \leq T$. If this transformation is slow enough, the adiabatic theorem guarantees that the system remains always near the instantaneous ground state [19] such that a measurement after time T yields the solution. This works in principle with every energy eigenstate of the system, but by using the ground state one hopes that the

evolution is dissipation-free (which becomes relevant when the system is coupled to a low-temperature reservoir [20]). The runtime T thus is a measure for the algorithmic complexity. It can be optimized by adapting the speed of the interpolation $\dot{s}(t)$ to the energy gap above the ground state [8, 21, 22]. In this case however, the time-dependent Hamiltonian is still a convex combination of initial and final Hamiltonian (hence the terminology straight-line interpolations), and we will not consider such extensions here.

To study the efficiency of such an algorithm, i.e., the runtime scaling with increasing system size n, we need to determine an adiabatic runtime $T_s(n)$ by simulating the evolution of a system prepared in the initial ground state. It is governed by the Schrödinger equation

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t} |\psi(t)\rangle = \mathcal{H}(t) |\psi(t)\rangle \,, \tag{8}$$

which for the expansion in Eq. (2) becomes a whole set of N coupled ordinary differential equations with initial condition $|\psi(0)\rangle = |\psi_0\rangle$. A fast numerical integration is achieved by using the SQH format for $\mathcal{H}(t)$ and a fourth-order predictor-corrector scheme [23], which requires only a single evaluation of $\mathcal{H} |\psi\rangle$ per integration step.

1.3 3-Bit Exact Cover

A common problem for probing adiabatic quantum algorithms (AQAs) is 3-bit exact cover (EC3), which is NP-complete. In a nutshell, solutions to problems in the class NP can be verified in a time that is polynomial in the length of their input. The completeness property in addition implies that every other problem in NP can be mapped to EC3 with polynomial overhead only. On an n-bitstring $z_n\, z_{n-1} \ldots z_2\, z_1$ we define an instance of EC3 as a set of m different clauses c_i each involving three different bits

$$c_i = (c_i^{(1)}, c_i^{(2)}, c_i^{(3)}), \qquad c_i^{(k)} \in \{1 \ldots n\} \tag{9}$$

with $1 \leq i \leq m$. A clause is satisfied, if and only if one of the involved bits z_i equals 1, i.e., when

$$z_{c_i^{(1)}} + z_{c_i^{(2)}} + z_{c_i^{(3)}} = 1 \,, \tag{10}$$

where '$+$' denotes the ordinary integer sum. A solution to an instance is a bitstring satisfying all clauses in the set, which is easy to check. In contrast, finding such a bitstring is a combinatorial search problem for which no efficient classical algorithm is known. Figure 1 visualizes an EC3 instance for 13 bits with a unique solution.

For an AQA the problem is encoded as a cost function, where each unsatisfied clause adds an energy penalty to the Hamiltonian [24],

$$h_i = \left[\mathbb{1} - \frac{1}{2} \sum_k^3 \left(\mathbb{1} - \sigma^z_{c_i^{(k)}} \right) \right]^2 . \tag{11}$$

The final Hamiltonian $\mathcal{H}_f$ is simply constructed as a sum over all clauses and can be simplified to [25]

$$\mathcal{H}_f = \sum_{i=1}^m \Omega h_i = \Omega m \mathbb{1} - \Omega \sum_i^n \frac{n_i}{2} \sigma_i^z + \Omega \sum_{i<j}^n \frac{n_{i,j}}{2} \sigma_i^z \sigma_j^z . \tag{12}$$

