IEEE Transactions on Quantum Engineering

Received 24 September 2024; accepted 14 April 2025; date of publication 22 April 2025; date of current version 27 May 2025.

Digital Object Identifier 10.1109/TQE.2025.3563805

Runtime–Coherence Tradeoffs for Hybrid Satisfiability Solvers

VAHIDEH ESHAGHIAN[1], SÖREN WILKENING[2,3], JOHAN ÅBERG[1], AND DAVID GROSS[1]

[1] Institute for Theoretical Physics, University of Cologne, 50937 Cologne, Germany
[2] Institut für Theoretische Physik, Leibniz Universität Hannover, 30167 Hanover, Germany
[3] Volkswagen AG, 38440 Wolfsburg, Germany

Corresponding author: Vahideh Eshaghian (e-mail: eshaghian.vahideh@gmail.com).

This work was supported by the Bundesministerium für Bildung und Forschung under project QuBRA. The UzK team was also supported by the Germany's Excellence Strategy—Cluster of Excellence Matter and Light for Quantum Computing (ML4Q) under Grant EXC 2004/1 (390534769).

ABSTRACT Many search-based quantum algorithms that achieve a theoretical speedup are not practically relevant since they require extraordinarily long coherence times, or lack the parallelizability of their classical counterparts. This raises the question of how to divide computational tasks into a collection of parallelizable subproblems, each of which can be solved by a quantum computer with limited coherence time. Here, we approach this question via hybrid algorithms for the k-satisfiability problem (k-SAT). Our analysis is based on Schöning's algorithm, which solves instances of k-SAT by performing random walks in the space of potential assignments. The search space of the walk allows for "natural" partitions, where we subject only one part of the partition to a Grover search, while the rest is sampled classically, thus resulting in a hybrid scheme. In this setting, we argue that there exists a simple tradeoff relation between the total runtime and the coherence time, which no such partition-based hybrid scheme can surpass. For several concrete choices of partitions, we explicitly determine the specific runtime coherence time relations and show saturation of the ideal tradeoff. Finally, we present numerical simulations, which suggest additional flexibility in implementing hybrid algorithms with the optimal tradeoff.

INDEX TERMS Coherence time, quantum algorithm, quantum search, runtime, satisfiability problem.

I. INTRODUCTION

Consider a quantum algorithm that takes exponential time to run, but still offers a polynomial speedup over the best classical method. Examples include Grover searches to brute-force a password or for finding the solution for a combinatorial optimization problem for which no classical heuristics exist. Fully quantum implementations might not be desirable for two reasons: 1) quantum hardware that can sustain very long computations might not be available and 2) quantum algorithms, such as Grover's search, might not be easily amenable to parallelization. This leads to the question of how to best break up such instances into a set of smaller parallelizable subproblems that can individually be solved on quantum hardware.

We consider the well-known satisfiability problem with k being the number of literals in each clause, (k-SAT), and focus particularly on 3-SAT since it provides an attractive test bed to investigate such questions. k-SAT is the archetypical combinatorial optimization problem and represents a class of use cases with considerable practical relevance. Moreover, there is a classical randomized algorithm [1], [2] due to Schöning, with a performance close to the best known algorithms with provable performance, and which furthermore allows for a closed-form asymptotic runtime analysis. It is indeed the case that the algorithm obtained by replacing the classical search of the Schöning procedure by a Grover search [3] yields a quantum Schöning algorithm with a quadratic improvement vis-à-vis its classical counterpart [4]. (In the following, we will refer to quantum algorithms that arise this way as Groverizations of their classical versions).

However, such "fully quantized" Schöning's SAT solvers cannot be performed in parallel, which arguably is a relevant feature for algorithms that run in exponential time. Hybrid schemes, based on "partial" Groverizations of Schöning's algorithm, where Grover search procedures are applied only to certain subroutines, usually do allow for parallelizations.

The starting point of our analysis is the stochastic nature of Schöning's algorithm as a random walk. This point of

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see http://creativecommons.org/licenses/by/4.0/

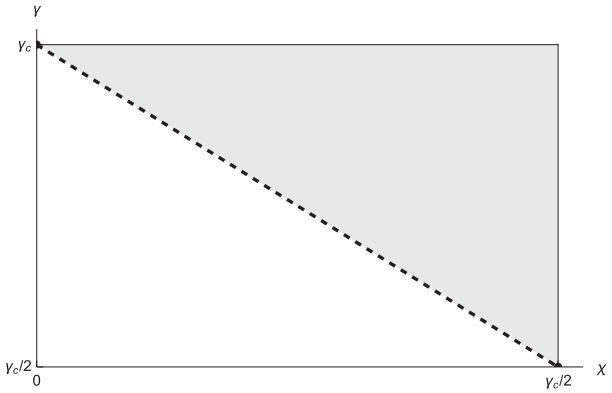

FIGURE 1. Visualization of the behavior of algorithms by indicating their position in a "runtime rate versus coherence time rate" chart. Classical algorithms require no coherence and thus lie on the γ-axis. In the example given, the point on the upper left-hand side represents a classical probabilistic search with runtime rate γ_C. A completely Groverized version has coordinates $(\gamma_C/2, \gamma_C/2)$ (bottom right), meaning that it will spend its entire runtime coherently. Hybrid algorithms that use Grover only for a subset of the search space must lie in the shaded area above or on the dashed line segment connecting these two points.

view yields two classes of hybrid algorithms, where one class Groverizes the random choice of the initial state of the walk, whereas the other class Groverizes the randomness in the walk itself. Within an established model of Schöning's algorithm, we optimize the resulting runtimes by balancing the resources allocated to the subroutines.

A. RUNTIME–COHERENCE TIME TRADEOFFS

Before specializing to the Schöning process, let us briefly outline the tradeoffs between runtime and coherence time that can be expected for quantum search problems. Consider an algorithm that solves instances of size n with runtime $T(n)$. For exponential-time algorithms, we work with a somewhat coarser measure, the (asymptotic) runtime rate

$$\gamma = \lim_{n\to\infty} \frac{1}{n} \log T(n)$$

where we drop the base of the logarithm from here on; the base is 2 unless explicitly stated otherwise. In other words, $T \in O^*(2^{\gamma n})$, where O^* denotes scaling behavior up to polynomial factors. The aim is to trade it off against the coherence time required to run the algorithm. If $C(n)$ is the longest time over which coherence has to be maintained while running the algorithm, then the coherence time rate is

$$\chi = \lim_{n\to\infty} \frac{1}{n} \log C(n).$$

Now restrict attention to search algorithms with a classical runtime rate γ_C. A completely Groverized version runs with rate $\gamma_G = \gamma_C/2$. All of its runtime will be spent coherently, specifically executing Grover iterations. Therefore, $\chi_G = \gamma_C/2$ as well. We can visualize these two points in a "runtime rate versus coherence time rate" chart, a mode of visualization that we will employ frequently (see Fig. 1).

To achieve a tradeoff between total runtime and coherence time, we will consider algorithms that apply Grover's procedure only to a subset of the search space. It is easy to see that any algorithm that results from such a procedure must have coordinates (χ, γ) that lie on or above the line segment

$$L = \{(\chi, \gamma_C - \chi) \mid \chi \in [0, \gamma_C/2]\}$$

that connects the purely classical point $(0, \gamma_C)$ to the completely Groverized one $(\gamma_C/2, \gamma_C/2)$.

Indeed, take a partial Groverization that achieves parameters (χ, γ). Then, one can replace the Grover part by a classical search. The resulting classical algorithm will have parameters $(0, \gamma + \chi)$, because the Grover search contributed χ to the runtime rate, but its classical simulation will contribute 2χ instead. However, if the initial parameters were below the line, i.e., if $\gamma < \gamma_C - \chi$, then the resulting classical algorithm runtime rate is $\gamma + \chi < \gamma_C$, contradicting the assumption that γ_C describes the classical complexity of the search.

B. RELATED WORK

Dunjko et al. [5] have previously considered partial Groverizations of Schöning's algorithm. They aimed to minimize a different metric, i.e., total number of clean qubits, rather than coherence time. In fact, they work in a highly constrained regime, where the number of available clean qubits only scales as cn, with $0 < c < 1$ and n the number of variables of the given 3-SAT formula. Surprisingly, they show that even this meager allotment of qubits, in principle, yields a speedup compared to the classical Schöning's algorithm.[1]

Despite the superficial similarities, their and our papers are quite different. We allow for qubit counts that are quasilinear in n, i.e., $O(n \log n)$, reasoning that for exponential-time algorithms, coherence time and parallelizability might be more limiting than the number of available qubits. As it will turn out, the setting considered here can interpolate between the classical and the fully Groverized performance, whereas the runtime rates obtainable in [5] stay close to the classical ones. While Dunjko et al. [5] use derandomization techniques, our approach builds more directly on the original Schöning's algorithm. This makes our approach technically less involved, and it also makes the lessons learned more widely applicable, since the basic technique of using Grover search over a subset of all variables directly generalizes to any NP problem, whereas derandomizations to a larger extent rely on the particular structure of the problem at hand.

[1] According to [5, Supplemental Material Sect. B.4], the relative speedup to the classical Schöning's rate is $f(c) = (1 - \log\sqrt{3})\beta(c)$, where the Beta function up to $O(\frac{\log n}{n})$ is implicitly given as $A\beta(c)\ln\frac{1}{\beta(c)} + B\beta(c) = c$. As mentioned in [5] using a straightforward encoding of each trit into two qubits, one can assume $A = 10$ and $B = 50$. To be consistent with our encoding, we consider $\log_2 3$ qubits to encode a trit and then calculate the maximum speedup in the rate, i.e., $f(1) \approx 0.0028$.

IEEE Transactions on Quantum Engineering

Algorithm 1: Schöning's Algorithm.

1: **function** SCHOENING($C_1, \ldots, C_L, N, m$)
2: **for** $i = 1 \ldots N$ **do**
3: $x \leftarrow$ uniformly random value from $\{0,1\}^{\times n}$
4: **for** $j = 1 \ldots m$ **do**
5: **if** x satisfies $C_1, \ldots, C_L$ **then**
6: **return** x
7: **else**
8: $k \leftarrow$ index of first unsatisfied clause
9: $l \leftarrow$ index of one of the three variables occurring in C_k, chosen uniformly at random
10: $x \leftarrow x$, with the lth bit of x flipped
11: **end if**
12: **end for**
13: **end for**
14: **return** False
15: **end function**

II. SETTING THE STAGE

A. SCHÖNING'S ALGORITHM

Here, we provide a very brief introduction to the pertinent aspects of Schöning's 3-SAT solver. For a more thorough review, we refer the reader to [1] and [2]. In the 3-SAT problem, we are given a collection of clauses $C_1, \ldots, C_L$ on n binary variables, where each clause is of the form $C_j = l_0^{(j)} \vee l_1^{(j)} \vee l_2^{(j)}$, and where each of the literals $l_0^{(j)}$, $l_1^{(j)}$, and $l_2^{(j)}$ is one of the binary variables or its negation. The 3-SAT formula is the conjunction of all the given clauses, $C := \wedge_{j=1}^{L} C_j$, and the computational task is to determine whether there exists an assignment of the n binary variables that satisfies C. According to Schöning [1], an algorithm exists that, although with runtime that is exponential in n, can perform better than an exhaustive search through all potential assignments.

Schöning's algorithm (see Algorithm 1) depends on two parameters N and m to be determined later. It begins by choosing an assignment $x \in \{0,1\}^{\times n}$ uniformly at random. The algorithm then performs an m-step random walk over the space of n-bit strings (the inner loop in Algorithm 1, from line 5). In every step, it checks (according to a predetermined order) all the clauses $C_1, \ldots, C_L$. If all are satisfied, then x is a solution and the algorithm terminates. Otherwise, it finds the first unsatisfied clause and chooses one of the three variables corresponding to the literals of that clause uniformly at random. The value of x is then updated, by negating that variable. This concludes the step. If no solution is found after m steps, the walk is terminated. Up to N such walks are attempted (the outer loop in Algorithm 1), each time using a fresh uniformly random starting point x.

B. ANALYSIS OF THE RUNTIME OF SCHÖNING'S ALGORITHM

The analysis of the runtime of Schöning's algorithm is sketched in [1] and [2], and a more in-depth analysis can be found in [6]. Here, we follow a very similar line of reasoning, with our particular ansatz in mind. In the following, we present an overview; see the Appendix for a more detailed account.

Assume that there is at least one satisfying assignment $x^\star$. We first aim to lower bound the probability that a given random walk finds a solution. Let x_0 be the (random) initial configuration and x_l be the one attained after the lth step of the random walk. The probability that any solution is found during any step of the walk is certainly at least as large as the probability $P(x_m = x^\star)$ that the walk finds $x^\star$ at the mth step. To analyze $P(x_m = x^\star)$, we follow in the steps of Schöning [1], [2] and focus on the evolution of the Hamming distance $d_H(x_l, x^\star)$ between the current configuration and the selected satisfying assignment $x^\star$.

The fundamental insight is that if a clause C_k is violated at the lth step, then at least one of the three variables that appear in C_k must differ between x_l and the satisfying assignment $x^\star$. Thus, the random flip decreases the Hamming distance to the solution with probability at least $1/3$

$$P\left(d_H(x_{l+1}, x^\star) = d_H(x_l, x^\star) - 1\right) \geq \frac{1}{3}. \tag{1}$$

This suggests to pass from a description of the process on bit strings to its projection $x_l \mapsto d_H(x_l, x^\star)$ onto $\mathbb{N}$. However, this would generally yield a process that would be no easier to analyze than the original one. One may, for example, note that although the Schöning -process $(x_l)_l$ is Markovian on the space of bit strings $\{0,1\}^{\times n}$, one cannot generally expect its projection $(d_H(x_l, x^\star))_l$ to be Markovian on $\mathbb{N}$.

The general idea for the analysis is to replace (via a coupling) the true projection $(d_H(x_l, x^\star))_l$ with another process $(d_l)_l$ on $\mathbb{Z}$, which is Markovian and which moreover upper-bounds the true Hamming distance

$$d_H(x_l, x^\star) \leq d_l. \tag{2}$$

More precisely, the Markov process $(d_l)_l$ is defined by the transition probabilities

$$P(d_{l+1} = d_l + 1) = \frac{2}{3}, \quad P(d_{l+1} = d_l - 1) = \frac{1}{3}. \tag{3}$$

The transition probabilities (3) can be interpreted as worst case scenarios of each step in the Schöning process.

From the bound (2), it follows that $P(x_l = x^\star) \geq P(d_l \leq 0)$. In other words, the success probability of the Schöning process is lower bounded by the probability that the substitute process d_l reaches 0.

Given the lower bound $P(d_m \leq 0)$ on the probability of success of each given walk, we expect at least one out of $N = 1/P(d_m \leq 0)$ walks to find $x^\star$. More precisely, if ϵ is the tolerated probability for failure, then the number of repetitions needed in order to find an existing solution satisfies

$$N \geq \frac{\log \epsilon}{\log(1 - P(d_m \leq 0))}. \tag{4}$$

The required number N of repetitions will be exponential in n. It is then common to take a coarser point of view and

only analyze the corresponding rate

$$\gamma := -\lim_{n\to\infty} \frac{1}{n} \log P(d_m \leq 0)$$
$$\text{so that} \quad N = O^*(2^{\gamma n}) \tag{5}$$

where O^* denotes scaling behavior up to polynomial factors in order to achieve any constant probability of failure ϵ. With the choice $m = n$ (i.e., the termination time is equal to the number of variables), it turns out [1], [2] that $\gamma \leq \log \frac{4}{3} \approx 0.415$.

It is surprisingly technically difficult to rigorously derive the "global bound" $P(x_l = x^\star) \geq P(d_l \leq 0)$ from the "local bound" (1). However, the Markovian version $(d_l)_l$ of the Hamming distance random walk is commonly accepted as a good (in fact, conservative) model of the Schöning process. In the main body of this article, we will, therefore, phrase our arguments in terms of that model. More technical details on the relation between the two processes are given in the Appendix.

