Relaxing constraints in Simon's algorithm

Daniel J. Bernstein[1,2]

[1] Department of Computer Science, University of Illinois at Chicago, USA
[2] Institute of Information Science, Academia Sinica, Taiwan
djb@cr.yp.to

Abstract. Simon has recently claimed a quantum polynomial-time algorithm for the dihedral coset problem. This paper relaxes some constraints in Simon's algorithm, straightforwardly obtaining a more general algorithm that (1) appears to still satisfy the main intuition stated for the original claim while (2) including much faster cases than the original algorithm. This paper does not provide evidence for or against the claim in the original cases, and does not provide evidence for or against the claim in the extended cases.

It's often very difficult to tell the difference between an extremely slow algorithm and an algorithm that doesn't work at all.

The algorithm in Simon's recent paper [1] sounds like it uses more than n^{13} qubits. This makes it easy to conclude that obtaining experimental evidence for or against the claimed performance of the algorithm is infeasible on today's general-purpose computers even for $n = 2$.

The point of this paper is that the n^{13} is an artifact of parameter constraints in [1] that don't seem to be explained in [1]. Perhaps I'm missing something, but the intuition presented in [1] appears to be compatible with an algorithm using $\Theta(n^{1.5} \lg n)$ qubits.

Warning: I haven't double-checked any of this yet.

Basic notation. Vectors are 0-indexed. Matrices start with row 0 and column 0.

I'll use dot-product notation not just for vectors but also for matrices: $M \cdot N = \sum_{r,c} M_{r,c} N_{r,c}$.

The result of removing position i from a vector V is written $\operatorname{erase}_i V$. For example, $\operatorname{erase}_0(2,7,1,8) = (7,1,8)$ and $\operatorname{erase}_1(2,7,1,8) = (2,1,8)$.

Quantum states are arbitrary nonzero vectors, not required to be normalized.

The function $x \mapsto [x = p]$ is written with Dirac's ket notation $|p\rangle$ rather than with Kronecker's notation δ_p.

Permanent ID of this document: 5e4adbfcb2d6dd8ab14ed0e302be7990d404532f. Date: 20260813. This work was funded by the Academia Sinica Grand Challenge Project (AS-GCP-114-M01) and by an Amazon Research Award. "Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s)." No LLM usage.

Parameters. Fix positive integers N, Z, R, C, T with $Z \leq R$ and $N \in 2^T\mathbb{Z}$. Also fix an integer Δ.

(The objective is to find the bottom bit of a hidden shift mod N. The algorithm uses RC samples, arranged here as matrices with R rows and C columns. The algorithm hopes for the B matrix below to have at least Z all-zero rows. For each of those Z rows, the algorithm extracts T top bits of a corresponding subset sum; Δ is a new optional addition to the sum of those truncated subset sums, compensating for the bias introduced by truncation. [1] takes "$N = 2^n$"; describes the rows as "groups"; takes groups "of size $c \log n$", i.e., takes C as "$c \log n$", where "$\log n$" appears to mean an integer close to $\lg n = \log_2 n$; takes RC, the total number of samples, as "$Q = kn^{c+1}$" with "$k > c$"; takes Z as "$a = n/\log n$"; and takes the top T bits as "the most significant $\log n$ bits", i.e., takes T as "$\log n$". [1] is thus defining N, Z, R, C, T from just "n" and "c" and "k". [1] also does not have the Δ addition; i.e., it takes $\Delta = 0$.)

Top bits. For each $t \in \{1, 2, \ldots, T\}$, define $\operatorname{top}_t : \mathbb{Z}/N \to \mathbb{Z}/2^t$ and $\operatorname{chop}_t : \mathbb{Z}/N \to \mathbb{Z}/(N/2^t)$ as the maps induced by $x \mapsto \lfloor (2^t/N)x \rfloor$ from $\mathbb{Z}$ to $\mathbb{Z}/2^t$ and $x \mapsto x$ from $\mathbb{Z}$ to $\mathbb{Z}/(N/2^t)$ respectively. Note that x is determined by the pair $(\operatorname{top}_t x, \operatorname{chop}_t x)$.

Also define $\operatorname{top}'_1 : \mathbb{Z}/2^T \to \mathbb{Z}/2$ and $\operatorname{chop}'_1 : \mathbb{Z}/2^T \to \mathbb{Z}/2^{T-1}$ in the same way.