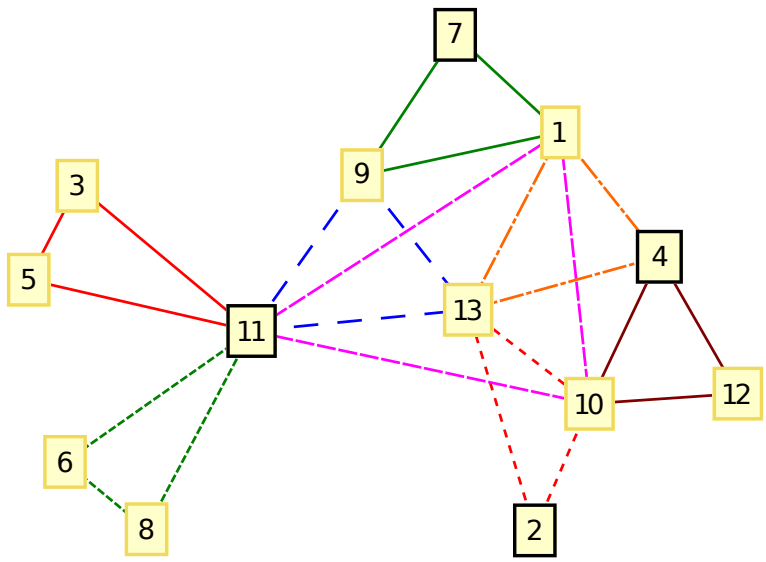

Figure 1: Instance of EC3, where the labeled vertices represent the 13 bits z_i, edges of same color and line style represent a clause c_i. Bold-bordered vertices indicate the upbits in the solution $z = 1098 = (0010001001010)$.

Here, $\Omega > 0$ just denotes an energy scale, the coefficient n_i denotes the number of clauses involving the i-th qubit, and $n_{i,j}$ is the number of clauses, which contain the i-th and j-th qubit. For example, in Fig. 1 we have $n_{11} = 4$ and $n_{11,13} = 1$. The Hamiltonian $\mathcal{H}_f$ corresponds to a frustrated antiferromagnet in a non-uniform magnetic field (with an energy shift m) [25]. The coupling strength between the spins, however, is not defined by the experimental geometry (e.g., only between nearest neighbors) but by the edges of the clauses $n_{i,j}$, which may define a highly disordered network.

1.4 Hard Instances

To provide statistical evidence for our simulations, we generate for each system size 100 hard instances. These were characterized by a unique solution, a number of clauses close to the classical EC3 phase transition $m \approx \frac{2}{3}n$ from satisfiable to unsatisfiable problems [26], and the constraint that $n_{ij} \in \{0, 1\}$. The last constraint implies that clauses should only share corners and not edges. It is motivated by the fact that in case an edge is shared by two clauses, one may easily conclude that in the solution, the opposite corners must have the same value. Effectively, clauses that share edges would thus reduce the size of the problem. Altogether, these constraints lead to $\mathcal{O}(n)$ terms in Eq. (12), and $\mathcal{H}_f$ is efficiently stored in SQH format.

The recipe for generating a random (hard) instance is shown in figure 2. We start with a full pool of all possible clauses. First, only clauses which intersect with the existing graph are added to ensure connectivity. When all bits are connected to the graph, the number of solutions is checked each time after a clause is drawn. A clause is discarded, if it reduces the number of solutions to zero. The algorithm finishes if only one solution is left. If no such unique satisfying assignment (USA) has been found but all clauses are dropped from the pool *or* the number of allowed clauses m is reached, the algorithm restarts itself.

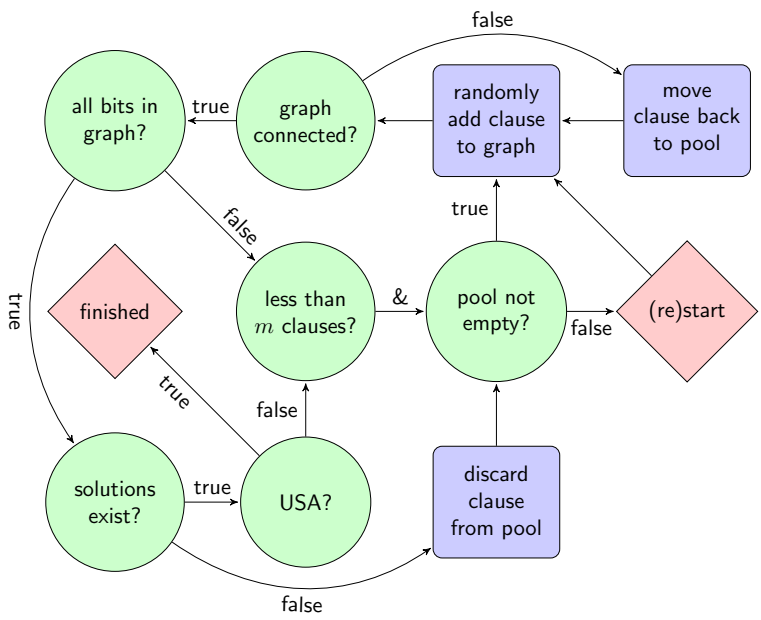

Figure 2: Schema of our algorithm for randomly generating hard instances of EC3. Rectangles and circles indicate instructions and if-clauses, respectively. The algorithm starts and stops at diamonds.