C. PARTIAL GROVERIZATIONS: THE GENERAL IDEA

For random walks, we naturally tend to think of the randomness as being generated whenever needed, like when we assign the initial state, or make the random choices along the path. However, we can alternatively picture the walk as a deterministic process that is fed with an external random string S, a list from which it picks the next entry whenever a random choice is to be made. When the purpose of the walk is to find (an efficiently recognizable) solution to some computational problem, one can thus view the walk as a (deterministic) map that designates each input string S as being "successful" or "unsuccessful," in the sense of the walk reaching the satisfying solution $x^\star$ or not. To this mapping, we can in principle apply a Grover search procedure, since the walk (as well as the solution-recognition procedure) can be performed via reversible circuitry and can thus also be implemented coherently.

As described in the previous section, Schöning's algorithm proceeds with an initialization, followed by a random walk on the space of 2^n assignments. The initialization requires n bits of randomness, S_I, since the initial state is selected uniformly over all 2^n strings. A walk of length m requires a string S_W of $m \log 3$ bits to encode the needed randomness. The log 3-factor is due to the fact that, at each step, the algorithm randomly selects which one of the three literals (of the first violated clause) should be flipped. An m-step Schöning walk can thus be viewed as a map from $S = (S_I, S_W)$ to a binary variable that tells us whether a satisfying assignment has been reached or not.

With a coherent circuit that implements this map, we can thus replace the uniformly distributed random variable S, with a uniform superposition over a corresponding number of qubits, and proceed via standard Grover-iterations [3]. We would expect such a procedure to yield a satisfying assignment at a runtime that scales as $O^*(2^{n\gamma_G})$ iterations, with $\gamma_G = \frac{1}{2} \log \frac{4}{3} \approx 0.208$ [4], i.e., the standard quadratic speedup. Up to a few constant qubits, one needs $n + (\log 3 + \log L)m$ qubits to encode this map as a quantum circuit, where L is the number of clauses in the 3-SAT formula (more details are given in Section V). Since the number of clauses grows linearly in n for the regime of interest by the SAT phase transition conjecture [7], and for the Schöning walk $m = n$, the space complexity of such encoding is $O(n \log n)$.

The view of random walks as maps on random input strings opens up for the concept of partial Groverizations. Nothing would, in principle, prevent us from regarding only a part of the input string S as the input of the Grover procedure, while keeping the rest of the string classical. Needless to say, one would generally expect the result to be less efficient than the "full" Groverization. However, the gain would be that the partial Groverization breaks the tasks into a collection of subproblems, each of which can be run in parallel on a quantum device that requires shorter coherence time.

Although it seems reasonable to expect that such a division, in principle, is always possible, one may also expect that it, in general, would be challenging to find a quantum circuit that implements it in an economical manner. (We can always resort to a full coherent circuit for S in its entirety, putting the "classical part" in a diagonal state.) However, there may be "natural" divisions of the process, which can be exploited. For Schöning's algorithm, it is close to hand to consider the division $S = (S_I, S_W)$, i.e., the division of the required randomness into the initialization part and the walk part. One can, thus, consider two particularly natural classes of "partial" Groverizations of Schöning's algorithm. For one of these, the Groverized Initialization (GI), the choice of the initial state is implemented coherently, while the walk is kept "classical." For the Groverized Walk (GW), the choice of initial state is kept classical, while the walk itself is performed coherently.

As described in Section II-B, the actual analysis is based on the random walk $(d_l)_l$ on $\mathbb{Z}$, rather than the true Schöning walk on strings in $\{0, 1\}^{\times n}$. The idea is nevertheless the same; the required randomness is divided into the initialization and the walk per se, resulting in GI and GW processes. As described in Section II-B, the rate of the true Schöning process can be bounded by the rate of the substitute process $(d_l)_l$. It turns out that a similar argument can be made for GW (see the Appendix), thus yielding a rigorous bound for the rate also in this case. However, for the other processes, we rather regard the $(d_l)_l$ process as a model of the genuine Schöning walk, without rigorous guarantees of analogous bounds.

III. PARTIAL GROVERIZATIONS

The previous section introduced two types of partial Groverizations of Schöning's algorithm, GI and GW, based on the

IEEE Transactions on Quantum Engineering

Algorithm 2: Schöning Walk and Oracle.

function ORACLE(x_0, w)
  return TRUE if SCHOENINGWALK(x_0, w) satisfies all clauses, else FALSE
end function

function SCHOENINGWALK(x, w)
  for $j = 1, \ldots, m$ **do**
    if x violates one of $C_1, \ldots, C_L$ **then**
      $k \leftarrow$ index of first unsatisfied clause
      $l \leftarrow$ index of the w_jth variable occurring in C_k
      $x \leftarrow x$, with the lth bit of x flipped
    end if
  end for
  return x
end function

Algorithm 3: Groverized Initialization.

for $i = 1, \ldots, N_2$ **do**
  $w \leftarrow$ uniformly random value from $\{1, 2, 3\}^{\times m}$
  $x \leftarrow$ Grover search for $\lfloor \sqrt{N_1} \rfloor$ iterations using $\text{ORACLE}_w()$
  if x satisfies all clauses **then**
    return x
  end if
end for

Algorithm 4: Groverized Walk.

for $i = 1, \ldots, N_1$ **do**
  $x_0 \leftarrow$ uniformly random value from $\{0, 1\}^{\times n}$
  $w \leftarrow$ Grover search for $\lfloor \sqrt{N_2} \rfloor$ iterations using $\text{ORACLE}_{x_0}()$
  $x \leftarrow \text{SCHOENINGWALK}(x_0, w)$
  if x satisfies all clauses **then**
    return x
  end if
end for
return False

Algorithm 5: Fractional Groverized Initialization.

for $i = 1 \ldots N_2$ **do**
  $w \leftarrow$ uniformly random value from $\{1, 2, 3\}^{\times m}$
  for $j = 1 \ldots N_1^{(c)}$ **do**
    $x_c \leftarrow$ uniformly random value from $\{0, 1\}^{\times \lceil (1-z)n \rceil}$
    $x_q \leftarrow$ Grover search for $\left\lfloor \sqrt{N_1^{(q)}} \right\rfloor$ iterations using $\text{ORACLE}_{(x_c, w)}()$
    $x = (x_c, x_q)$
    if x satisfies all clauses **then**
      return x
    end if
  end for
end for
return False

division $S = (S_I, S_W)$, i.e., the initial and the walk randomness. In this section, we describe these schemes in detail and further discuss their "fractional" cases.

In the GI scheme, there is an outer loop that classically samples S_W and is followed by a Grover search inner loop over the space of all possible S_I. Similarly, GW starts with a classical outer loop that samples S_I and is followed by a Grover search inner loop over the space of all possible S_W (this space is well defined as the walk length is fixed). We obtain Fractional Groverized Initialization (FGI) by adapting GI to a regime where only a fraction z of the variables in the initialization can be searched coherently, with $0 \leq z \leq 1$. Fractional Groverized Walk (FGW) is similarly an adaption of GW to a regime where Grover search can be performed on the randomness of walks of at most m_q steps, with $0 \leq m_q$. In both these fractional schemes, two classical outer loops contain a Grover search inner loop. The algorithms introduced here depend on parameters (N_1, N_2, etc.), which will be specified explicitly in Section IV.

All Grover searches will use an oracle derived from the function shown in Algorithm 2: It tests whether a Schöning walk with an initial configuration $x \in \{0, 1\}^n$ and walk randomness $w \in \{1, 2, 3\}^m$ will lead to a satisfying assignment. For notational convenience, we let the elements of w take ternary in values, with the interpretation that w_l determines which of the three literals occurring in the first violated clause (if any) in step l of the walk is flipped. For a qubit-based implementation, it is not difficult to relabel the decision variables using $\lceil m \log 3 \rceil$ binary variables.

For the different variants of partial Groverizations discussed later, we will fix a subset of arguments to the oracle and consider it as a function of the remaining ones. Fixed arguments will be denoted as subscripts, e.g., $\text{ORACLE}_w : x \mapsto \text{ORACLE}(x, w)$. With these conventions, we have the following algorithms.

One may note that the Grover search in the GW only is guaranteed to succeed (with high probability) for a specific collection of initial states. The number of rounds N_1 of the outer loop is selected in such a way that it with high probability hits the set of advantageous initial states at least once, thus allowing the Grover procedure to reach the satisfying assignment. Similar remarks apply to the other partial Groverizations.

Next, we discuss the "fractional searches." In the first one, the argument x of the oracle is broken up as $x = (x_c, x_q)$, with x_q taking $\lfloor z \cdot n \rfloor$ bits and x_c being $\lceil (1 - z) \cdot n \rceil$ bits long. Here, $z \in [0, 1]$ is a free parameter whose value will be determined later.

The second fractional algorithm breaks up the walk randomness as $w = (w_c, w_q)$, with $w_c \in \{1, 2, 3\}^{m_c}$ and $w_q \in \{1, 2, 3\}^{m_q}$, respectively. Again, the values of m_c and m_q are chosen later.

Algorithm 6: Fractional Groverized Walk.

for $i = 1, \ldots, N_1$ **do**
  $x_0 \leftarrow$ uniformly random value from $\{0, 1\}^{\times n}$
  for $j = 1, \ldots, N_2^{(c)}$ **do**
    $w_c \leftarrow$ uniformly random value from $\{1, 2, 3\}^{\times m_c}$
    $w_q \leftarrow$ Grover search for $\left\lfloor \sqrt{N_2^{(q)}} \right\rfloor$ iterations using $\textsc{Oracle}_{(x_0, w_c)}()$
    $w = (w_c, w_q)$
    $x \leftarrow \textsc{SchoeningWalk}(x_0, w)$
    if x satisfies all clauses **then**
      return x
    end if
  end for
end for
return False

Algorithm 7: Evenly Fractionalized Grover.

for $i = 1, \ldots, N^{(c)}$ **do**
  $x_c \leftarrow$ uniformly random value from $\{0, 1\}^{\times \lceil (1-z)n \rceil}$
  $w_c \leftarrow$ uniformly random value from $\{1, 2, 3\}^{\times \lceil (1-z)m \rceil}$
  $(x_q, w_q) \leftarrow$ Grover search for $\left\lfloor \sqrt{N^{(q)}} \right\rfloor$ iterations using $\textsc{Oracle}_{(x_c, w_c)}()$
  $w = (w_c, w_q)$
  $x_0 = (x_c, x_q)$
  $x \leftarrow \textsc{SchoeningWalk}(x_0, w)$
  if x satisfies all clauses **then**
    return x
  end if
end for
return False

In the final algorithm (Algorithm 7), a fraction of $z \in [0, 1]$ of both types of variables, the ones corresponding to the initialization and the ones corresponding to the walk, will be treated quantum mechanically.

IV. RUNTIME ANALYSIS

We will now lower bound the probability of success of the various approaches. As a preparation, in Section IV-A, we give a brief account of the analysis of the classical case, before moving on to the Groverized versions in Section IV-B.

A. CLASSICAL SCHÖNING PROCESS

The main ideas of the classical analysis are close to their presentation in [1] and [2]. We work in the Markovian model $(d_l)_l$ for the behavior of the Hamming distances, as laid out in Section II-B. Frequently, it will be convenient to measure quantities "in units of n or m." For example, we will soon choose a number $\kappa \in [0, 1]$ and assume that the initial value d_0 is equal to κn. Of course, this only makes sense if κn is an integer. In order to keep the notation clean, we will implicitly assume that such expressions have been rounded to the next integer.

Choose numbers $\kappa, \nu \in [0, 1]$. A given walk $(d_l)_l$ is certainly successful (in the sense that $d_m \leq 0$) if:

1) the initial value is $d_0 = \kappa n$;
2) the random walk decreases the Hamming distance in exactly νm of its m steps;
3) the condition
$$\kappa n \leq (2\nu - 1)m \tag{6}$$
holds.

Indeed, the right-hand side of (6) is the difference between the number of steps where the Hamming distance has been decreased, νm, and the number of steps where the Hamming distance has been increased, $(1 - \nu)m$.

For any fixed pair of values κ, ν subject to (6), we will now compute the probability of this particular route to success. Denote the first event by E_1 and the second event by E_2. They occur with respective probabilities

$$P(E_1) = \frac{1}{2^n}\binom{n}{\kappa n}$$
$$P(E_2) = \binom{m}{\nu m}\left(\frac{1}{3}\right)^{\nu m}\left(\frac{2}{3}\right)^{(1-\nu)m}. \tag{7}$$

Since the two events are independent, the success probability of the walk is lower bounded by

$$\begin{aligned} P(x_m = x^\star | \kappa) &\geq P(d_m \leq 0 | x) \\ &\geq P(E_1 \wedge E_2) = P(E_1)P(E_2) \\ &= \frac{1}{2^n}\binom{n}{\kappa n}\binom{m}{\nu m}\left(\frac{1}{3}\right)^{\nu m}\left(\frac{2}{3}\right)^{(1-\nu)m}. \end{aligned} \tag{8}$$

The various binomial coefficients can be conveniently related to entropies. To this end, recall the definition of the binary entropy function

$$H(p) = -p \log p - (1 - p) \log(1 - p)$$

for $p \in [0, 1]$ and the relative entropy

$$D(p \parallel q) = -p \log q - (1 - p) \log(1 - q) - H(p)$$

for $p, q \in [0, 1]$. Then, using the well-known estimate [8, Ch. 11.1]

$$\frac{1}{n+1} 2^{nH(\kappa)} \leq \binom{n}{\kappa n} \leq 2^{nH(\kappa)}$$

equation (8) can, after some straightforward calculations, be concisely rewritten as

$$P(d_m \leq 0 | x) \gtrsim 2^{-(1-H(\kappa))n} 2^{-D(\nu \parallel \frac{1}{3})m} \tag{9}$$

where $\gtrsim$ denotes that an inequality holds asymptotically, up to a polynomial factor. Equation (9) directly gives an upper bound on the rate γ defined in (5). Since the rate expresses the logarithm of the complexity "in units of n," it makes sense

to also express the length of the walk in terms of $\mu := m/n$. Then

$$\gamma = -\lim_{n\to\infty} \frac{1}{n} \log P(d_m \leq 0|x) \leq 1 - H(\kappa) + \mu D(\nu \parallel 1/3) =: \gamma(\mu, \kappa, \nu). \quad (10)$$

In particular, the infimum of $\gamma(\mu, \kappa, \nu)$ subject to the constraints (6) and $0 \leq \mu, 0 \leq \nu, \kappa \leq 1$ is a valid bound for γ. We will perform such optimizations explicitly for the partially Groverized versions in Section IV-B. For the classical procedure, we just state the final result

$$\mu = 1, \quad \kappa = \frac{1}{3}, \quad \nu = \frac{2}{3}, \quad \gamma_C = \log \frac{4}{3} \simeq 0.4150. \quad (11)$$

Remark: One might be tempted to search a tighter bound by summing the contributions to the probability of success that arise from all consistent values for μ, κ, and ν, instead of just considering the extremal value. However, the rate of a sum of exponentially processes is asymptotically determined by the rate of the dominating summand alone, i.e., for all collections of $\gamma_i > 0$, it holds that

$$\lim_{n\to\infty} -\frac{1}{n} \log \sum_i 2^{-\gamma_i n} = \sup_i \gamma_i$$

(assuming convergence). Therefore, considering only the dominating term does not affect the overall asymptotic rate.

B. PARTIALLY GROVERIZED PROCESSES

In this section, we derive the main results of this article: bounds on the asymptotic rates for partially Groverized versions of Schöning's scheme.