Row sums. For each $b \in \{0,1\}^{R \times C}$, each $y \in (\mathbb{Z}/N)^{R \times C}$, and each r with $0 \leq r < R$, define $\rho_{b,y,r} = b_r \cdot y_r = \sum_{0 \leq c < C} b_{r,c} y_{r,c} \in \mathbb{Z}/N$.

Also write $\rho_{b,y} = (\rho_{b,y,0}, \ldots, \rho_{b,y,R-1}) \in (\mathbb{Z}/N)^R$.

(In [1], the b matrix is written as a vector "ϕ"; the entries are written "b_i" with "i" ranging from 1 through "Q"; the row number is written "j"; the row sum $\rho_{b,y,r}$ is written "r_j"; applying top_T to "r_j" gives "s_j".)

Matrix sums. For each $b \in \{0,1\}^{R \times C}$ and each $y \in (\mathbb{Z}/N)^{R \times C}$, define $\sigma_{b,y} = \sum_{0 \leq r < R} \rho_{b,y,r} \in \mathbb{Z}/N$.

(This is "z" in [1]; applying top_1 and chop_1 to "z" gives "h" and "z'" respectively.)

Extracting all-zero rows. For each $B \in \{0,1\}^{R \times C}$, define perm_B as a specific easily computable permutation of $\{0, 1, \ldots, R-1\}$ with the following property: if B has at least Z all-zero rows then row $B_{\operatorname{perm}_B(r)}$ is all-zero for each $r' \in \{0, 1, \ldots, Z-1\}$. Here's an adequate specification: first create R bits saying which rows are zero, and then use a sorting network, let's say Knuth's Algorithm M, to sort those bits into non-decreasing order.

(In [1], the B matrix is written as a vector "D" with entries "b'_i".)

Sums of some truncated row sums. For each $B, b \in \{0,1\}^{R \times C}$ and each $y \in (\mathbb{Z}/N)^{R \times C}$, define $\tau_{B,b,y} \in \mathbb{Z}/N$ as $\Delta + \sum_{0 \leq r < Z} \operatorname{top}_T \rho_{\operatorname{perm}_B(r),b,y}$.

(This sum, for the case $\Delta = 0$, is "s^*" in [1].)

Algorithm input. The input is

$$\sum_{b \in \{0,1\}^{R \times C}} |b, a + bd\rangle$$

for some $a \in (\mathbb{Z}/N)^{R \times C}$ and some $d \in \mathbb{Z}/N$. Success of the algorithm is defined to mean that the algorithm output is $d \bmod 2$.

(An algorithm that flips a coin will succeed half the time, so what's interesting would be an algorithm that succeeds, say, 53% of the time. The algorithms under discussion have success chance independent of a, so it doesn't matter whether probability is defined for a particular a or for an average a, and it doesn't matter how the choices of a might be related across repeated runs. The context allows the algorithm to run repeatedly for the same d, so an algorithm that succeeds 53% of the time can be iterated to reliably find $d \bmod 2$, and then an appropriate further iteration finds all the bits of d.)

Algorithm steps. The algorithm begins by applying a quantum Fourier transform on the $a + bd$ entries using $\omega = \exp(2\pi i/N)$, obtaining

$$\sum_{b \in \{0,1\}^{R \times C}} \sum_{y \in (\mathbb{Z}/N)^{R \times C}} \omega^{(a+bd) \cdot y} |b, y\rangle .$$

(At this point the y qubits won't change any more so they can be measured—and reused for storing other data. This measurement can also be interleaved into the quantum Fourier transform. Similarly, various other quantities below can be measured before the end of the algorithm. But I'll keep everything in the notation for the algorithm state.)

The algorithm next computes (in superposition) $\sigma_{b,y}$ and

$$\operatorname{top}_T \rho_{b,y} = (\operatorname{top}_T \rho_{b,y,0}, \ldots, \operatorname{top}_T \rho_{b,y,R-1}),$$

obtaining

$$\sum_{b \in \{0,1\}^{R \times C}} \sum_{y \in (\mathbb{Z}/N)^{R \times C}} \omega^{(a+bd) \cdot y} |b, y, \sigma_{b,y}, \operatorname{top}_T \rho_{b,y}\rangle .$$

(This is steps 2 and 3 in [1].)

The algorithm next computes a Hadamard transform at each position in b, obtaining

$$\sum_{B \in \{0,1\}^{R \times C}} \sum_{b \in \{0,1\}^{R \times C}} \sum_{y \in (\mathbb{Z}/N)^{R \times C}} (-1)^{B \cdot b} \omega^{(a+bd) \cdot y} |B, y, \sigma_{b,y}, \operatorname{top}_T \rho_{b,y}\rangle .$$

Recall that the notation in this paper does not require states to be normalized.