2 Straight Line Interpolation

In this section, we simulate and compare the results of three AQAs defined by different initial Hamiltonians $\mathcal{H}_i$. The interpolation path between initial and final Hamiltonian is a straight line given by Eq. (7), traversed at constant speed $s(t) = t/T$. As a benchmark, we compare with the analytically solvable Ising model in Eq. (6), which exhibits an inverse $1/n$-scaling of the minimum energy gap between ground and first coupled excited state, leading to a quadratic scaling of the adiabatic runtime [27]. We begin by summarizing the explored algorithms.

2.1 Algorithms

2.1.1 X-Algorithm

The original approach [5] used a single-qubit structure

$$\mathcal{H}_i = \mathcal{H}^x = \Omega \sum_i^n \frac{n_i}{2}(\mathbb{1} - \sigma_i^x) \tag{13}$$

with the ground state

$$|S\rangle = \frac{1}{\sqrt{2^n}} \sum_{z=0}^{N-1} |z\rangle \,. \tag{14}$$

However, this does not reflect the interaction topology of the final Hamiltonian (12), i.e., the coefficient n_i does not carry the full information about the clauses c_i.

2.1.2 XYZ-Algorithm

A choice with two-qubit interactions is the Heisenberg ferromagnet [25, 14]

$$\mathcal{H}_i = \mathcal{H}^{xyz} = \Omega \sum_{i<j}^{n} \frac{n_{i,j}}{2}(\mathbb{1} - \boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_j). \tag{15}$$

Both $\mathcal{H}^{xyz}$ and $\mathcal{H}_f$ are invariant under rotations around the Σ^z-axis,

$$\Sigma^z = \sum_{i}^{n} \frac{1}{2}(\mathbb{1} - \sigma_i^z). \tag{16}$$

The eigenvalue Δ of Σ^z is directly related to the number of 1-bits in the solution (also denoted as Hamming weight Δ). It is therefore a constant of motion and conserved during dynamics. Only the subspace $|\psi\rangle : \Sigma^z |\psi\rangle = \Delta_w |\psi\rangle$ with the fixed Hamming weight Δ_w of the solution has to be considered here, as all subspaces with different Hamming weights evolve independently. The ground state of $\mathcal{H}^{xyz}$ in the appropriate subspace is given by a balanced superposition over all basis states $|u\rangle$ with $\Sigma^z |u\rangle = \Delta_w |u\rangle$,

$$|\psi_0\rangle^{xyz} = \binom{n}{\Delta_w}^{-\frac{1}{2}} \sum_{u} |u\rangle \,. \tag{17}$$

which can be prepared efficiently [28] by adiabatic evolution $\mathcal{H}^x \xrightarrow{T} \mathcal{H}^P$. In that reference, the final Hamiltonian $\mathcal{H}^P$ used an energy penalty to separate the subspaces: It was given by $\mathcal{H}^P = (\Sigma^z - \Delta_w)^2$, which has highly degenerate energy levels, but due to symmetry arguments the correct angular momentum eigenstates were selected. In our numerical considerations we circumvent this preparation step and directly prepare the initial state (17), such that the adiabatic algorithm only consists in a deformation of $\mathcal{H}^{xyz}$ to the final problem Hamiltonian (12).

Realistically, the solution $|w\rangle$ and thus the Hamming weight Δ_w would not be known in advance, which would make repeated runs of the AQA in different subspaces necessary. But even in the worst case, when every possible value of $\Delta \in \{0 \dots n\}$ has to be tried, the computational overhead scales only linearly in n.

2.1.3 XY-Algorithm

Similar to the previous example is the x,y-ferromagnet

$$\mathcal{H}_i = \mathcal{H}^{xy} = 3m\Omega\mathbb{1} - \Omega \sum_{i<j}^{n} \frac{n_{i,j}}{2}(\sigma_i^x \sigma_j^x + \sigma_i^y \sigma_j^y). \tag{18}$$

It has already been shown numerically that it yields on average a better performance than $\mathcal{H}^{xyz}$ on a common instance of EC3 with a unique solution [25]. Again, the Hamming weight is a constant of motion and we only consider the corresponding subspace. The ground state of (18) is analytically unknown but can be initialized by adiabatic evolution $\mathcal{H}^x \xrightarrow{T} \mathcal{H}^{xy}$. Using an ARPACK eigensolver [29, 30] accepting SQH as input variable, we found numerically for the examples considered that the minimum gap during the initial preparation is roughly independent of the system size. We expect therefore only a mild algorithmic scaling of this preparation step with the system size n, which would enable an efficient adiabatic preparation of the ground state of Eq. (18).