1) GROVERIZED INITIALIZATION, ALGORITHM 3

For the parameters N_1 and N_2, we choose constant multiples of $1/P(E_1)$ and $1/P(E_2)$, respectively. The value of the constant depends on the acceptable probability ϵ of failure, as exhibited in (4). Since this constant does not affect the rate, we will not specify it here. The probabilities depend essentially on the parameters μ, κ, and ν, though. We will, therefore, write $N_1(\kappa)$ and $N_2(\mu, \nu)$. Because the asymptotic complexity of a Grover search is the square root of the classical complexity, the rate function of GI is then given by

$$\gamma_{GI}(\mu, \kappa, \nu) = \lim_{n\to\infty} \frac{1}{n} \log\left(\sqrt{N_1(\kappa)} N_2(\nu, \mu)\right) = \frac{1 - H(\kappa)}{2} + \mu D(\nu \parallel 1/3). \quad (12)$$

Likewise, the required coherence time scales with the number of Grover iterations, i.e., as $O^*(2^{\chi n})$, for

$$\chi(\kappa) := \lim_{n\to\infty} \frac{1}{n} \log \sqrt{N_1(\kappa)} = \frac{1 - H(\kappa)}{2}. \quad (13)$$

The parameters are constrained by

$$0 \leq \kappa \leq 1, \quad 0 \leq \mu, \quad 0 \leq \nu \leq 1, \quad \frac{\kappa}{2\nu - 1} \leq \mu \quad (14)$$

where the final condition is a rearranged version of the success criterion (6).

We now determine the minimal rate γ_{GI} over the consistent parameters. Because relative entropy is nonnegative, it is always advantageous to reduce the value of μ until it is minimal subject to the constraints. This is achieved by changing the final inequality in (14) to equality. Rearranging, we arrive at

$$0 \leq \kappa \leq 1, \quad 0 \leq \mu, \quad \nu = \frac{1}{2} + \frac{\kappa}{2\mu} \quad (15)$$

which allows us to eliminate $\nu = \nu(\kappa, \mu)$ from the problem. Varying γ with respect to μ gives rise to the criticality condition

$$0 \stackrel{!}{=} \partial_\mu \gamma_{GI}(\kappa, \mu) = \partial_\mu \mu D(1/2 + \kappa/(2\mu) \parallel 1/3) = \frac{1}{2} \log\left(\frac{\mu + \kappa}{\mu}\right) + \frac{1}{2} \log\left(\frac{\mu - \kappa}{\mu}\right) + \log 3 - \frac{3}{2}. \quad (16)$$

This can be solved explicitly, e.g., using a computer algebra system [9], leading to

$$\mu = 3\kappa \quad \Rightarrow \quad \nu = \frac{2}{3}, \quad \mu D(\nu \parallel 1/3) = \kappa. \quad (17)$$

Eliminating μ, we get

$$\gamma_{GI}(\kappa) = \frac{1 - H(\kappa)}{2} + \kappa$$
$$\chi_{GI}(\kappa) = \frac{1 - H(\kappa)}{2}. \quad (18)$$

The pair of equations (18) contain all information about the asymptotic behavior of the GI procedure. Each value of κ gives a solution for the two undetermined constants $N_1(\kappa), N_2(\mu = 3\kappa, \nu = \frac{2}{3})$ in Algorithm 3, in such a way that it will run with a small probability of returning a false negative. Varying κ, we thus obtain a family of algorithms that find different compromises between the required coherence time and the total runtime. The achievable pairs of values are shown in Fig. 2.

Finally, we explicitly determine the minimal rate achievable in the GI scheme. With the help of a computer algebra system [9], one easily finds

$$0 \stackrel{!}{=} \partial_\kappa \gamma_{GI}(\kappa) = \frac{1}{2} \log \frac{\kappa}{1 - \kappa} + 1$$
$$\Leftrightarrow \quad \log\left(\frac{1}{\kappa} - 1\right) = 2 \quad \Rightarrow \quad \kappa = \frac{1}{5} \quad (19)$$

which gives

$$\mu = \frac{3}{5}, \quad \gamma_{GI} = \frac{3 - \log 5}{2} \approx 0.339, \quad \chi_{GI} \simeq 0.139. \quad (20)$$

Remark: One can cast the final minimization into the form

$$\gamma_{GI} = \inf_\kappa \gamma_{GI}(\kappa) = \inf_\kappa \left(\frac{1 - H(\kappa)}{2} + \kappa\right) = -\sup_\kappa \left(-\kappa - \frac{H(\kappa) - 1}{2}\right).$$

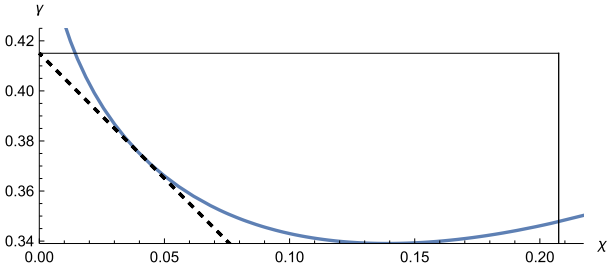

FIGURE 2. Rates $(\chi_{\mathrm{GI}}(\kappa), \gamma_{\mathrm{GI}}(\kappa))$ for the required coherence time and the total runtime of the GI algorithm, as the parameter κ is varied. The horizontal bar denotes the runtime rate achieved by the classical Schöning process. In other words, points above this line are uninteresting. The vertical bar denotes the coherence rate that allows one to run a completely Groverized version of the Schöning process. This, arguably, makes points to the right of this line uninteresting as well. Points to the left of the minimum (at $(\gamma, \chi) \simeq (0.339, 0.139)$) can represent advantageous choices if either the total coherence time of a quantum computer is limited or a larger degree of parallelization is desired. The dashed line is the lower bound on the runtime rate given the coherence time, as introduced in Fig. 1. It is achieved for $\kappa = \frac{1}{3}$.

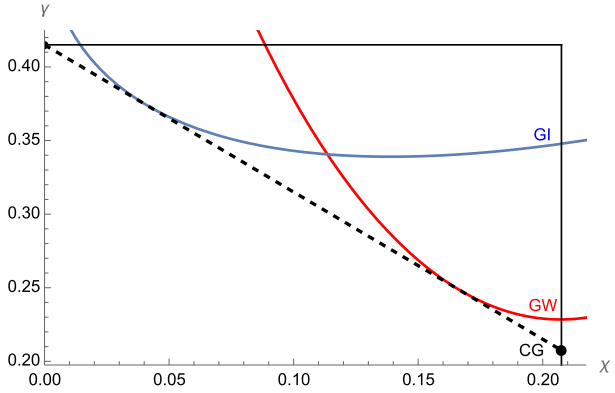

FIGURE 3. Runtime rate versus coherence time rate curves for Groverized Initialization (GI, blue) and Groverized Walk (GW, red). The point marked "CG" at the bottom right of the diagram represents the complete Groverization of the Schöning process. For long coherence times, GW is preferable, whereas for shorter coherence times GI achieves a lower total runtime.

This expression shows that the optimization amounts to computing a Legendre transform. Indeed, with $f(\kappa) := 1/2(H(\kappa) - 1)$, the right-hand side equals $-f^*(-1)$. For physicist readers, it might be amusing to note that $S(n\kappa) = nH(n\kappa)$ formally equals the entropy of an n-spin paramagnet as a function of the total magnetization. The Legendre transform of the entropy is a Massieu thermodynamic potential, equal to F/T (with F the free energy) expressed as a function of the inverse temperature [10, Ch. 5.4]. We will, however, not pursue this analogy here.

2) GROVERIZED WALK, ALGORITHM 4

The analysis proceeds in close analogy to the aforementioned case. The asymptotic rate function of GW is

$$\begin{aligned}\gamma_{\mathrm{GW}}(\kappa, \mu, \nu) &= \lim_{n\to\infty} \frac{1}{n} \log(N_1(\kappa)\sqrt{N_2(\nu, \mu)}) \\ &= 1 - H(\kappa) + \frac{\mu}{2} D(\nu \parallel 1/3) \qquad (21)\end{aligned}$$

subject to the set of constraints (14). The parameters ν and μ can be treated in exactly the same way as before, leading again to (17). In particular, the coherence time rate takes the simple form $\chi = \kappa/2$, which allows us to eliminate κ in favor of χ. We immediately obtain

$$\gamma_{\mathrm{GW}}(\chi) = 1 - H(2\chi) + \chi. \qquad (22)$$

Again, it is not difficult to solve for the lowest runtime [9]

$$\begin{aligned}\mu = 3(\sqrt{2} - 1) \quad \kappa = \sqrt{2} - 1 \\ \gamma_{\mathrm{GW}} \approx 0.228 \quad \chi_{\mathrm{GW}} \simeq 0.2071. \qquad (23)\end{aligned}$$

At the optimal point, the runtime scales with a rate that is very close to the one of a full Groverization of Schöning's process, namely, $\gamma_{\mathrm{CG}} = \gamma_{\mathrm{C}}/2 \simeq 0.2075$. The flip side is that the required coherence times are basically identical

$$\chi_{\mathrm{CG}} - \chi_{\mathrm{GW}} \simeq 0.0004.$$

The findings are summarized in Fig. 3.

3) FRACTIONAL GROVERIZED INITIALIZATION, ALGORITHM 5

In the case of Algorithm 5, the initial Hamming distance is the sum of two terms $d_0 = \kappa_c(1 - z)n + \kappa_q z n$, which model $d_H(x_c, x_c^\star)$ and $d_H(x_q, x_q^\star)$, respectively. Define the analogues

$$P(E_1^c) = \frac{1}{2^{(1-z)n}} \binom{(1-z)n}{\kappa_c(1-z)n} \quad P(E_1^q) = \frac{1}{2^{zn}} \binom{zn}{\kappa_q z n}$$

of $P(E_1)$ introduced in (7). Analogous to the discussion in Section IV-B1, the parameters N_1^c and N_1^q are defined as the reciprocals of these probabilities, times a constant that influences the probability of a false negative, but will not be discussed as it has no impact on the asymptotic rates. The success criterion is now

$$(1 - z)\kappa_c + z\kappa_q \le (2\nu - 1)\mu$$

and the other constraints are

$$0 \le \kappa_c, \kappa_q, \nu, z \le 1, \quad 0 \le \mu.$$

The asymptotic rate function for the runtime of FGI reads

$$\begin{aligned}&\gamma_{\mathrm{FGI}}(\kappa_c, \kappa_q, \nu, \mu; z) \\ &= \lim_{n\to\infty} \frac{1}{n} \log\left(N_1^c(\kappa_c; z)\sqrt{N_1^q(\kappa_q; z)} N_2(\nu, \mu)\right) \\ &= (1 - z)\,(1 - H(\kappa_c)) + \frac{z}{2}\left(1 - H(\kappa_q)\right) + \mu D(\nu \parallel 1/3). \qquad (24)\end{aligned}$$

Arguing as in Section IV-B1, the inequality in the success criterion may be replaced by an equality. Solving for ν gives

$$\nu = \frac{1}{2} + \frac{(1 - z)\kappa_c + z\kappa_q}{2\mu}.$$

IEEE Transactions on Quantum Engineering

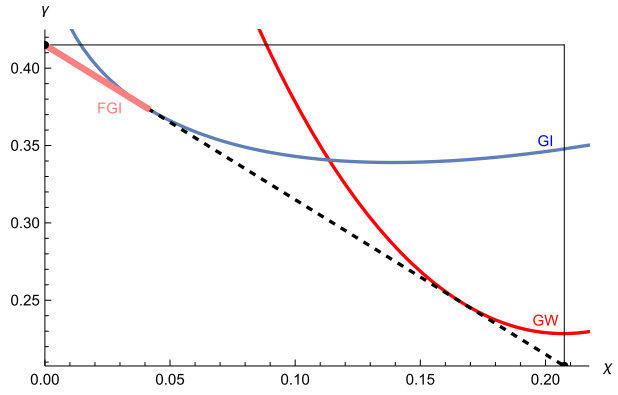

FIGURE 4. Runtime rate versus coherence time rate for the FGI algorithm. This fractional scheme's performance is the convex combination of the classical point $(0, \gamma_C)$, and GI at the tangent point to the theoretical lower bound. One can note that the FGI partially saturates the optimal performance relation.

We proceed as in the first two cases. Criticality of $\partial_\mu \gamma_{\mathrm{FGI}}$ with respect to μ occurs at

$$\mu = 3((1-z)\kappa_c + z\kappa_q)$$
$$\Rightarrow \quad \mu D(v \parallel 1/3) = (1-z)\kappa_c + z\kappa_q, \quad v = \frac{2}{3}.$$

Plugging in, we arrive at

$$\gamma_{\mathrm{FGI}}(\kappa_c, \kappa_q; z) = (1-z)\,(1 - H(\kappa_c) + \kappa_c) + z\left(\frac{1-H(\kappa_q)}{2} + \kappa_q\right). \tag{25}$$

In other words, the runtime rate function is a convex combination of the ones for the classical Schöning process and for the GI scheme, with weights $(1-z)$ and z, respectively. Because the classical part does not affect the coherence time, we may set κ_c to its optimal value $\kappa_c^* = 1/3$ [cf. (11)]. Geometrically, as we vary $z \in [0, 1]$, (25) describes a line connection $(\chi_{\mathrm{GI}}(\kappa_q), \gamma_{\mathrm{GI}}(\kappa_q))$ with the parameters of the classical Schöning process $(0, \gamma_C)$. By the convexity of the GI curve, the fractional algorithm will have a better runtime rate to the left of the value of κ_q at which the line becomes tangent to the curve. In other words, the critical κ_q is defined by the condition

$$\frac{\partial \gamma_{\mathrm{GI}}}{\partial \chi} = \frac{\gamma_{\mathrm{GI}} - \gamma_C}{\chi}.$$

By a computer calculation [9], this happens for $\kappa_q = \frac{1}{3}$ (i.e., equal to κ_c), resulting in Fig. 4.

4) FRACTIONAL GROVERIZED WALK, ALGORITHM 6

In the FGW scheme, we assume that the classical and Groverized walks decrease the Hamming distance in exactly $v_c m_c$ and $v_q m_q$ steps, respectively, where we have used a subscript to differentiate between the classical and Groverized random walks. The probabilities of such walks occurring are given as follows:

$$P(E_2^c) = \binom{m_c}{v_c m_c}\left(\frac{1}{3}\right)^{v_c m_c}\left(\frac{2}{3}\right)^{(1-v_c)m_c}$$
$$P(E_2^q) = \binom{m_q}{v_q m_q}\left(\frac{1}{3}\right)^{v_q m_q}\left(\frac{2}{3}\right)^{(1-v_q)m_q}.$$

Analogous to the discussion in Section IV-B1, the parameters N_2^c and N_2^q are defined as the reciprocals of the probabilities $P(E_2^c)$ and $P(E_2^q)$, times a constant that influences the probability of failure, but will not be discussed as it has no impact on the asymptotic rates. We further parameterize the walk lengths as $m_c = \mu_c n$ and $m_q = \mu_q n$. The runtime rate is

$$\begin{aligned}&\gamma_{\mathrm{FGW}}(\kappa, v_c, \mu_c, v_q, \mu_q)\\ &= \lim_{n\to\infty}\frac{1}{n}\log\left(N_1(\kappa)N_2^c(v_c,\mu_c)\sqrt{N_2^q(v_q,\mu_q)}\right)\\ &= 1 - H(\kappa) + \mu_c D(v_c \parallel 1/3) + \frac{\mu_q}{2}D(v_q \parallel 1/3)\end{aligned} \tag{26}$$

with parameters subject to the constraints

$$\begin{aligned}&0 \le \kappa \le 1\\ &0 \le \mu_c, \mu_q\\ &0 \le v_c, v_q \le 1\\ &\kappa \le (2v_c - 1)\mu_c + (2v_q - 1)\mu_q.\end{aligned} \tag{27}$$

The first steps of the analysis should now be familiar. There is no loss of generality in assuming that the final inequality is tight, which can be rearranged to give

$$v_q = \frac{\kappa - (2v_c - 1)\mu_c}{2\mu_q} + \frac{1}{2}.$$

The rate γ_{FGW} is stationary as a function of μ_q if

$$\mu_q = 3(\kappa - (2v_c - 1)\mu_c)$$
$$\Rightarrow \quad v_q = \frac{2}{3}, \frac{\mu_q}{2}D(v_q \parallel 1/3) = 1/2(\kappa - (2v_c - 1)\mu_c)$$
$$= \chi(\kappa, v_c, \mu_c).$$

Eliminating κ in favor of the coherence rate χ gives

$$\kappa = 2\chi + (2v_c - 1)\mu_c$$

and thus

$$\mu_q = 6\chi$$
$$\gamma_{\mathrm{FGW}}(v_c, \mu_c; \chi) = 1 - H(2\chi + (2v_c - 1)\mu_c) + \mu_c D(v_c \parallel 1/3) + \chi.$$

We now need to minimize γ_{FGW} for fixed χ as a function of μ_c and v_c, subject to

$$\begin{aligned}&0 \le 2\chi + (2v_c - 1)\mu_c \le 1\\ &0 \le \mu_c\\ &0 \le v_c \le 1.\end{aligned}$$

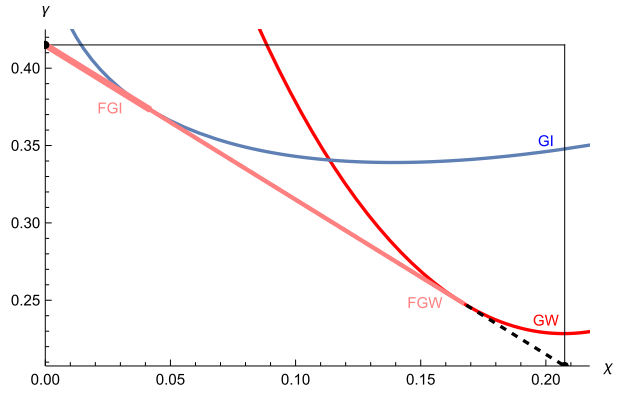

FIGURE 5. Runtime rate versus coherence time rate for the FGW algorithm. This fractional scheme's performance connects the GW curve to the classical Schöning point and is tangent to the curve. It achieves the optimal performance relation partially for a larger regime than FGI, and for low coherence times, it comes to lie on top of the FGI line.