The algorithm next computes perm_B (this is part of step 4 in [1], along with computing the above Hadamard transform) and $\tau_{B,b,y} = \Delta +$

$\sum_{0\le r<Z} \operatorname{top}_T \rho_{\operatorname{perm}_B(r),b,y}$, and uses $\tau_{B,b,y}$ to erase position $\operatorname{perm}_B(0)$ from $\operatorname{top}_T \rho_{b,y}$, obtaining

$$\sum_{B\in\{0,1\}^{R\times C}} \sum_{b\in\{0,1\}^{R\times C}} \sum_{y\in(\mathbb{Z}/N)^{R\times C}} (-1)^{B\cdot b}\omega^{(a+bd)\cdot y} \left|B, y, \sigma_{b,y}, \operatorname{erase}_{\operatorname{perm}_B(0)} \operatorname{top}_T \rho_{b,y}, \operatorname{perm}_B, \tau_{B,b,y}\right\rangle.$$

(In the context of early measurement, B is measured before this so perm_B doesn't have to be computed in superposition; there are just some additions in superposition of appropriate entries of $\operatorname{top}_T \rho_{b,y}$ to obtain $\tau_{B,b,y}$, overwriting entry $\operatorname{perm}_B(0)$ of $\operatorname{top}_T \rho_{b,y}$.)

At this point there's an optional filtering step that I'll suppress from formulas since I believe its effect is swamped by other aspects of this paper's relaxation, but I'll mention it here for completeness: the step is to compute $\ell \in \{0,1\}$ ([1] writes "l_{s^*}") as the product of the top T' bits of $\tau_{B,b,y}$ except for the topmost bit, where T' is another attack parameter. If this option is used then the algorithm result is skipped in the case $\ell = 1$.

The algorithm splits $\sigma_{b,y}$ into $\operatorname{top}_1 \sigma_{b,y}$ and $\operatorname{chop}_1 \sigma_{b,y}$, and splits $\tau_{B,b,y}$ into $\operatorname{top}'_1 \tau_{B,b,y}$ and $\operatorname{chop}'_1 \tau_{B,b,y}$. The algorithm overwrites $\operatorname{top}_1 \sigma_{b,y}$ with $\operatorname{top}_1 \sigma_{b,y} + \operatorname{top}'_1 \tau_{B,b,y} \in \mathbb{Z}/2$. (In [1], this is part of step 7, overwriting "h" with "$h \oplus h^*$".)

Finally, the algorithm applies Hadamard transforms to the qubits of $\operatorname{chop}_1 \sigma_{b,y}$ and to $\operatorname{top}'_1 \tau_{B,b,y}$, obtaining

$$\sum_{F\in\{0,1\}} \sum_{S\in\{0,1\}^{\lceil \log_2 N-1\rceil}} \sum_{B\in\{0,1\}^{R\times C}} \sum_{b\in\{0,1\}^{R\times C}} \sum_{y\in(\mathbb{Z}/N)^{R\times C}} (-1)^{F\cdot \operatorname{top}'_1 \tau}(-1)^{S\cdot \operatorname{chop}_1 \sigma}(-1)^{B\cdot b}\omega^{(a+bd)\cdot y} \left|B, y, \operatorname{top}_1 \sigma + \operatorname{top}'_1 \tau, S, \operatorname{erase}_{\operatorname{perm}_B(0)} \operatorname{top}_T \rho, \operatorname{perm}_B, F, \operatorname{chop}'_1 \tau\right\rangle.$$

Here $\rho_{b,y}$, $\sigma_{b,y}$, and $\tau_{B,b,y}$ have been abbreviated ρ, σ, and τ, and the dot product with $\operatorname{chop}_1 \sigma$ views the bits of $\operatorname{chop}_1 \sigma$ as a vector of length $\lceil \log_2 N - 1\rceil$. (In [1], $\operatorname{chop}_1 \sigma_{b,y}$ is "$\bar{s}^*$"; the Hadamard transform on that is part of step 6; the Hadamard transform on $\operatorname{top}'_1 \tau$, written "$h^*$", is after step 7.)

The main claim. My understanding is that the main claim from [1] is that the following statement holds: assume that $n \ge 2$, $N = 2^n$, $c \ge 12$, $C \ge c \lg n$, $k > c$, $RC \ge kn^{c+1}$, $n/\lg n \le Z \le n/\lg n + 2$, $\lg n \le T \le \lg n + 2$, $\Delta = 0$, and the optional filtering is used; then there's at least a $1/n^{1+o(1)}$ chance that the measured B has at least Z all-zero rows and the measured S is 0 and $\ell = 0$; also, if these conditions hold then the measured F is at least $1/n^{1+o(1)}$ correlated with $d \bmod 2$.