2.2 Results

The probability to find the system in the solution state $|w\rangle$ after the runtime T is $P_1(T) = |\langle\psi(T)|w\rangle|^2$ and would ideally approach one. However, to avoid too long computation times but simultaneously ensure a high fidelity, we define a successful runtime T_s by measuring the system's energy $E(T_s)/\Omega = \frac{1}{2}$. Here, we exploit the fact, that any excited state raises the energy by an amount greater or equal Ω, cf. Eq. (12). The first excited state of $\mathcal{H}_f$ therefore has an energy $E_2 \geq \Omega$. The energy is thus lower bounded by (using $E_1(T) = 0$)

$$\begin{aligned} E(T) &= \sum_{n=1}^{N} E_n(T)P_n(T) \geq E_2(P_2 + \ldots + P_N) \\ &\geq \Omega(1 - P_1)\,, \end{aligned} \tag{19}$$

and defining this criterion as a measure for a successful runtime means that the ground state occupation is at least one-half.

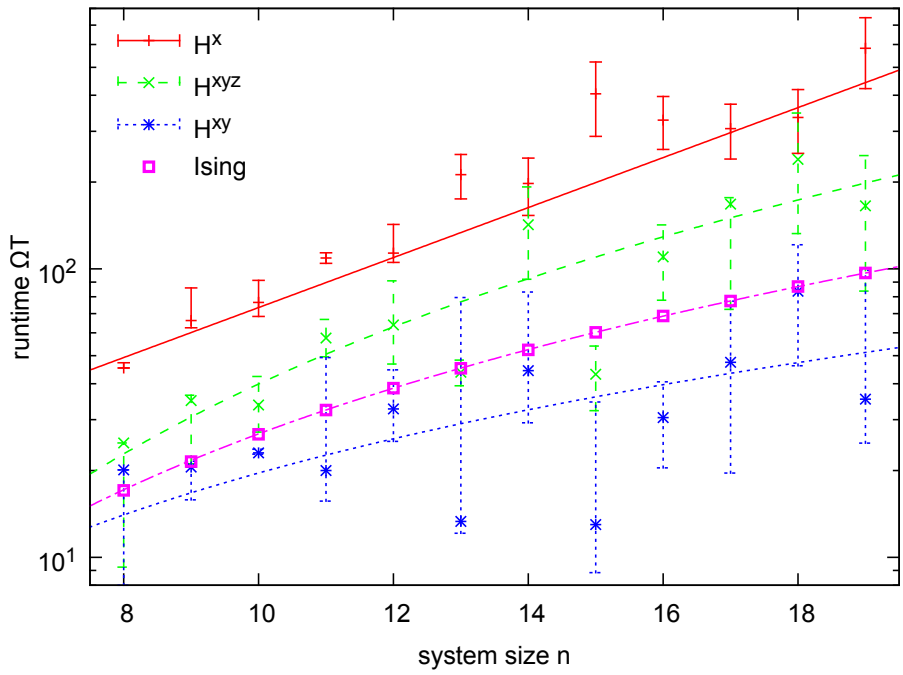

Figure 3: Scaling of the dimensionless runtime $\Omega T_s(n)$ (symbols, fit lines only serve to guide the eye) for algorithms X (solid), XYZ (long-dashed), XY (dotted) compared with the quadratically scaling Ising model (dash-dotted). Symbols represent the median and error bars the first and third quartile out of 100 randomly chosen hard instances. Although the scaling behavior is inconclusive, it is evident that different initial states may drastically improve the performance of the AQA.

Figure 3 shows computed median runtimes $T_s(n)$ for the three different algorithms compared with an adiabatic version of the Ising model, Eq. (6), traversed at constant interpolation speed $s(t) = t/T$. The latter serves as a benchmark with a runtime scaling known to be quadratically [27]. Our results show, that the XY-algorithm is the fastest followed by the XYZ-algorithm. Both stay close to the Ising curve which would indicate a polynomial scaling in the observed region. A clear statement however is hampered by the large deviations and fitting remains ambiguous.

In contrast to previous studies [5], where the runtime of the X-algorithm appeared polynomial on small system sizes, our results on harder instances suggest an exponential scaling of the original algorithm already for these moderate sizes n.