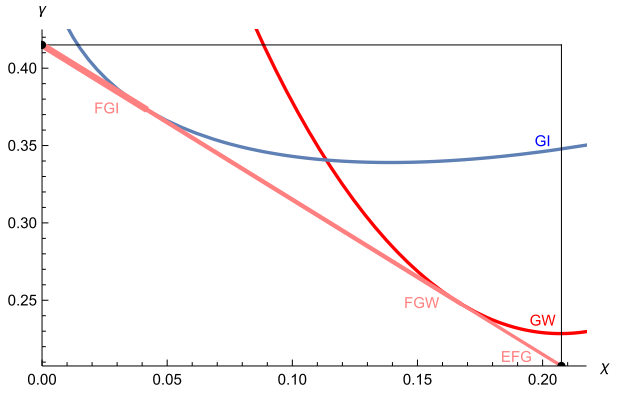

FIGURE 6. Runtime–coherence time rate curves for the covered algorithms. The linear interpolation between the classical and the completely Groverized points are realizable using an increasing number of methods—first only EFG, then also FGW, and finally also FGI—as the coherence time decreases.

We may assume that $\mu_c \neq 0$, for else we are just replicating the GW scheme. A computer calculation [9] gives

$$\partial_{\mu_c}(\gamma \ln 2) + \frac{2-4\nu_c}{4\mu_c}\partial_{\nu_c}(\gamma \ln 2)$$
$$= -\arctan(1-2\nu_c) + \ln(3-3\nu_c) - \frac{1}{2}\ln 2$$

which has zeros at $\nu_c = \frac{1}{3}$ and $\nu_c = \frac{2}{3}$.

For $\nu_c = \frac{1}{3}$, one finds

$$\partial_{\mu_c}(\gamma \ln 2) = \frac{2}{3}\arctan(1 - 4\chi + 2/3\mu_c)$$

which has one zero, at $\mu_c = \frac{3}{2}(4\chi - 1)$. The constraint $\mu_c \geq 0$ then implies $\chi \geq \frac{1}{4}$. However, this is larger than the coherence time rate $\gamma_C/2 \simeq 0.208$ sufficient to implement a completely Groverized version of Schöning's process, so this solution is not of interest.

We turn to the other solution, $\nu_c = \frac{2}{3}$. For it

$$\partial_{\mu_c}(\gamma \ln 2) = 1/3(-2\,\mathrm{arctanh}(1 - 4\chi - (2\mu_c)/3) + \ln 2)$$

which has one zero

$$\mu_c = 1 - 6\chi$$
$$\Rightarrow \quad \mu_q = 6\chi, \quad \nu_c = \nu_q = \frac{2}{3}, \quad \gamma_{\mathrm{FGW}} = \gamma_C - \chi.$$

The runtime versus coherence rate curve for the FGW scheme is given in Fig. 5.

5) EVENLY FRACTIONALIZED GROVER

The runtime rate is

$$\gamma_{\mathrm{EFG}} = (1-z)\,(1 - H(\kappa_c) + \mu_c D(\nu_c \parallel 1/3))$$
$$+ z/2\,\big(1 - H(\kappa_q) + \mu_q D(\nu_q \parallel 1/3)\big) \qquad (28)$$

with the success criterion

$$(1-z)\kappa_c + z\kappa_q = (1-z)(2\nu_c - 1)\mu_c + z(2\nu_q - 1)\mu_q$$

which is in particular true if the following two equations hold:

$$\kappa_c = (2\nu_c - 1)\mu_c \quad \kappa_q = (2\nu_q - 1)\mu_q.$$

However, this is just the convex interpolation between a completely classical and a completely Groverized process. In particular, by choosing the parameters as for the original Schöning process

$$\nu_c = \nu_q = \frac{2}{3} \quad \kappa_c = \kappa_q = \frac{1}{3} \quad \mu_c = \mu_q = 1$$

we obtain a coherence time–runtime rate curve that linearly connects the classical point $(0, \gamma_C)$ to the completely Groverized one $(\gamma_C/2, \gamma_C/2)$ (see Fig. 6).

C. HEURISTIC DERANDOMIZATION OF THE GI SCHEMES

In this section, we provide evidence that the GI schemes can reach further into the γ–χ chart than what the Markovian model suggests. To see why this is plausible, note that the role of randomness for the initial configuration x is very different from the role of randomness for the walk decisions w. In the first case, there is an "absolute measures of the quality of the initial configuration," namely, the Hamming distance to the solution. The probability that the walk does find the solution is quite obviously a function of that metric. Therefore, baring major algorithmic insights, it is unavoidable to consider many different initial configurations before encountering one that will likely lead to a solution.

In contrast, it is not implausible that "every walk works for equally many initial configurations," i.e., that there are no choices for w that are "intrinsically better than others." More precisely, it seems reasonable to assume that for sufficiently large n, and generic SAT formulas, it holds that with high probability in w

$$\frac{-1}{n}\log \Pr_x[\mathrm{SchöningWalk}(x, w) = x^\star \mid d_H(x, x^\star) = h, w]$$

IEEE Transactions on Quantum Engineering

Algorithm 8 Heuristically Derandomized FGI.

$w \leftarrow$ uniformly random value from $\{1, 2, 3\}^{\times m}$
for $j = 1 \ldots N_1^{(c)}$ **do**
  $x_c \leftarrow$ uniformly random value from $\{0, 1\}^{\times \lceil (1-z)n \rceil}$
  $x_q \leftarrow$ Grover search for $\left\lfloor \sqrt{N_1^{(q)}} \right\rfloor$ iterations using $\textsc{Oracle}_{(x_c, w)}()$
  $x = (x_c, x_q)$
  if x satisfies all clauses **then**
    return x
  end if
end for
return False

$$\simeq \frac{-1}{n} \log \Pr_{x,w'}[\text{SchöningWalk}(x, w') = x^\star \mid d_H(x, x^\star) = h]. \tag{29}$$

The right-hand side can be easily calculated, as in [1], for $\mu = 3$

$$\Pr_{x,w}[\text{SchöningWalk}(x, w) = x^\star \mid d_H(x, x^\star) = h] = 2^{-h}.$$

Under assumption (29), one can restrict the outer loop over w's from Algorithm 3 to $N_2 = 1$ iteration and compensate by increasing the number of Grover iterations for x to $N_1 = O^*(2^{\gamma_C/2n})$. In other words, the GI scheme with these parameters would lie on the optimal point $(\chi, \gamma) = (\gamma_C/2, \gamma_C/2)$.

Being even bolder, one could then speculate that the analysis of Section IV-B3 carries over and that, as one varies the fraction of initialization bits that are subjected to a Grover search, one could trace out the optimal (χ, γ) line. In other words, it does not seem impossible that Algorithm 8, with parameter choice

$$N_1^{(c)} = O^*(2^{\gamma_C(1-z)n}) \quad N_1^{(q)} = O^*(2^{\gamma_C zn/2})$$

achieves the optimal tradeoff.

To gather evidence in favor of assumption (29), we have resorted to numerical methods. A first ansatz is to compute the left-hand side of (29) exactly, which is possible for small values of n by iterating over all 2^n assignments to x. Results are shown in Fig. 7 for a randomly chosen set of 3-SAT formulas with $n = 20$ variables and $L = 91$ clauses. The number of satisfying assignments t_0 of the formulas are varied. Only the case $t_0 = 1$ can be directly compared to the analytic bounds. However, note that even for this case, the empirically observed rate of $\gamma_{GI} \simeq 0.12 \pm 0.02$ is much lower than the value $\gamma_C/2 \simeq 0.208$ that we would expect theoretically. Presumably, $n = 20$ is still too small to show the asymptotic behavior.

To test this assumption, we had to turn to numerical heuristics, to at least probe the behavior for much larger values of n, where an exact computation is no longer possible. The results are shown in Fig. 8. We used a SAT instance with $n = 1414$ variables that we believe to have a single satisfying assignment $x^\star$, which is explicitly known. To generate the instance, a 128-bit plain text was encoded by a 128-bit key using the XTEA block cipher truncated to three rounds. The formula represents the conditions on an input key to map the known plain text to the known ciphertext. The clauses are designed such that they enforce the correct evaluation of bitwise operations of the algorithm with respect to the given input and output. XTEA was restricted to three rounds in order to keep the size of the formula manageable. While we have no formal proof, it is reasonable to assume that there is a unique key that satisfies the formula. This is supported by consistency checks in terms of running SAT solvers on a version of this problems with even fewer rounds [11].

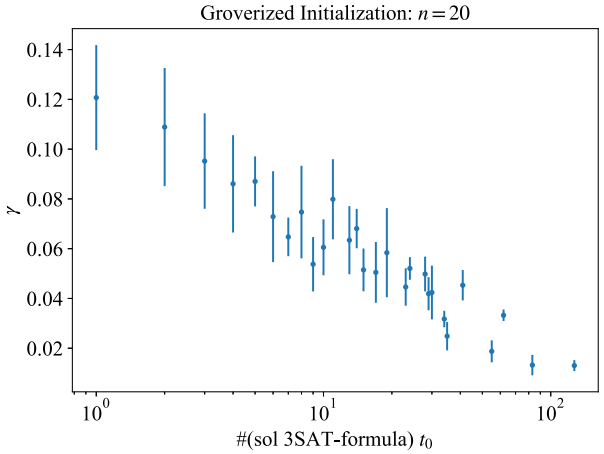

FIGURE 7. Plot of the runtime rate for the heuristically derandomized GI scheme. Error bars indicate variation as a function of the formulas and the walk variables w. On the x-axis, we show the number of satisfying assignments in the formula. Only the case of $t_0 = 1$ should be directly comparable to the analytic bounds. The empirically observed behavior is much better than the analytic results, suggesting that $n = 20$ is too small to capture the asymptotic behavior.

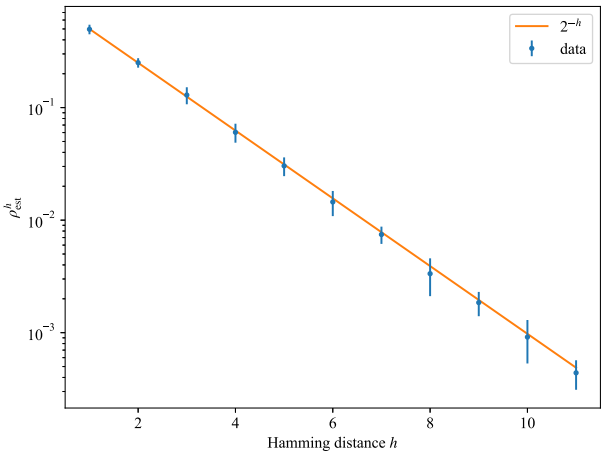

FIGURE 8. Estimated probability for a uniformly random initial configuration x with Hamming distance h to be mapped to $x^\star$ under a Schöning walk, for a fixed randomly chosen set of walk decisions w (c.f. Algorithm 8). The SAT instance has $n = 1414$ variables and is believed to have a unique satisfying assignment [11]. For each data point, 10^4 initial configurations x were sampled uniformly from the Hamming distance sphere $M^h(x^\star)$.

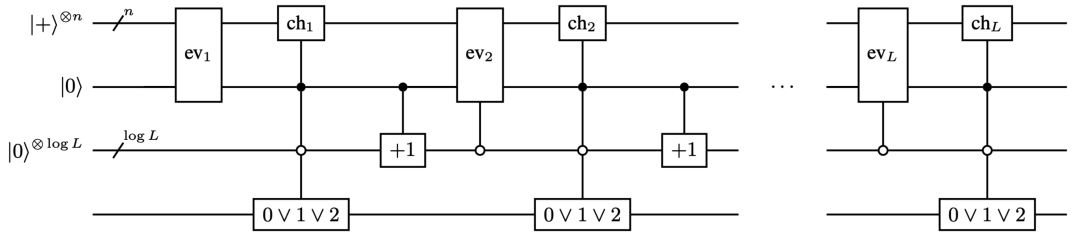

FIGURE 9. Quantum implementation of a single Schöning's step for a general implementation of the partial Groverization of Schöning's algorithm. The ev_j gates evaluate the jth clause on the corresponding variables and the controlled gates containing ch_i and $0 \vee 1 \vee 2$ act on all the registers and check if the jth clause is the first violated clause and if so, flip one of three variables in it based on the randomness provided by the if-statement, $0 \vee 1 \vee 2$. Here, $0 \vee 1 \vee 2$ represents a triple controlled gate where the control qutrit is the subspaces of the computational basis (visualized in Fig. 10). The log L auxiliary qubits are needed for uncomputation.

Let us denote the sphere of strings with Hamming distance h from $x^\star$ by $M^h(x^\star)$. For a fixed walk randomness w, and for $h = 1, \ldots 11$, we have drawn x uniformly from $M^h(x^\star)$. In order to compare the numerical results to the theory prediction, we have to use the value of the right-hand side of assumption (29) for nonasymptotic values of n. Fig. 8 shows the empirically estimated probabilities of Schöning's walk (with $\mu = 3$) arriving at the solution, when starting from a random initial configuration of given Hamming distance. The findings show the expected behavior of averaging over w, already for a fixed random value of w. In this sense, they are compatible with assumption (29). We note, however, that we were not able to probe the assumption for larger values of h. Garnering a better understanding for the concentration properties of the Schöning walk as a function of the walk choices remains, therefore, an open question.

V. CIRCUITS

In this section, we discuss an implementation of the partial Groverization schemes and present the main building blocks of their quantum circuits. Given n variables and the length of Schöning's walk m, the quantum implementation requires $n + m \log 3$ qubits to encode the initializations and walk randomness. The oracles of the partial Groverization schemes are some adaptation of one or more Schöning walks, and regardless of the search space they act on, the label of the violated clause at each step needs to be stored in their workspaces. This is necessary since such oracles are typically realized using uncomputation; therefore, $\log L$ extra auxiliary qubits are needed at each step, amounting to $m \log L$ qubits in total for the workspace. As a result, encoding any Groverization of Schöning's algorithm asymptotically needs $n + (\log 3 + \log L)m$ qubits.