3 Alternative Paths

A straight line interpolation between initial Hamiltonian $\mathcal{H}_i$ and final Hamiltonian $\mathcal{H}_f$ is a convex combination of only of two Hamiltonians. However, there are plenty of other paths connecting these two but involving a third or even more intermediate Hamiltonians. For example, in the simple Ising model (6) it is known that even for constant-speed interpolations $s(t) = t/T$, the runtime can be improved from quadratic scaling (straight line) to linear scaling (nonlinear path) [31].

Since the XY-algorithm showed the best median performance in the previous section, we set the initial Hamiltonian to $\mathcal{H}^{xy}$ and probe two alternative algorithms based on paths shown in Fig. 4.

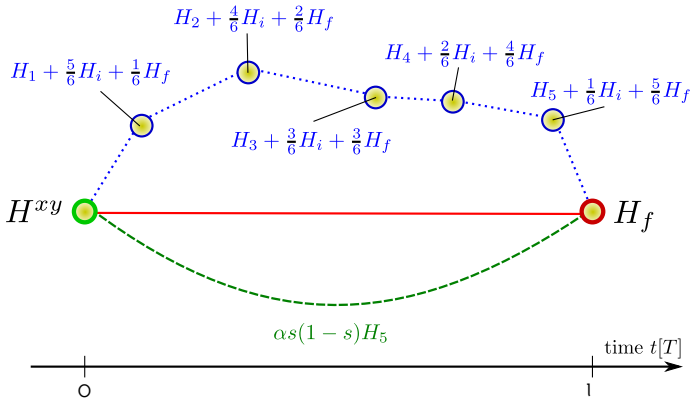

Figure 4: Sketch of alternative paths for adiabatic algorithms all starting from $\mathcal{H}^{xy}$. As an example, the number of clauses is set to $m = 6$. The red, solid line is the straight line algorithm. The green, dashed curve denotes a path with an additional, nonlinear term $s(1-s)H_5$, which is defined in the text. A clause-by-clause algorithm is shown by the blue, dotted lines.

3.1 Algorithms

3.1.1 Nonlinear Smooth Interpolation

We add a third term to the straight line interpolating Hamiltonian $\mathcal{H}(s)$ in Eq. (7), which is quadratic in s:

$$\mathcal{H}(s) = (1-s)\mathcal{H}^{xy} + s\mathcal{H}_f + \alpha s(1-s)\mathcal{H}_{m-1}, \tag{20}$$

where $\mathcal{H}_{m-1} = \Omega \sum_{i=1}^{m-1} h_i$ is the final Hamiltonian reduced by one (arbitrarily chosen) clause and α is a coupling strength. To motivate this path, we note that for large n, the related reduced EC3 problem may be expected to have many solutions, since there exists a phase transition from satisfiable to unsatisfiable EC3 problems at a clause-to-size ratio $m/n \approx 0.62$ [26]. Thus, reducing the number of clauses moves the problem into the satisfiable phase. Intuitively, we

expect that the additional term in Eq. (20), which becomes dominant during the evolution, will already at this state suppress states which are not a solution to $\mathcal{H}_{m-1}$ (and therefore neither of $\mathcal{H}_m$). Thereby, the search space to find the solution of $\mathcal{H}_f$ is reduced, and the algorithm could be expected to be faster compared to conventional straight line interpolation.

It should be noted, that for m clauses there are m different Hamiltonians $\mathcal{H}_{m-1}$. The best reduction of the search space is then obtained for reduced Hamiltonians $\mathcal{H}_{m-1}$ with the smallest number of solutions. However, this number will in realistic experiments not be known. In our numerical simulations, we have decided to remove only clauses when the connectivity of the graph is not destroyed. In the example of Fig. 1, allowed clauses to be removed are $(1, 4, 13), (1, 10, 11)$ and $(9, 11, 13)$.