Fig. 9 represents a single step of Schöning walk, schematically. The first register encodes the space of all possible initialization. The gates ev_j, for $j \in \{1, \ldots, L\}$, act on the first two registers. Each gate consists of a few controlled gates where the control qubits correspond to the three variables in the jth clause, and the target qubit is the second register. The second register is an auxiliary qubit, initially set to $|0\rangle$, and is negated as soon as the first violated clause is detected. The third register consists of $\log L$ auxiliary qubits that are used to count the number of clauses from where the first violated clause has happened. The last register is a qutrit providing the randomness of the corresponding walk step. The controlled gates ch_j, for $j \in \{1, \ldots, L\}$ act on the first three registers, and take care of variable flipping wherever the first violated clause is detected. The $0 \vee 1 \vee 2$ block represents a triple controlled gate where the control qutrit is the subspaces corresponding to the computational basis states $|0\rangle, |1\rangle, |2\rangle$. Fig. 10 depicts the controlled gates, including ch_j, in detail. The subfigure on the right shows the corresponding controlled gate for GI, where the walk randomness is fed classically to the last register.

All partial Groverization of the Schöning algorithm can be implemented using slight modifications. For the GW algorithm, the n-qubit variable register will not be initialized in the uniform superposition of all possible assignments $|+\rangle^{\otimes n}$, but rather in a state with classically randomly defined variables $|x_1 \cdots x_n\rangle$. For the GI algorithm, the qutrit within every Schöning's step can be removed since we can, for every Schöning's step, generate a random number $r \in \{0, 1, 2\}$ and apply only the X gates based on the classically determined r (see Fig. 10).

VI. SUMMARY AND OUTLOOK

This work considers hybrid schemes for search-based quantum algorithms, with the aim to allow for parallelizability, and to reduce the need for long coherence times. The basic gist is to partition the randomness of an underlying classical probabilistic algorithm into a part that is subject to Grover search, while the rest is sampled classically. Such "partial Groverizations" allow for the parallelization of the classical sampling, as well as enable adaption to available coherence times. We consider exponential-time algorithms, which is why our analysis focuses on the asymptotic runtime rates and coherence time rates. We argue that these two types of rates are bounded by a general tradeoff relation that no hybrid scheme can beat. For our concrete analysis, we consider hybrid schemes based on Schöning's algorithm, where the latter solves 3-SAT (or more generally k-SAT) problems by random walks in the space of assignments. The walk procedure allows for several partial Groverization schemes. We determine the corresponding runtimes and coherence times

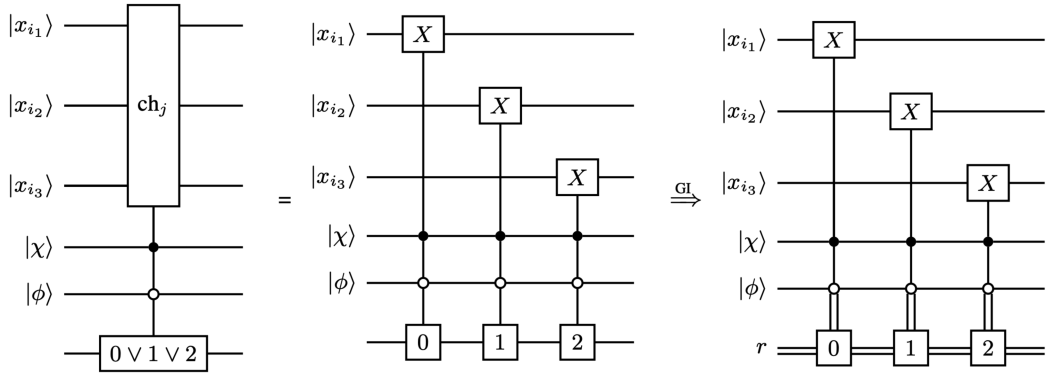

FIGURE 10. Implementation of the variable flips of Schöning's walk within amplitude amplification. Here, x_{i_1}, x_{i_2}, and x_{i_3} are the variables of the jth clause. The $0 \vee 1 \vee 2$ block represents a triple controlled gate where the control qutrit is the subspaces of the basis $|0\rangle, |1\rangle, |2\rangle$. For the GI algorithm, the walk randomness can be provided by fixing a random number $r \in \{0, 1, 2\}$ for every walk step.

of these schemes and demonstrate saturation of the general tradeoff relation. Many of these partial Groverizations intuitively lend themselves for efficient circuit implementations, and we provide the main building blocks of these. On a more speculative note, we present numerical evidence that the GI scheme can be partially derandomized, in the sense that a single "typical" instance of the classical randomness of the walk appears to mimic the effects of the repeated sampling. This would open for an additional flexibility in the implementation of these hybrid schemes, still maintaining the optional tradeoff.

In this investigation, we have focused on partial Groverizations of Schöning's algorithm. However, this approach should, in principle, be applicable to any classical probabilistic search scheme, since it essentially only rests on partitions of the underlying randomness. The main concern would be to find "natural" partitions that are algorithmically accessible, in the sense that the partial Groverization can be implemented efficiently. Explicit runtime and coherence time rates would also require a classical scheme, as well as partitions, that are sufficiently tractable for analysis, unless one would resort to numerical estimates.

The partial derandomization of the GI scheme that is suggested by our numerical explorations would deserve further investigations. In particular, the question is to what extent, and in what sense, the hypothetical relation (29) would be true. Moreover, one may ask if something similar also would apply to FGI. For numerical investigations, it would be relevant to extend to larger Hamming distances, further classes of 3-SAT instances, as well as problem sizes. This would likely involve challenges to design reliable numerical estimates, since exact calculations by the very nature of the problem quickly become intractable. For purely analytical approaches, some notion of concentration of measure of walks would be interesting.

In the spirit of [1] and [2], we have in this investigation employed "the walk on $\mathbb{Z}$" as a model of the true Schöning procedure. In the Appendix (see also [6]), we additionally provide bounds for the true rates of Schöning procedure and the GW procedure, in terms of the mirroring processes on $\mathbb{Z}$. It would be relevant to obtain similar bounds also for the GI process, as well as for the various fractional schemes.

APPENDIX
FROM THE TRUE SCHÖNING PROCESS TO THE MARKOV PROCESS ON $\mathbb{Z}$

A. PURPOSE OF THIS APPENDIX

For the calculation of rates, we replace the genuine searches of solutions for 3-SAT problems (the "true Schöning process") with a Markovian random walk on the "Hamming distance" (although we strictly speaking consider a walk on $\mathbb{Z}$). This is analogous to Schöning's analysis of the performance of Schöning's algorithm [1], [2], where it is argued that this substitute process yields an upper bound on the rates of the runtime of the algorithm. The purpose of this appendix is to give a more detailed argument for why the success probability of Schöning's algorithm is lower bounded by the success probability of the substitute walk on $\mathbb{Z}$. The reader may also wish to consult [6] for a previous analysis along these lines. Apart from bounding the success probability for the true Schöning process, we also provide the analogous bound for the GW process.

B. SCHÖNING PROCESS

As described in the main text, the 3-SAT problem consists of a collection of clauses $C_1, \ldots, C_L$ on n binary variables, where each clause is of the form $C_j = l_0^{(j)} \vee l_1^{(j)} \vee l_2^{(j)}$, and where each of the literals $l_0^{(j)}, l_1^{(j)}$, and $l_2^{(j)}$ is one of the binary variables, or its negation. The 3-SAT formula is the conjunction of all the given clauses, $C := \wedge_{j=1}^{L} C_j$, and the task is to determine whether there exists an assignment $x \in \{0, 1\}^{\times n}$ of the n binary variables, which satisfies C. In the following analysis we assume either that C has a unique satisfying assignment $x^\star \in \{0, 1\}^{\times n}$ or, alternatively, that $x^\star$ is *selected* among a set of solutions.

Schöning's procedure can be regarded as a stochastic process $(x_l)_{l=0}^{m}$ with $x_l \in \{0, 1\}^{\times n}$. The process is initialized by

a random assignment x_0 of the bit string, drawn uniformly over all of $\{0,1\}^{\times n}$. On this state, it checks all the clauses $C_1, \ldots, C_L$ (according to a predetermined order). If all are satisfied, then the initial string satisfies C and the algorithm terminates. Otherwise, it finds the first unsatisfied clause and randomly negates one of the three variables corresponding to the literals of that clause. The algorithm continues according to this random walk until it either finds a satisfying assignment, or it reaches a predetermined termination time K. For our purposes, it is convenient to think of the state x_l of the process as a function of a collection of random variables. The initialization is represented by the random variable A, which takes values in $\{0,1\}^{\times n}$. The randomness in the walk is captured by the variables $B = (B_1, \ldots, B_m)$ as random variables where each B_l takes values in $\{0,1,2\}$ (and thus B takes values in $\{0,1,2\}^{\times m}$). Hence, B_l represents one of the three possible choices of which literal to flip at step l. We assume that $A, B_1, \ldots, B_m$ are independent and uniformly distributed, i.e., for $b = (b_1, \ldots, b_m)$, we have

$$\begin{aligned} &P(A=a, B=b) = P(A=a)P(B=b) \\ &= P(A=a)P(B_1=b_1)\cdots P(B_m=b_m), \\ &P(A=a) = \frac{1}{2^n} \quad \forall a \in \{0,1\}^{\times n} \\ &P(B_l=b_l) = \frac{1}{3} \quad \forall b_l \in 0,1,2. \end{aligned} \tag{30}$$

Hence, we can write the Schöning process as $(x_l)_l = (x_l(A,B))_l$, where

$$x_0(a,b) := a \tag{31}$$

i.e., a the initial state. At the lth step, Schöning's process is based on the state x_{l-1} of the previous step. On this state, all the clauses $C_1, \ldots, C_L$ (according to a predetermined order) are checked. If all are satisfied, then $x_{l-1} = x^\star$ and the process remains in that state, i.e., $x_l = x^\star$. (In other words, $x^\star$ is an absorbing state for the Schöning process.) Otherwise, it finds the first unsatisfied clause, which we refer to as C_{j_l}. The selected clause, C_{j_l}, contains the three literals $(l_0^{(j)}, l_1^{(j)}$, and $l_2^{(j)})$. The process constructs x_l by negating the variable corresponding to literal $l_{b_l}^{(j)}$. In other words, it is the lth component of b that determines which of these three choices are selected. One may note that the process, by construction, satisfies

$$x_l(a,b) = x_l(a,b_1,\ldots,b_l). \tag{32}$$

Hence, the value of $x_l(a,b)$ only depends on the values of $b_1, \ldots, b_l$, not any of the "later" variables $b_{l+1}, b_{l+1}, \ldots$. One may also note that $A, B_1, B_2, \ldots, B_K$ encompass all the randomness in the process. In other words, the state x_l is uniquely determined by $a, b_1, \ldots, b_l$.

C. PROOF IDEA

As described previously, the true Schöning process $(x_l)_l$ is a walk on bit strings. However, for the analysis of the optimal rates, we follow the steps of Schöning [1], [2] and instead focus on the Hamming distance to the (selected) solution $x^\star$. In principle, nothing prevents us from projecting the state x_l of the Schöning process to the Hamming distance $d_H(x_l, x^\star)$ (i.e., projecting onto $\mathbb{N}$). However, this would generally yield a process that would be no easier to analyze than the original Schöning process. One may, for example, note that although the Schöning process $(x_l)_l$ is Markovian on the space of bit strings, one cannot generally expect its projection $(d_H(x_l, x^\star))_l$ to be Markovian on $\mathbb{N}$. The general idea for the analysis is to replace (via a coupling) the true projection $(d_H(x_l, x^\star))_l$ with another process $(\tilde{d}_l)_l$ on $\mathbb{N}$, which is Markovian and which moreover upper-bounds the true Hamming distance, $d_H(x_l, x^\star) \leq \tilde{d}_l$. One may note that the Schöning process is "successful" if it finds the solution $x^\star$. Hence, we can express the success probability at step l as $P(x_l = x^\star) = P(d_H(x_l, x^\star) = 0)$. From the bound $d_H(x_l, x^\star) \leq \tilde{d}_l$, it follows that $P(x_l = x^\star) \geq P(\tilde{d}_l = 0)$. In other words, the success probability of the Schöning process is lower bounded by the probability that the substitute process $\tilde{d}_l$ reaches 0. The fact that $(\tilde{d}_l)_l$ is Markovian makes the analysis more tractable. However, the value 0 corresponds to an absorbing boundary. (If we find the solution at an earlier stage, we should terminate the process rather than walking on). To further ease the analysis, we remove this boundary and instead introduce yet another walk $(d_l)_l$ on $\mathbb{Z}$, which we regard as "successful" whenever $d_l \leq 0$. For this process, we moreover establish the bound $P(\tilde{d}_l = 0) \geq P(d_l \leq 0)$ and thus $P(x_l = x^\star) \geq P(d_l \leq 0)$. By the trivial bound $P(d_l \leq 0) \geq P(d_l = 0)$, we thus ultimately get the bound $P(x_l = x^\star) \geq P(d_l = 0)$. For the calculation of the optimal rates, our starting point is an expression for $P(d_l = 0)$. By the inequality $P(x_l = x^\star) \geq P(d_l = 0)$, it follows that the calculated rates are upper bounds to the true rates of the Schöning process.

D. CONSTRUCTING A WALK $(\tilde{d}_l)_l$ ON $\mathbb{N}$ SUCH THAT $d_H(x_l, x^\star) \leq \tilde{d}_l$

Related to the Schöning process $(x_l)_l$, we here wish to construct another process $(\tilde{d}_l)_l$, where $\tilde{d}_l$ takes values in $\mathbb{N}$ for all $l \in \mathbb{N}$ and is such that

$$\begin{aligned} &d_H\left(x_l(a,b_1,\ldots,b_l), x^\star\right) \leq \tilde{d}_l(a,b_1,\ldots,b_l) \quad \forall a \in \{0,1\} \\ &\forall b \in \{0,1,2\}^{\times m}, \quad l = 0,1,2,\ldots,m. \end{aligned} \tag{33}$$

In other words, we want to make sure that $\tilde{d}_l$ *always* is an upper bound to the Hamming distance between x_l and $x^\star$. This requires a considerable coordination between the two processes. In particular, whenever x_l moves in the "wrong" direction (i.e., increases the Hamming distance to $x^\star$), then $\tilde{d}_l$ also has to increase. To this end, we consider the list of clauses $C_1, \ldots, C_L$. For each clause C_j, it is the case that $C_j(x^\star) = 1$. Hence, for each j, at least one of the literals

IEEE Transactions on Quantum Engineering

$l_0^{(j)}, l_1^{(j)}$, and $l_2^{(j)}$ is satisfied by $x^\star$. Among these satisfied clauses, we select one of these satisfied literals, and let $r_j \in \{0, 1, 2\}$ be its index. In other words, we are guaranteed that $l_j^{(r_j)}(x^\star) = 1$.

As already described earlier, the Schöning process $(x_l(a, b))_l$ is uniquely determined by $(a, b_1, \ldots, b_l)$ and does, in turn, uniquely determine the unsatisfied clauses C_{j_l}, as long as $x_l(a, b_1, \ldots, b_l) \neq x^\star$. Consequently, it also uniquely determines a sequence of "selected" literals r_{j_l}, whenever $x_l(a, b_1, \ldots, b_l) \neq x^\star$. For each $l \in 1, 2, \ldots$, we define a mapping $(a, b_1, \ldots b_{l-1}) \mapsto f_l(a, b_1, \ldots, b_{l-1}) \in \{0, 1, 2\}$ by

$$f_1(a) := \begin{cases} 0, & \text{if } x_0 \equiv a = x^\star \\ r_{j_1}, & \text{if } x_0 \equiv a \neq x^\star \end{cases}$$

$$f_l(a, b_1, \ldots, b_{l-1}) := \begin{cases} 0, & \text{if } x_{l-1}(a, b_1, \ldots, b_{l-1}) = x^\star, \\ r_{j_l}, & \text{if } x_{l-1}(a, b_1, \ldots, b_{l-1}) \neq x^\star. \end{cases} \quad l = 2, 3, \ldots. \tag{34}$$

The purpose of $f_l(a, b_1, \ldots, b_{l-1})$ is to determine the value of b_l that should correspond to a "successful" move for the $(\tilde{d}_l)_l$ process. More precisely, we define $(\tilde{d}_l(a, b))_l$ by (35), shown at the bottom of this page. In words, the first condition in the bracket means that 0 is an absorbing state, i.e., if $\tilde{d}_l(a, b) = 0$ for some l, then $\tilde{d}_{l'}(a, b) = 0$ for all $l' \geq l$. The other two cases make sure that $\tilde{d}_l$ moves in "coordination" with the Schöning process $(x_l(a, b))_l$, in such a manner that it is guaranteed that $d_H(x_l(a, b_1, \ldots, b_l), x^\star)$ does not increase above $\tilde{d}_l(a, b_1, \ldots, b_l)$.

Lemma 1: The Schöning process $(x_l)_{l\in\mathbb{N}}$ and the process $(\tilde{d}_l)_{l\in\mathbb{N}}$, as defined by (34) and (35) satisfy

$$d_H\left(x_l(a, b_1, \ldots, b_l), x^\star\right) \leq \tilde{d}_l(a, b_1, \ldots, b_l) \quad \forall a \in \{0, 1\}^{\times n} \quad \forall b \in \{0, 1, 2\}^{\times l}, \quad l = 0, 1, 2, \ldots. \tag{36}$$

One may note that (36) holds for every single element in the event space, and Lemma 1 does thus not depend on the actual probability distribution of $A, B_1, \ldots, B_l$. However, there are other steps in our proofs that do depend crucially on these variables being independent and uniformly distributed.

Proof: We first note that

$$x_0(a, b) = a, \quad \tilde{d}_0 = d_H(a, x^\star) \tag{37}$$

and thus (36) is satisfied for $l = 0$ for all a, b.

Now, assume that (36) holds for some $l - 1, a, b$. We have the following cases.

1) *Case $x_{l-1}(a, b) = x^\star$:* Since we assume that $x^\star$ is absorbing, it follows that $x_l(a, b) = x^\star$ and consequently $d_H(x_l(a, b_1, \ldots, b_l), x^\star) = 0$. Concerning $\tilde{d}_{l-1}$, we can distinguish yet two subcases.
   a) *Case $\tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) = 0$:* By construction [first case in (35)], $\tilde{d}_l(a, b_1, \ldots, b_l) = 0$, and (36) is thus satisfied for l, a, b.
   b) *Case $\tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \neq 0$:* Then, $\tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \geq 1$. Since the process d can change at most one step, it follows that $\tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \geq 0$, and thus, (36) is satisfied for l, a, b.
2) *Case $x_{l-1}(a, b) \neq x^\star$:* Since we assume that (36) holds for $l - 1, a, b$, it follows that $\tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \geq 1$. Moreover, since $x_{l-1}(a, b) \neq x^\star$, we have $f_l(a, b_1, \ldots, b_{l-1}) = r_{j_l}$. Again, we can distinguish two subcases.
   a) *Case $f_l(a, b_1, \ldots, b_{l-1}) = b_l$:* In this case, the d-process decreases one step. However, by construction, r_{j_l} is one of the "successful" flips for the Schöning process; hence, the x-process also decreases one step. By assumption, the inequality (36) is satisfied for $l - 1, a, b$, and since both the x-process and the d-process decrease one step, (36) remains satisfied for l, a, b.
   b) *Case $f_l(a, b_1, \ldots, b_{l-1}) \neq b_l$:* In this case, the d-process increases one step. The x-process may increase or decrease, but with at most one step, so (36) remains satisfied for l, a, and b.

By induction, we can conclude that (36) is satisfied for all l, a, b. ■

E. $(\tilde{d}_l)_l$ IS A MARKOV CHAIN

In the following, we wish to show that $(\tilde{d}_l)_l$ is a Markov chain. Recall that both the genuine Schöning process $(x_l)_l$, as well as the walk $(\tilde{d}_l)_l$, are determined by a sequence of "walk variables" $(B_l)_l$ (and initial-state variable A). The Schöning walk itself is Markovian, but it is a priori not obvious that the process $(\tilde{d}_l)_l$ is also Markovian, particularly since the lth step of the latter is determined by a complicated function of all the walk variables up to the lth step, as described by (35). However, in spite appearances, it turns out that (35) defines a mapping from the original set of random variables $(B_l)_l$ to a new set of variables $(\tilde{B}_l)_l$, in such a manner that the change from

$$\tilde{d}_0(a, b) := d_H\left(x_0(a, b), x^\star\right) = d_H(a, x^\star)$$

$$\tilde{d}_l(a, b_1, \ldots, b_l) := \begin{cases} 0 & \text{if } \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) = 0 \\ \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) + 1 & \text{if } \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \neq 0, \quad b_l \neq f_l(a, b_1, \ldots, b_{l-1}) \\ \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) - 1 & \text{if } \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \neq 0, \quad b_l = f_l(a, b_1, \ldots, b_{l-1}) \end{cases} \quad l = 1, 2, \ldots \tag{35}$$

$\tilde{d}_{m-1}$ to $\tilde{d}_m$ is determined by $\tilde{B}_m$, and *only* by $\tilde{B}_m$. Moreover, it turns out that $(\tilde{B}_l)_l$ is an independent identically distributed (i.i.d.) sequence. Since all $\tilde{B}_l$ are independent, it follows that $(\tilde{d}_l)_l$ must be a Markov chain. In order to show that the new sequence of variables $(\tilde{B}_l)_l$ is i.i.d., what we actually do is to show that (35) induces a bijection on $\{0,1\}^{\times n} \times \{0,1,2\}^l$. Since $(A, (B_l)_l)$ is uniformly distributed [see (30)], it follows by the bijection that $(A, (\tilde{B}_l)_l)$ is also uniformly distributed and, thus, in particular that $(\tilde{B}_l)_l$ is i.i.d.

F. BIJECTION

The following lemma introduces functions f_s. Later, in the proof of Proposition 6, we will let these mappings be the functions $f_l(a, b_1, \ldots, b_{l-1})$ in (34). Since the latter are algorithmically defined, via the Schöning process $(x_l)_l$, it is challenging to get a hold on the properties of these mappings. It is thus worth noting that (apart from domains and ranges) Lemma 3 (and Lemma 5) makes no assumptions on the properties of the mappings f_s. Hence, our lack of control over the mappings $f_l(a, b_1, \ldots, b_{l-1})$ will not be an issue in the subsequent proofs.

As preparation, we make the following observations.

Lemma 2: If $t, t', r \in \{0,1,2\}$, then

$$(t-r) \bmod 3 = (t'-r) \bmod 3 \quad \Leftrightarrow \quad t = t'. \tag{38}$$

Moreover, if $t, r \in \{0,1,2\}$, then

$$((t+r) \bmod 3 - r) \bmod 3 = t. \tag{39}$$

Lemma 3: Let $f_1 : \{0,1\}^{\times n} \to \{0,1,2\}$ and $f_s : \{0,1\}^{\times n} \times \{0,1,2\}^{\times(s-1)} \to \{0,1,2\}$ for $s = 2, \ldots, l$ be given. Define the mapping $\{0,1\}^{\times n} \times \{0,1,2\}^{\times l} \ni (b_1, \ldots, b_l) \mapsto Q(a, b_1, \ldots, b_l) = (\tilde{a}, \tilde{b}_1, \ldots, \tilde{b}_l) \in \{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$ by

$$\begin{aligned}
\tilde{a} &:= a \\
\tilde{b}_1 &:= (b_1 - f_1(a)) \bmod 3 \\
\tilde{b}_2 &:= (b_2 - f_2(a, b_1)) \bmod 3 \\
\tilde{b}_3 &:= (b_3 - f_3(a, b_1, b_2)) \bmod 3 \\
&\vdots \\
\tilde{b}_l &:= (b_l - f_l(a, b_1, \ldots, b_{l-1})) \bmod 3.
\end{aligned} \tag{40}$$

Then, Q is a bijection on $\{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$.

Proof: To show that Q is a bijection, we first show that it is injective and then that it is surjective.

Let $(a, b_1, \ldots, b_l), (a', b'_1, \ldots, b'_l) \in \{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$ be such that

$$Q(a, b_1, \ldots, b_l) = Q(a', b'_1, \ldots, b'_l). \tag{41}$$

By the first line of (40), it follows that

$$a = \tilde{a} = a'. \tag{42}$$

By the second line of (40), it follows that

$$(b_1 - f_1(a)) \bmod 3 = \big(b'_1 - f_1(a')\big) \bmod 3 \tag{43}$$

which combined with (42) yields

$$(b_1 - f_1(a)) \bmod 3 = \big(b'_1 - f_1(a)\big) \bmod 3. \tag{44}$$

Since $f_1(a), b_1, b'_1 \in \{0,1,2\}$, it follows by (38) that

$$b_1 = b'_1. \tag{45}$$

As an induction hypothesis, assume that for some $s \geq 2$, it is the case that

$$a = a' \quad b_j = b'_j, \quad j = 1, \ldots, s-1. \tag{46}$$

The sth line of (41) implies that

$$\begin{aligned}
&(b_s - f_s(a, b_1, \ldots, b_{s-1})) \bmod 3 \\
&= \big(b'_s - f_s(a', b'_1, \ldots, b'_{s-1})\big) \bmod 3.
\end{aligned} \tag{47}$$

By the induction hypothesis, this implies that

$$\begin{aligned}
&(b_s - f_s(a, b_1, \ldots, b_{s-1})) \bmod 3 \\
&= \big(b'_s - f_s(a, b_1, \ldots, b_{s-1})\big) \bmod 3.
\end{aligned} \tag{48}$$

Since $b_s, b'_s, f_s(a, b_1, \ldots, b_{s-1}) \in \{0,1,2\}$, it follows by (38) that

$$b_s = b'_s. \tag{49}$$

Since the induction hypothesis is true for $s = 2$, we can conclude that it is true for all $s = 2, \ldots, l$. We can thus conclude that Q is injective.

Next, we wish to show that Q is surjective onto $\{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$. Let $(\tilde{a}', \tilde{b}'_1, \ldots, \tilde{b}'_l) \in \{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$. Define

$$\begin{aligned}
a &:= \tilde{a}' \\
b_1 &:= \big(\tilde{b}'_1 + f_1(\tilde{a}')\big) \bmod 3 = \big(\tilde{b}'_1 + f_1(\tilde{a}')\big) \bmod 3
\end{aligned} \tag{50}$$

and the sequence $(b_j)_{j=2}^l$ recursively by

$$b_j := \Big(\tilde{b}'_j + f_j(a, b_{j-1}, \ldots, b_1)\Big) \bmod 3, \quad j = 2, \ldots, l \tag{51}$$

for a' and b'_1, as defined in (50). In the following, we wish to show that $Q(a, b_1, \ldots, b_l) = (\tilde{a}', \tilde{b}'_1, \ldots, \tilde{b}'_l)$. For notational convenience, we introduce the components $Q_0(a, b_1, \ldots, b_l) := \tilde{a}$ and $Q_j(a, b_1, \ldots, b_l) := \tilde{b}_j$, $j = 2, \ldots, l$.

By the first line of (40), we have

$$Q_0(a, b_1, \ldots, b_l) = a = \tilde{a}'. \tag{52}$$

By the second line of (40), we have

$$\begin{aligned}
Q_1(a, b_1, \ldots, b_l) &= (b_1 - f_1(a)) \bmod 3 \\
&= \big(\big(\tilde{b}'_1 + f_1(\tilde{a}')\big) \bmod 3 - f_1(a)\big) \bmod 3 \\
&= \big(\big(\tilde{b}'_1 + f_1(\tilde{a}')\big) \bmod 3 - f_1(\tilde{a}')\big) \bmod 3 \\
&\quad \text{[By (39)]} \\
&= \tilde{b}'_1.
\end{aligned} \tag{53}$$

For all $j \geq 2$, we moreover have

$$\begin{aligned} & Q_j(a, b_1, \ldots, b_l) \\ = & \left(b_j - f_j(a, b_1, \ldots, b_{j-1})\right) \bmod 3 \\ & \text{[by (51)]} \\ = & \left(\left(\tilde{b}'_j + f_j(a, b_{j-1}, \ldots, b_1)\right) - f_j(a, b_1, \ldots, b_{j-1})\right) \bmod 3 \\ & \text{[by (39)]} \\ = & \tilde{b}'_j. \end{aligned} \tag{54}$$

Hence, we can conclude that $Q(a, b_1, \ldots, b_l) = (\tilde{a}', \tilde{b}'_1, \ldots, \tilde{b}'_l)$. Hence, Q is surjective and thus bijective. ■

G. TRANSFORMATIONS THAT PRESERVE UNIFORMITY

We make the following basic observation.

Lemma 4: Let $\mathcal{S}$ be some finite set. Let $Q : \mathcal{S} \to \mathcal{S}$ be invertible. Let R be some random variable on $\mathcal{S}$. If R is uniformly distributed over $\mathcal{S}$, $Q(R)$ is also uniformly distributed over $\mathcal{S}$.

Proof:

$$P\left(Q(R) = s\right) = P\left(R = Q^{-1}(s)\right) = \frac{1}{|\mathcal{S}|}. \tag{55}$$

■

Lemma 5: Let $f_1 : \{0,1\}^{\times n} \to \{0,1,2\}$ and $f_s : \{0,1\}^{\times n} \times \{0,1,2\}^{\times(s-1)} \to \{0,1,2\}$ for $s = 2, \ldots, l$ be given. Assume that $B_1, \ldots, B_l$ are random variables that take values in $\{0,1,2\}$, A be a random variable that takes values in $\{0,1\}^{\times n}$, and that these are distributed as

$$P(A = a, B_1 = b_1, \ldots, B_l = b_l) = \frac{1}{2^n 3^l} \quad \forall a \in \{0,1\}^{\times n} \quad \forall (b_1, \ldots, b_l) \in \{0,1,2\}^{\times l}. \tag{56}$$

Define $\tilde{B}_1, \ldots, \tilde{B}_l$ by

$$\begin{aligned} \tilde{B}_1 &:= (B_1 - f_1(A)) \bmod 3 \\ \tilde{B}_2 &:= (B_2 - f_2(A, B_1)) \bmod 3 \\ \tilde{B}_3 &:= (B_3 - f_3(A, B_1, B_2)) \bmod 3 \\ &\vdots \\ \tilde{B}_l &:= (B_l - f_l(A, B_1, \ldots, B_{l-1})) \bmod 3. \end{aligned} \tag{57}$$

Then

$$P(A = a, \tilde{B}_1 = \tilde{b}_1, \ldots, \tilde{B}_l = \tilde{b}_l) = \frac{1}{2^n 3^l} \quad \forall a \in \{0,1\}^{\times n} \quad \forall (\tilde{b}_1, \ldots, \tilde{b}_l) \in \{0,1,2\}^{\times l}. \tag{58}$$

Consequently, $A, \tilde{B}_1, \ldots, \tilde{B}_l$ are independent and uniformly distributed.

Proof: By (56), we know that $(A, B_1, \ldots, B_l)$ is uniformly distributed on $\{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$. With the additional definition $\tilde{A} := A$, we note that (57) can be rewritten as

$$(\tilde{A}, \tilde{B}_1, \ldots, \tilde{B}_l) := Q(A, B_1, \ldots, B_l) \tag{59}$$

where $Q : \{0,1\}^{\times n} \times \{0,1,2\}^{\times l} \to \{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$ is as defined in Lemma 3. By Lemma 3, we moreover know that Q is a bijection on $\{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$ and thus invertible. Hence, by Lemma 58, we know that $(\tilde{A}, \tilde{B}_1, \ldots, \tilde{B}_l)$ is also uniformly distributed over $\{0,1\}^{\times n} \times \{0,1,2\}^{\times l}$. Since $\tilde{A} = A$, we can conclude that (58) holds.