3.1.2 Nonlinear Clause-By-Clause Interpolation

We try again to reduce to search space by applying an additional term to the straight line interpolation. In contrast to the previous case however, the reduction is conducted not in a single step but by switching on the clauses one after another. This can be written formally as

$$\mathcal{H}(s) = (1-s)\mathcal{H}^{xy} + s\mathcal{H}_f + \mathcal{H}_d(s), \tag{21}$$

$$\mathcal{H}_d(s) = \sum_{k=1}^{m} \left[(1-s_k)\mathcal{H}_{k-1} + s_k\mathcal{H}_k\right] \times \Theta(s_k)\Theta(1-s_k), \tag{22}$$

$$\text{where} \qquad s_k = ms - k + 1. \tag{23}$$

The primary interpolation $s : 0 \to 1$ of Eq. (21) is thus split into m steps, which consist of secondary interpolations $s_k : 0 \to 1$ from $\mathcal{H}_{k-1}$ to $\mathcal{H}_k$ in Eq. (22). Here, $\mathcal{H}_k$ consists of k clauses from $\mathcal{H}_f$. Note that this is equivalent to adding a single clause with each step denoted by $\mathcal{H}_k - \mathcal{H}_{k-1}$. As the initial and final Hamiltonians, $\mathcal{H}(0) = \mathcal{H}^{xy}$ and $\mathcal{H}(1) = \mathcal{H}_f$ should not be changed by $\mathcal{H}_d$, we define $H_0 = \mathcal{H}_m = 0 \cdot \mathbb{1}$. The order, in which clauses are added, is ambiguous, which results in $m!$ possible paths. In our numerical simulations, the path was defined by the order in which the clauses were stored.

3.2 Results

The qubit system is prepared in the ground state of $\mathcal{H}^{xy}$, which can be done efficiently as stated in section 2. We then compare for the presented algorithms the final energy E_f of the system after a runtime T. For an adiabatic runtime, the energy should be close to 0. Figure 5 exemplary shows our results for an instance with 13 qubits. First, it is visible that for short runtimes, the conventional algorithm rapidly decreases its final energy, but it becomes increasingly hard to further reduce the energy below the critical threshold one. Both the clause-by-clause algorithm and the XY-algorithm decrease the final energy significantly faster, but the nonlinear path shows an even better performance (we attribute the final plateaus to imperfect numerical preparation of the initial ground state). Although a broad statistics would exceed the scope of this paper, we observed a similar behavior for many instances and different system sizes.

However, there are very hard instances, where an arbitrarily chosen nonlinear algorithm failed. The graph depicted in Fig. 1 is such an example, which is as hard to solve for the XY-algorithm

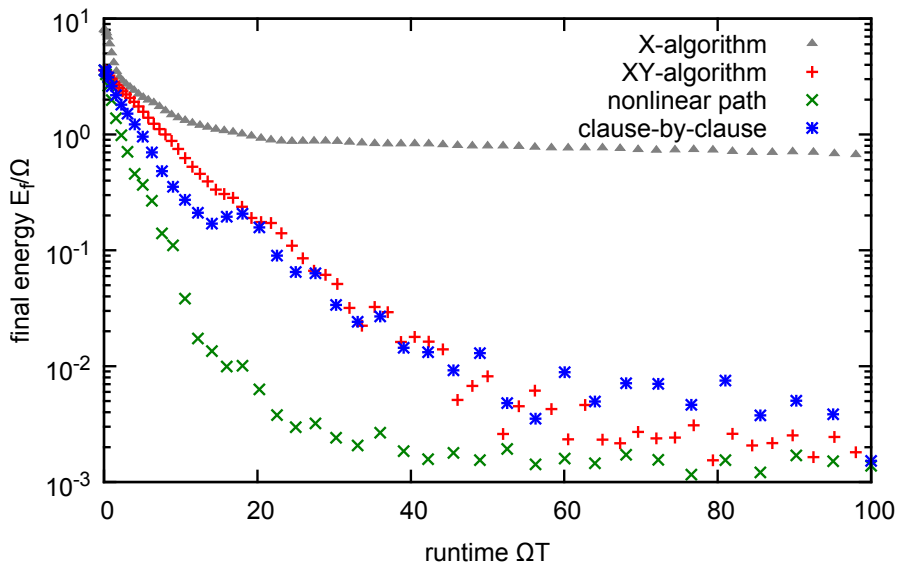

Figure 5: Plot of the final energy for different paths between $\mathcal{H}^{xy}$ and $\mathcal{H}_f$ for a 13 qubit example. Symbol shapes and colors correspond to the paths in Fig. 4. The X-algorithm is also shown as an example of a slow adiabatic behavior. For the nonlinear path, the coupling is chosen as $\alpha = 8$.

as it is for the X-algorithm. In this case, our chosen nonlinear path performed for a large coupling constant α even worse.