By (58), it follows that

$$\begin{aligned} & P(A = a, \tilde{B}_1 = \tilde{b}_1, \ldots, \tilde{B}_l = \tilde{b}_l) = P(A = a) P(\tilde{B}_1 = \tilde{b}_1) \\ & \cdots P(\tilde{B}_l = \tilde{b}_l) \\ & P(A = a) = \frac{1}{2^n} \quad P(\tilde{B}_1 = \tilde{b}_1) = \frac{1}{3}, \ldots, P(\tilde{B}_l = \tilde{b}_l) = \frac{1}{3} \end{aligned} \tag{60}$$

and thus A and $\tilde{B}_1, \ldots, \tilde{B}_l$ are independent and uniformly distributed. ■

H. PROCESS $(\tilde{d}_l)_l$ IS A MARKOV CHAIN

Proposition 6: Let $(\tilde{d}_l)_l$ be the process as defined by (34) and (35), with respect to the variables $A, B_1, B_2, \ldots$ distributed as in (30). For each m, there exist variables $\tilde{B}_1, \ldots, \tilde{B}_m$ that are i.i.d. and uniformly distributed on $\{0,1,2\}$ and are independent of A, such that

$$\tilde{d}_0 := d_H(A, x^\star),$$

$$\tilde{d}_l := \begin{cases} 0, & \text{if } \tilde{d}_{l-1} = 0 \\ \tilde{d}_{l-1} + 1, & \text{if } \tilde{d}_{l-1} \neq 0, \quad \tilde{B}_l \neq 0 \\ \tilde{d}_{l-1} - 1, & \text{if } \tilde{d}_{l-1} \neq 0, \quad \tilde{B}_l = 0 \end{cases} \quad l = 1, 2, \ldots, m. \tag{61}$$

Hence, $(\tilde{d}_l)_l$ is a Markov chain described by the transition probabilities

$$\begin{aligned} & P(\tilde{d}_{l+1} = j | \tilde{d}_l = k) = \delta_{j,0}\delta_{k,0} \\ & + (1 - \delta_{k,0}) \left(\frac{1}{3}\delta_{j,k-1} + \frac{2}{3}\delta_{j,k+1}\right) \quad \forall j, k \in \mathbb{N} \quad \forall l \end{aligned} \tag{62}$$

with initial distribution

$$P(\tilde{d}_0 = j) = P\left(d_H(A, x^\star) = j\right). \tag{63}$$

Moreover, for the distribution of A as in (30), we have

$$P(\tilde{d}_0 = j) = \begin{cases} \frac{1}{2^n}\binom{n}{j}, & 0 \leq j \leq n \\ 0, & \text{otherwise.} \end{cases} \tag{64}$$

In (62), the term $\delta_{j,0}\delta_{k,0}$ signifies $d = 0$ being an absorbing state. In the second term, the effect of the factor $(1 - \delta_{k,0})$ is that if the chain is not in the absorbing state, then the transition probabilities are given by $\frac{1}{3}\delta_{j,k-1} + \frac{2}{3}\delta_{j,k+1}$. Hence, with probability 1/3, it takes a step "down," and with probability 2/3, it takes a step "up."

Proof: For $t, r \in \{0,1,2\}$, it is the case that

$$t = r \quad \Leftrightarrow \quad (t - r) \bmod 3 = 0. \tag{65}$$

By this observation, it follows that (35) can be rewritten as (66) shown at the bottom of this page. Next, we rewrite (66) as (67), shown at the bottom of this page, such that we suppress the explicit dependence on the elementary events (a, b). If we define $\tilde{B}_1, \ldots, \tilde{B}_l$ by

$$\begin{aligned}\tilde{B}_1 &:= f_1(A)\\ \tilde{B}_2 &:= (B_2 - f_2(A, B_1)) \bmod 3\\ \tilde{B}_3 &:= (B_3 - f_3(A, B_1, B_2)) \bmod 3\\ &\vdots\\ \tilde{B}_l &:= (B_l - f_l(A, B_1, \ldots, B_{l-1})) \bmod 3\end{aligned} \tag{68}$$

then we can rewrite (67) as

$$\tilde{d}_0 := d_H(A, x^\star)$$
$$\tilde{d}_l := \begin{cases} 0, & \text{if } \tilde{d}_{l-1} = 0 \\ \tilde{d}_{l-1} + 1, & \text{if } \tilde{d}_{l-1} \neq 0, \quad \tilde{B}_l \neq 0 \\ \tilde{d}_{l-1} - 1, & \text{if } \tilde{d}_{l-1} \neq 0, \quad \tilde{B}_l = 0 \end{cases} \quad l = 1, 2, \ldots. \tag{69}$$

By Lemma 5, we know that $A, \tilde{B}_1, \ldots, \tilde{B}_l$ are independent and uniformly distributed. Since the lth step is determined solely by $\tilde{B}_l$, and these are independent of each other, and of A, it follows that $(\tilde{d}_l)_l$ is a Markov chain. By inspecting (69), we first see that

$$P(\tilde{d}_l = j | \tilde{d}_{l-1} = 0) = \delta_{j,0} \tag{70}$$

while for $\tilde{d}_{l-1} = k \neq 0$, we have

$$\begin{aligned}P(\tilde{d}_l = j | \tilde{d}_{l-1} = k) &= \delta_{j,k+1} P(\tilde{B}_l \neq 0) + \delta_{j,k-1} P(\tilde{B}_l = 0)\\ &= \frac{2}{3}\delta_{j,k+1} + \frac{1}{3}\delta_{j,k-1}\end{aligned} \tag{71}$$

where the last step follows since each $\tilde{B}_l$ is uniformly distributed over $\{0, 1, 2\}$. By combining the cases (70) and (71), we obtain (62). By (69), it moreover follows that $P(\tilde{d}_0 = j) = P(d_H(A, x^\star) = j)$. Since A is uniformly distributed over $\{0, 1\}^{\times n}$, it means that $d_H(A, x^\star)$ is binomially distributed. Thus, for $0 \leq j \leq n$, we have $P(\tilde{d}_0 = j) = \frac{1}{2^n}\binom{n}{j}$. ■

I. RELATING PROBABILITIES OF $(x_l)_l$ AND $(\tilde{d}_l)_l$

The reason why we introduce the walk $(\tilde{d}_l)_l$ is in order to bound the relevant success probabilities of the more complicated true Schöning walk $(x_l)_l$. The following lemma considers two such inequalities, which we will use when we determine the bounds for the GW.

Lemma 7: Let $(x_l)_{l\in\mathbb{N}}$ be the Schöning process for bit strings of length n, with $x^\star$ being the selected satisfying assignment. Let $(\tilde{d}_l)_l$ be the process as defined by (34) and (35). Then

$$P(x_m = x^\star) \geq P(\tilde{d}_m = 0) \tag{72}$$
$$P\left(x_m = x^\star | d_H(x_0, x^\star) = j\right) \geq P(\tilde{d}_m = 0 | \tilde{d}_0 = j). \tag{73}$$

Proof: We begin by proving inequality (72). For the sake of notational simplicity, we let ω denote the elements of the event space [where we could regard ω as (a, b) or $(a, \tilde{b})$]. By Lemma 1, we know that

$$d_H\left(x_m(\omega), x^\star\right) \leq \tilde{d}_m(\omega) \tag{74}$$

which implies that

$$\tilde{d}_m(\omega) = 0 \quad \Rightarrow \quad d_H\left(x_m(\omega), x^\star\right) = 0 \tag{75}$$

and thus

$$\begin{aligned}\{\omega : \tilde{d}_m(\omega) = 0\} &\subset \{\omega : d_H\left(x_m(\omega), x^\star\right) = 0\}\\ &= \{\omega : x_m(\omega) = x^\star\}\end{aligned} \tag{76}$$

and thus

$$\begin{aligned}P(\tilde{d}_m = 0) &= P\left(\{\omega : \tilde{d}_m(\omega) = 0\}\right)\\ &\leq P\left(\{\omega : x_m(\omega) = x^\star\}\right)\\ &= P(x_m = x^\star)\end{aligned} \tag{77}$$

which proves (72).

$$\begin{aligned}&\tilde{d}_0(a, b) := d_H(a, x^\star)\\ &\tilde{d}_l(a, b_1, \ldots, b_l)\\ &:= \begin{cases} 0, & \text{if } \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) = 0 \\ \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) + 1, & \text{if } \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \neq 0, \quad (b_l - f_l(a, b_1, \ldots, b_{l-1})) \bmod 3 \neq 0 \\ \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) - 1, & \text{if } \tilde{d}_{l-1}(a, b_1, \ldots, b_{l-1}) \neq 0, \quad (b_l - f_l(a, b_1, \ldots, b_{l-1})) \bmod 3 = 0. \end{cases} \quad l = 1, 2, \ldots\end{aligned} \tag{66}$$

$$\tilde{d}_0 := d_H(A, x^\star)$$
$$\tilde{d}_l := \begin{cases} 0, & \text{if } \tilde{d}_{l-1} = 0 \\ \tilde{d}_{l-1} + 1, & \text{if } \tilde{d}_{l-1} \neq 0, \quad (B_l - f_l(A, B_1, \ldots, B_{l-1})) \bmod 3 \neq 0 \\ \tilde{d}_{l-1} - 1, & \text{if } \tilde{d}_{l-1} \neq 0, \quad (B_l - f_l(A, B_1, \ldots, B_{l-1})) \bmod 3 = 0 \end{cases} \quad l = 1, 2, \ldots \tag{67}$$

We next turn to the proof of (73). By definition of the walk $(\tilde{d}_l)_l$, we have $\tilde{d}_0(\omega) = d_H(x_0(\omega), x^\star)$ and thus

$$\{\omega : \tilde{d}_0(\omega) = j\} = \{\omega : d_H\left(x_0(\omega), x^\star\right) = j\} \tag{78}$$

and consequently

$$P(\tilde{d}_0 = j) = P\left(d_H(x_0, x^\star) = j\right). \tag{79}$$

By combining (76) and (78), we obtain

$$\begin{aligned}\{\omega : \tilde{d}_m(\omega) = 0\} \cap \{\omega : \tilde{d}_0(\omega) = j\} \subset \{\omega : x_m(\omega) \\ = x^\star\} \cap \{\omega : d_H\left(x_0(\omega), x^\star\right) = j\}\end{aligned} \tag{80}$$

and consequently

$$\begin{aligned}&P(\tilde{d}_m = 0, \tilde{d}_0 = j) \\ &= P\left(\{\omega : \tilde{d}_m(\omega) = 0\} \cap \{\omega : \tilde{d}_0(\omega) = j\}\right) \\ &\leq P\left(\{\omega : x_m(\omega) = x^\star\} \cap \{\omega : d_H\left(x_0(\omega), x^\star\right) = j\}\right) \\ &= P\left(x_m = x^\star, d_H(x_0, x^\star) = j\right).\end{aligned} \tag{81}$$

By combining this with (79), we can conclude that

$$P(\tilde{d}_m = 0 | \tilde{d}_0 = j) \leq P\left(x_m = x^\star | d_H(x_0, x^\star) = j\right) \tag{82}$$

which proves (73). ■

J. FROM WALKS ON $\mathbb{N}$ TO WALKS ON $\mathbb{Z}$

So far, we have replaced the projection of the Schöning process $(x_l)_l$ to the Hamming distance $d_H(x_m, x^\star)$ with the substitute Markov chain $(\tilde{d}_l)_l$. Similar to $x^\star$ being an absorbing state of (x_l), the process $(\tilde{d}_l)_l$ has 0 as the absorbing state. As a model of the true Schöning process, this absorbing state certainly makes sense, since it corresponds to a setting where we, at each step, monitor whether a solution has been reached, and the process is terminated once this happens. For the sake of obtaining tractable expressions for the relevant probabilities, we here take one step further and instead consider a walk on $\mathbb{Z}$. Analogously to how Lemma 7 bounds the relevant probabilities of the true Schöning process, with the corresponding quantities in $(\tilde{d}_l)_l$, Lemma 8 bounds the relevant probabilities of $(\tilde{d}_l)_l$ in terms of corresponding quantities for a Markov chain $(d_l)_l$ extended to the whole of $\mathbb{Z}$.

As a bit of a side remark, one may note that the results in (8) do not necessarily refer to the particular Markov chain defined by (34) and (35), but could be any Markov chain on $\mathbb{N}$ with fixed transition probabilities and absorbing boundary condition at 0.

Lemma 8: Let $(\tilde{d}_l)_{l\in\mathbb{N}}$ be a Markov chain on $\mathbb{N}$, with transition probabilities

$$\begin{aligned}P(\tilde{d}_{l+1} = j | \tilde{d}_l = k) = \delta_{j,0}\delta_{k,0} \\ +(1-\delta_{k,0})\left((1-q)\delta_{j,k-1} + q\delta_{j,k+1}\right) \forall j, k \in \mathbb{N} \quad \forall l \in \mathbb{N}\end{aligned} \tag{83}$$

for some $0 \leq q \leq 1$. Let $(d_l)_{l\in\mathbb{N}}$ be a Markov chain on $\mathbb{Z}$, with transition probabilities

$$P(d_{l+1} = j | d_l = k) = (1-q)\delta_{j,k-1} + q\delta_{j,k+1} \quad \forall j, k \in \mathbb{Z} \quad \forall l \in \mathbb{N}. \tag{84}$$

Then

$$P(d_m \leq 0 | d_0 = j) \leq P(\tilde{d}_m = 0 | \tilde{d}_0 = j) \quad \forall m \in \mathbb{N} \quad \forall j \in \mathbb{N}. \tag{85}$$

Consequently, if the initial state d_0 is such that

$$P(d_0 = j) = \begin{cases} P(\tilde{d}_0 = j), & j \geq 0 \\ 0, & j < 0 \end{cases} \tag{86}$$

then

$$P(d_m = 0) \leq P(d_m \leq 0) \leq P(\tilde{d}_m = 0) \quad \forall m \in \mathbb{N}. \tag{87}$$

Proof: For notational convenience, we define

$$M_{j,k} := P(\tilde{d}_{l+1} = j | \tilde{d}_l = k) \tag{88}$$

and

$$\tilde{M}_{j,k} := P(d_{l+1} = j | d_l = k). \tag{89}$$

By comparing with (83) and (84), one can see that

$$M_{j,k} = \tilde{M}_{j,k} \quad \forall j > 0 \quad \forall k > 0. \tag{90}$$

We note that 0 is an absorbing state for $(\tilde{d}_l)_l$. Hence

$$\tilde{d}_{s-1} = 0 \quad \Rightarrow \quad \tilde{d}_s = 0 \tag{91}$$

which implies that

$$\tilde{d}_s > 0 \quad \Rightarrow \quad \tilde{d}_{s-1} > 0 \tag{92}$$

which, in turn, implies that

$$P(\tilde{d}_s = k_s | \tilde{d}_{s-1} = 0) = 0, \quad \text{if} \quad k_s > 0. \tag{93}$$

We begin by proving (85). For this purpose, assume that $j > 0$.

$$\begin{aligned}&P(\tilde{d}_l > 0 | \tilde{d}_0 = j) \\ &= \sum_{k_l > 0} P(\tilde{d}_l = k_l | \tilde{d}_0 = j) \\ &\quad \text{[By Markovianity]} \\ &= \sum_{k_l > 0} \sum_{k_{l-1}, \ldots, k_1} P(\tilde{d}_l = k_l | \tilde{d}_{l-1} = k_{l-1}) \\ &\quad P(\tilde{d}_{l-1} = k_{l-1} | \tilde{d}_{l-2} = k_{l-2}) \cdots P(\tilde{d}_2 = k_2 | \tilde{d}_1 = k_1) \\ &\quad P(\tilde{d}_1 = k_1 | \tilde{d}_0 = j) \\ &= \sum_{k_l > 0} \sum_{k_{l-1} : k_{l-1} > 0} \sum_{k_{l-2}, \ldots, k_1} P(\tilde{d}_l = k_l | \tilde{d}_{l-1} = k_{l-1}) \\ &\quad P(\tilde{d}_{l-1} = k_{l-1} | \tilde{d}_{l-2} = k_{l-2}) \cdots P(\tilde{d}_1 = k_1 | \tilde{d}_0 = j) \\ &\quad + \sum_{k_l > 0} \sum_{k_{l-2}, \ldots, k_1} P(\tilde{d}_l = k_l | \tilde{d}_{l-1} = 0) \\ &\quad P(\tilde{d}_{l-1} = 0 | \tilde{d}_{l-2} = k_{l-2}) \cdots P(\tilde{d}_1 = k_1 | \tilde{d}_0 = j)\end{aligned}$$