The choice of the nonlinear path turns out to be crucial as shown in Fig. 6. Here, both straight line algorithms do not reduce the energy significantly below 1. Also shown are the possible three nonlinear paths. Surprisingly, the energy of path A is almost constant 1 for long runtimes. At least two out of three nonlinear paths show a faster decrease in energy than the XY-algorithm, with path B having the best performance.

4 Conclusion

The employed universal integrator proved a flexible tool in simulating non-standard adiabatic quantum algorithms. The introduced SQH format offers an efficient storage scheme and a fast matrix-vector multiplication. Moreover, as the integrator is independent of the Hamiltonian's structure, it gives the flexibility to simulate hundreds of EC3 instances without changing the source code. Adapting the integrator to alternative interpolating paths could be done very easily.

Our simulations agree with previous results pointing to an exponential scaling of the X-algorithm [32] on hard instances. The performance of the XYZ- and the XY-algorithm is much better, indicating an Ising-like polynomial scaling for the samples and sizes considered. However, we note that variations are large, and the worst-case complexity does not even expose any scaling behavior. Even if this was not the case, finite-size simulations must remain inconclusive by construction. For the alternative paths, we did not study the runtime scaling versus the problem size. Instead, we considered the adiabatic behavior for exemplary instances. Here, the nonlinear path outperforms the linear algorithms even for very hard instances. However, in

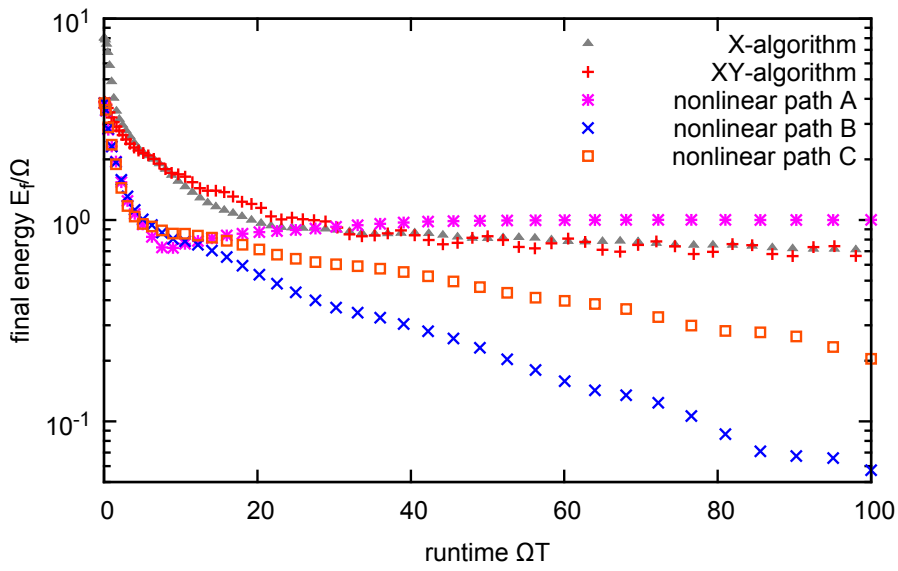

Figure 6: Final energies versus dimensionless runtimes for different choices of H_{m-1} with coupling $\alpha = 8$. Clauses removed from $\mathcal{H}_f$ for the nonlinear term are $A = (1, 10, 11), B = (1, 4, 13)$ and $C = (9, 11, 13)$. Symbol shapes and colors correspond to the clauses in Fig. 1. Whereas the short-time performance was much better for all nonlinear paths, it turned out that these differ strongly for large evolution times T, and choice A performs worse than the straight line interpolations.

general one will not know in advance, which of the possible choices for the nonlinear path is the best.

For further studies, an analysis of its scaling behavior is of interest. This requires extensive simulations for statistics, which will be a subject of future research.

Our results are of course limited to the specific examples considered, but may give rise to the hope that nonlinear paths may be an interesting road to explore in the field of adiabatic computation.

References

[1] J. Stolze and D. Suter. *Quantum Computing: A Short Course from Theory to Experiment*. Wiley-VCH, 2003.

[2] Richard P. Feynman. Simulating physics with computers. *International Journal for Theoretical Physics*, 21:467, 1982.

[3] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[4] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2000.

[5] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, Joshua Lapan, Andrew Lundgren, and Daniel Preda. A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. *Science*, 292:472–476, 2001.

[6] Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM J. Comput.*, 37:166, 2007.

[7] Ari Mizel, Daniel A. Lidar, and Morgan Mitchell. Simple proof of equivalence between adiabatic quantum computation and the circuit model. *Phys. Rev. Le*, 99:070502, 2007.

[8] Jeremie Roland and Nicolas J. Cerf. Quantum search by local adiabatic evolution. *Physical Review A*, 65:042308–6, 2002.

[9] A. Boulatov and V. N. Smelyanskiy. Quantum adiabatic algorithms and large spin tunnelling. *Physical Review A*, 68:062321, 2003.

[10] Marko Znidaric and Martin Horvat. Exponential complexity of an adiabatic algorithm for an NP-complete problem. *Physical Review A*, 73:022329, 2006.

[11] L. M. Ioannou and M. Mosca. Limitations of some simple adiabatic quantum algorithms. *International Journal of Quantum Information*, 6:419 – 426, 2008.

[12] A. P. Young, S. Knysh, and V. N. Smelyanskiy. Size dependence of the minimum excitation gap in the quantum adiabatic algorithm. *Physical Review Letters*, 101:170503, 2008.

[13] M. H. S. Amin and V. Choi. First-order quantum phase transition in adiabatic quantum computation. *Physical Review A*, 80:062326, 2009.

[14] G. Schaller and R. Schützhold. The role of symmetries in adiabatic quantum algorithms. *Quantum Information and Computation*, 10:0109–0140, 2010.

[15] B. Altshuler, H. Krovi, and J. Roland. Anderson localization makes adiabatic quantum optimization fail. *PNAS*, 107:12446, 2010.

[16] A. P. Young, S. Knysh, and V. N. Smelyanskiy. First-order phase transition in the quantum adiabatic algorithm. *Physical Review Letters*, 104:020502, 2010.

[17] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Daniel Nagaj. How to make the quantum adiabatic algorithm fail. *International Journal of Quantum Information*, 6:503–516, 2008.

[18] Thomas Monz, Philipp Schindler, Julio T. Barreiro, Michael Chwalla, Daniel Nigg, William A. Coish, Maximilian Harlander, Wolfgang Hänsel, Markus Hennrich, and Rainer Blatt. 14-qubit entanglement: Creation and coherence. *Phys. Rev. Lett.*, 106:130506, Mar 2011.

[19] M. S. Sarandy, L.-A. Wu, and D. A. Lidar. Consistency of the adiabatic theorem. *Quantum Information Processing*, 3(6):331–349, 2004.

[20] Andrew M. Childs, Edward Farhi, and John Preskill. Robustness of adiabatic quantum computation. *Physical Review A*, 65:012322, 2001.

[21] Sabine Jansen, Mary Beth Ruskai, and Ruedi Seiler. Bounds for the adiabatic approximation with applications to quantum computation. *Journal of Mathematical Physics*, 48:102111, 2007.

[22] Gernot Schaller, Sarah Mostame, and Ralf Schützhold. General error estimate for adiabatic quantum computing. *Physical Review A*, 73:062307, 2006.

[23] Neil Gershenfeld. *The Nature of Mathematical Modeling*. Cambridge University Press, Cambridge, 2000.

[24] M. C. Banuls, R. Orus, J. I. Latorre, A. Perez, and P. Ruiz-Femenia. Simulation of many-qubit quantum computation with matrix product states. *Physical Review A*, 73:022344, 2006.

[25] R. Schützhold and G. Schaller. Adiabatic quantum algorithms as quantum phase transitions: First versus second order. *Physical Review A*, 74:060304(R), 2006.

[26] Vamsi Kalapala and Christopher Moore. The phase transition in exact cover. *arXiv:*, cs.CC:0508037, 2005.

[27] Jacek Dziarmaga. Dynamics of a quantum phase transition: Exact solution of the quantum ising model. *Physical Review Letters*, 95:245701, 2005.

[28] Andrew M. Childs, Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. Finding cliques by quantum adiabatic evolution. *Quantum Information and Computation*, 2:181, 2002.

[29] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, 1998. http://www.caam.rice.edu/software/ARPACK.

[30] U. Kandler and C. Schröder. Backward error analysis of an inexact arnoldi method using a certain gram schmidt variant. *Preprint series of the Institute of Mathematics, TU Berlin*, pages 10–2013, 2013.

[31] G. Schaller. Adiabatic preparation without quantum phase transitions. *Physical Review A*, 78:032328, 2008.

[32] Marko Znidaric. Scaling of the running time of the quantum adiabatic algorithm for propositional satisfiability. *Physical Review A*, 71:062305, 2005.