[Since $k_l > 0$, it follows by (93) that

$P(\tilde{d}_l = k_l | \tilde{d}_{l-1} = 0) = 0.$]

$$= \sum_{k_l>0} \sum_{k_{l-1}:k_{l-1}>0} \sum_{k_{l-2},\ldots,k_1}$$

$$P(\tilde{d}_l = k_l | \tilde{d}_{l-1} = k_{l-1})$$

$$P(\tilde{d}_{l-1} = k_{l-1} | \tilde{d}_{l-2} = k_{l-2}) \cdots P(\tilde{d}_1 = k_1 | \tilde{d}_0 = j)$$

[By iteration]

$$= \sum_{k_l>0} \sum_{k_{l-1},\ldots,k_1:k_{l-1}>0,\ldots,k_1>0} P(\tilde{d}_l = k_l | \tilde{d}_{l-1} = k_{l-1})$$

$$P(\tilde{d}_{l-1} = k_{l-1} | \tilde{d}_{l-2} = k_{l-2}) \cdots P(\tilde{d}_1 = k_1 | \tilde{d}_0 = j)$$

[By (88)]

$$= \sum_{k_l>0} \sum_{k_{l-1},\ldots,k_1:k_{l-1}>0,\ldots,k_1>0} M_{k_l,k_{l-1}} \cdots M_{k_1,j}$$

[Since $k_l > 0,\ k_{l-1} > 0, \ldots, k_1 > 0, j > 0,$

it follows by (90) that]

$$= \sum_{k_l>0} \sum_{k_{l-1},\ldots,k_1:k_{l-1}>0,\ldots,k_1>0} \tilde{M}_{k_l,k_{l-1}} \cdots \tilde{M}_{k_1,j}$$

[$\tilde{M}_{k_l,k_{l-1}} \geq 0$]

$$\leq \sum_{k_l>0} \sum_{k_{l-1},\ldots,k_1} \tilde{M}_{k_l,k_{l-1}} \cdots \tilde{M}_{k_1,j}$$

[By (89)]

$$= \sum_{k_l>0} \sum_{k_{l-1},\ldots,k_1} P(d_l = k_l | d_{l-1} = k_{l-1})$$

$$P(d_{l-1} = k_{l-1} | d_{l-2} = k_{l-2}) \cdots P(d_1 = k_1 | d_0 = j)$$

[By Markovianity]

$$= \sum_{k_l>0} P(d_l = k_l | d_0 = j)$$

$$= P(d_l > 0 | d_0 = j). \tag{94}$$

Consequently

$$\begin{aligned} P(\tilde{d}_l = 0 | \tilde{d}_0 = j) &= 1 - P(\tilde{d}_l > 0 | \tilde{d}_0 = j) \\ &\geq 1 - P(d_l > 0 | d_0 = j) \\ &= P(d_l \leq 0 | d_0 = j), \quad j > 0. \end{aligned} \tag{95}$$

In the case $\tilde{d}_0 = 0$, we know that this is an absorbing state, and thus, $P(\tilde{d}_l = 0 | \tilde{d}_0 = 0) = 1$. Consequently, $P(d_l \leq 0 | d_0 = 0) \leq P(\tilde{d}_l = 0 | \tilde{d}_0 = 0) = 1$. This thus proves the inequality in (85).

With the initial distribution (86), we find

$$\begin{aligned} P(d_l \leq 0) &= \sum_{j\in\mathbb{Z}} P(d_l \leq 0 | d_0 = j) P(d_0 = j) \\ &= \sum_{j\geq 0} P(d_l \leq 0 | d_0 = j) P(\tilde{d}_0 = j) \\ &\leq \sum_{j\geq 0} P(\tilde{d}_l = 0 | \tilde{d}_0 = j) P(\tilde{d}_0 = j) \\ &= P(\tilde{d}_l = 0). \end{aligned} \tag{96}$$

■

K. BOUNDS FOR SCHÖNING WALKS AND GWS

Here, we combine the previous observations in order to obtain the following lower bounds on the success probability of the Schöning process. We also obtain the inequalities needed for determining the desired bound on the success probability of the GW.

Proposition 9: Let $(x_l)_{l\in\mathbb{N}}$ be the Schöning process for bit strings of length n, with $x^\star$ being the selected satisfying assignment. Let $(\tilde{d}_l)_l$ be the process as defined by (34) and (35). Let $(d_l)_l$ be the Markov chain as defined by the transition probabilities (84) for $q = 2/3$ in Lemma 8, for the initial state

$$P(d_0 = j) = \begin{cases} P(\tilde{d}_0 = j) = P\left(d_H(x_0, x^\star) = j\right) = \frac{1}{2^n}\binom{n}{j}, & n \geq j \geq 0 \\ 0, & \text{otherwise.} \end{cases} \tag{97}$$

Then

$$\begin{aligned} P(x_m = x^\star) &\geq P(\tilde{d}_m = 0) \\ &\geq P(d_m \leq 0) \\ &= \sum_{\substack{j,l: j+m-2l\leq 0, \\ 0\leq j\leq n, \\ 0\leq l\leq m}} \frac{1}{2^n}\binom{n}{j}\binom{m}{l}\left(\frac{1}{3}\right)^l\left(\frac{2}{3}\right)^{m-l} \end{aligned} \tag{98}$$

and

$$\begin{aligned} P\left(x_m = x^\star \middle| d_H(x_0|x^\star) = j\right) &\geq P(\tilde{d}_m = 0 | \tilde{d}_0 = j) \\ &\geq P(d_m \leq 0 | d_0 = j) \\ &= \sum_{\substack{l: j+m-2l\leq 0, \\ 0\leq l\leq m}} \binom{m}{l}\left(\frac{1}{3}\right)^l\left(\frac{2}{3}\right)^{m-l}. \end{aligned} \tag{99}$$

Proof: Equation (72) in Lemma 7 yields the first inequality in (98).

By Proposition 6, we know that $(\tilde{d}_l)_l$ is a Markov chain with transition probabilities as in (62) and initial distribution (64). By these observations, it follows that the second inequality in (98) is a direct application of (87) in Lemma 8. By Lemma 8, we also know that the Markov chain $(d_l)_l$ defined by the transition probabilities

$$P(d_{l+1}=j | d_l=k) = \frac{1}{3}\delta_{j,k-1} + \frac{2}{3}\delta_{j,k+1} \quad \forall j, k \in\mathbb{Z} \quad \forall l \in\mathbb{N} \tag{100}$$

and initial distribution

$$P(d_0{=}j){=}\begin{cases} P(\tilde{d}_0=j), & j\geq 0 \\ 0, & j<0 \end{cases} {=}\begin{cases} \frac{1}{2^n}\binom{n}{j}, & 0\leq j\leq n \\ 0, & \text{otherwise.} \end{cases} \tag{101}$$

From (100), it follows that

$$P(d_m\leq 0|d_0=j)=\sum_{l:j+m-2l\leq 0,\ 0\leq l\leq m}\binom{m}{l}\left(\frac{1}{3}\right)^l\left(\frac{2}{3}\right)^{m-l} \tag{102}$$

and thus

$$\begin{aligned} P(d_m\leq 0) &= \sum_j P(d_m\leq 0|d_0=j)P(d_0=j) \\ &\quad \text{[By (101)]} \\ &= \sum_{\substack{j,l:j+m-2l\leq 0,\\ 0\leq j\leq n,\\ 0\leq l\leq m}} \frac{1}{2^n}\binom{n}{j}\binom{m}{l}\left(\frac{1}{3}\right)^l\left(\frac{2}{3}\right)^{m-l}. \end{aligned} \tag{103}$$

Next, we turn to the inequalities in (99). Inequality (73) in Lemma 7 yields the first inequality in (99). The second inequality in (99) is a direct application of (85) in Lemma (8). By (102), we already know the last equality in (99). ■

L. RELATION TO THE LEADING-ORDER ANALYSIS OF THE SCHÖNING AND GW PROCESSES

Here, we connect to the analysis of the asymptotic scaling in the main text by obtaining the starting points, so to speak, of the leading-order analysis of the Schöning process and the GW process.

1. SCHÖNING PROCESS

For the Schöning process, the average number of repetitions needed to find a solution is given by

$$N_{\text{Schöning}}=\frac{1}{P(x_m=x^\star)}. \tag{104}$$

By sequences of lower bounds on the ideal success probability $P(x_m=x^\star)$, we thus obtain upper bounds on $N_{\text{Schöning}}$. The step from the true Schöning process to the walk on Z corresponds to one such inequality, i.e.,

$$P(x_m=x^\star)\geq P(d_m\leq 0) \tag{105}$$

in (98) in Proposition (9). The leading-order analysis in the main text is based on further such inequalities, with the rationale that the "loss" of probability weight becomes irrelevant for the rates $\gamma=\lim_{n\to\infty}\frac{1}{n}\log N_{\text{Schöning}}$, if the inequalities are chosen to correspond to the leading-order contributions. As a first step along these lines, we restrict to an event where we not only reach the desired solution, but also start the system x_0 at the Hamming distance $d_H(x_0,x^\star)=j$. Trivially

$$\begin{aligned} &P(d_m\leq 0)\geq P(d_m\leq 0, d_0=j)=P(d_0=j) \\ &P(d_m\leq 0|d_0=j) \end{aligned} \tag{106}$$

where can identify $P(d_0=j)$ with $P(E_1)$ in the main text, i.e.,

$$P(d_0=j)=P(E_1)=\frac{1}{2^n}\binom{n}{\kappa n},\quad j=\kappa n. \tag{107}$$

Next, we wish to connect the remaining factor in (106), i.e., $P(d_m\leq 0|d_0=j)$, to the probability $P(E_2)$, which, recalling from the main text, corresponds to the event E_2, where precisely νm steps decrease the Hamming distance, while precisely $(1-\nu)m$ steps increase the Hamming distance. (For the walk on $\mathbb{Z}$, this extends to νm steps in the negative direction, and $(1-\nu)m$ steps in the positive direction.) We conclude that the total decrease is

$$d_0-d_m=(2\nu-1)m. \tag{108}$$

Let us also recall that the combination of E_1 and E_2 is successful, i.e., leads to $d_m\leq 0$, if

$$(2\nu-1)m\geq\kappa n. \tag{109}$$

It is useful to note that $(d_l)_l$ is not only Markovian, but also translation symmetric, which means that the change d_0-d_m is independent of the initial state d_0, i.e., the joint distribution of these factorize. (As a side remark, this independence also means that $P(E_1\cap E_2)=P(E_1)P(E_2)$.) Hence

$$\begin{aligned} &P(d_m\leq 0|d_0=\kappa n) \\ &=P(d_0-d_m\geq\kappa n|d_0=\kappa n) \\ &\text{[Since } d_0-d_m \text{ is independent of } d_0\text{]} \\ &=P(d_0-d_m\geq\kappa n). \end{aligned} \tag{110}$$

By comparison of (110) with (108), it follows that

$$\begin{aligned} &P(d_m\leq 0|d_0=\kappa n)=P(d_0-d_m\geq\kappa n) \\ &\geq P(E_2),\quad \text{if}\quad (2\nu-1)m\geq\kappa n. \end{aligned} \tag{111}$$

Alternatively, we can reach the same conclusion by comparing (7) with (102) to see that

$$\begin{aligned} P(E_2)=\binom{m}{\nu m}\left(\frac{1}{3}\right)^{\nu m}\left(\frac{2}{3}\right)^{(1-\nu)m} &\leq P(d_m\leq 0|d_0=\kappa n), \\ \text{if}\quad (2\nu-1)m\geq\kappa n. & \end{aligned} \tag{112}$$

By (104)–(107) and (111), we can conclude that

$$N_{\text{Schöning}}\leq\frac{1}{P(E_1)P(E_2)},\quad \text{if}\quad (2\nu-1)m\geq\kappa n. \tag{113}$$

2. GW PROCESS

For the GW process, let us recall that it consists of a classical outer loop that at each round assigns a definite (classical) initial state, whereas the walk process is Groverized. We assume that the number of iterations of the Grover procedure is tuned to the density of successful walks for a specific

initial Hamming distance $j = \kappa n$, i.e., to the success probability $P(x_m = x^\star | d_H(x_0|x^\star) = \kappa n)$. In the analysis, we lower bound the success probability by assuming that the process fails whenever $d_H(x_0, x^\star) \neq \kappa n$ (which may be pessimistic). The probability to obtain the initial state x_0 with the Hamming distance κn is $P(d_H(x_0, x^\star) = \kappa n)$, and thus, in average, we need to repeat the outer loop $1/P(d_H(x_0, x^\star) = \kappa n)$ times to be guaranteed to reach the initial Hamming distance κn at least once. In the successful case, the Grover procedure requires $1/\sqrt{P(x_m = x^\star | d_H(x_0|x^\star) = \kappa n)}$ iterations. Consequently, an upper bound on the total number of steps is

$$
\begin{aligned}
N_{\mathrm{GW}} &\leq \frac{1}{P\left(d_H(x_0, x^\star) = \kappa n\right)\sqrt{P\left(x_m = x^\star \middle| d_H(x_0|x^\star) = \kappa n\right)}} \\
&\quad \text{[By Proposition 9]} \\
&\leq \frac{1}{P(d_0 = \kappa n)\sqrt{P(d_m \leq 0 | d_0 = \kappa n)}} \\
&\quad \text{[By (107) and (111)]} \\
&\leq \frac{1}{P(E_1)\sqrt{P(E_2)}}, \quad \text{if} \quad (2\nu - 1)m \geq \kappa n. \qquad (114)
\end{aligned}
$$

ACKNOWLEDGMENT

The authors thank Phillip Keldenich for providing the SAT instance that was used for numerical simulations resulted in Fig. 8.

REFERENCES

[1] T. Schöning, "A probabilistic algorithm for k-SAT and constraint satisfaction problems," in *Proc. 40th Annu. Symp. Found. Comput. Sci.*, 1999, pp. 410–414, doi: 10.1109/SFFCS.1999.814612.

[2] U. Schöning and J. Torán, *The Satisfiability Problem: Algorithms and Analyses*. Berlin, Germany: Lehmanns, 2013. [Online]. Availaible: https://www.lehmanns.de/shop/mathematik-informatik/27340853-9783865415271-the-satisfiability-problem

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th ACM Annu. Symp. Theory Comput.*, 1996, pp. 212–219, doi: 10.48550/arXiv.quant-ph/9605043.

[4] A. Ambainis, "Quantum search algorithms," *ACM SIGACT News*, vol. 35, no. 2, pp. 22–35, 2004, doi: 10.1145/992287.992296.

[5] V. Dunjko, Y. Ge, and J. I. Cirac, "Computational speedups using small quantum devices," *Phys. Rev. Lett.*, vol. 121, 2018, Art. no. 250501, doi: 10.1103/PhysRevLett.121.250501.

[6] R. A. Moser, "Exact algorithms for constraint satisfaction problems," doctoral dissertation, Dept. Comput. Sci., ETH Zürich, Zürich, Switzerland, 2012, doi: 10.3929/ethz-a-009755667.

[7] P. Cheeseman, B. Kanefsky, and W. M. Taylor, "Where the really hard problems are," in *Proc. 12th Int. joint Conf. Artif. Intell.*, 1991, vol. 1, pp. 331–337, doi: 10.5555/1631171.1631221.

[8] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Hoboken, NJ, USA: Wiley-Interscience, 2006, doi: 10.1002/047174882X.

[9] V. Eshaghian, S. Wilkening, J. Åberg, and D. Gross, "Data for explicit run-times for hybrid SAT-solvers," 2024. [Online]. Available: https://github.com/SoerenWilkening/QuantumSchoening

[10] H. Callen, *Thermodynamics and an Introduction to Thermostatistics*. New York, NY, USA: Wiley, 1985. [Online]. Availaible: https://www.wiley.com/en-us/Thermodynamics+and+an+Introduction+to+Thermostatistics%2C+2nd+Edition-p-9780471862567

[11] D. P. Keldenich, personal communication, Mar. 2024.