In this paper, I'm not providing evidence for or against this claim. I'm instead observing that what sound to me like the most important aspects of the intuition that [1] provides for this claim apply to more choices of (Z, R, C, T, Δ), including choices where the algorithm uses far fewer samples and runs much more quickly.

As a next step, it would be interesting to build quantitative predictions for the success probability of the algorithm as functions of (Z, R, C, T) for suitable Δ,

and to try to disprove those predictions by comparing them to simulated results for small sizes. This could help add evidence for or against the intuition, whereas the same strategy would be hopeless on current computers if the intuition were applicable only to much larger sizes.

As noted above, it's possible that I'm missing something that makes the intuition inapplicable to smaller choices of RC.

Intuition for the number of all-zero rows. A uniform random element of $\{0,1\}^C$ is all-zero with probability $1/2^C$. Presumably—let's ignore the possibility of correlations spoiling this—there will be about $R/2^C$ all-zero rows in B, so if Z is chosen somewhere around $R/2^C$ then there should be a good chance that the measured B will have at least Z all-zero rows. (These all-zero rows are important in the analysis of [1] because they drop out of $B \cdot b$.)

In other words, RC, the number of samples, should be around $ZC2^C$. With [1]'s choice of $Z \approx n/\lg n$ and $C \approx 12 \lg n$, the ZC factor is linear in n, but $2^C \approx n^{12}$, so there are about n^{13} samples. This motivates taking C much smaller—if this does not run into other problems.

Intuition for the entropy after top-bit measurement. Recall that $\rho_{b,y,r} = b_r \cdot y_r = \sum_{0 \le c < C} b_{r,c} y_{r,c} \in \mathbb{Z}/N$. After the y matrix is measured, measuring $\operatorname{top}_T \rho_{b,y,r}$ for any particular r reveals T bits of information about C bits $b_{r,0}, \ldots, b_{r,C-1}$. Presumably this leaves at least $C - T$ bits of entropy in that row.

One step in the analysis from [1] tallies this number of bits across the Z all-zero rows in B, obtaining $Z(C-T)$ bits of entropy, about $cn - n$ bits in the case $Z \approx n/\lg n$, $C \approx c \lg n$, $T \approx \lg n$; in particular, about $11n$ bits for $c = 12$, and then there's an analysis of what happens when roughly 2^{11n} quantities are spread pseudorandomly across 2^n states.

If C is taken as, say, $2T$ rather than $12T$, then $C - T$ drops to T rather than $11T$, but one can recover about the same entropy by multiplying 11 into Z. This also forces another factor 11 into R but drastically reduces $RC \approx ZC2^C$, the number of samples.

One can even take C as, e.g., $T + 3$. Taking Z linear in n should reach the previous amount of entropy. Then 2^C is $\Theta(n)$, and the number of samples is $\Theta(n^2 \lg n)$. I don't see how the intuition stated in [1] prohibits this case.

Intuition for the number of top bits. The reason in [1] for taking $T \approx \lg n$ is as follows.

The algorithm in [1] sets up conditions to guarantee that "the computed value of h^* will be the correct highest-order bit of z^*", meaning that $\sum_{0 \le r < Z} \operatorname{top}_T \rho_{\operatorname{perm}_B(r),b,y}$ has the same top bit as $\sum_{0 \le r < Z} \rho_{\operatorname{perm}_B(r),b,y}$.

Each $\rho_{\operatorname{perm}_B(r),b,y}$ can be viewed as its top T bits, scaled appropriately, plus an error between 0 and $N/2^T$, the output of chop_T. The sum of errors is at most $ZN/2^T$, which is about $(n/\lg n)N/n = N/\lg n$ with the choices $Z \approx n/\lg n$ and $T \approx \lg n$. The filtering step takes $T' \approx \lg \lg n$, discarding the cases where this sum is being added to something within $N/\lg n$ of the next multiple of $N/2$.

For more general choices of Z, this approach mandates taking $ZN/2^T$ noticeably smaller than $N/2$, i.e., taking T noticeably above $\lg(2Z)$. The increases of Z mentioned above, from $n/\lg n$ to a multiple of that or to $\Theta(n)$, then require a logarithmic increase in T.

But [1] also allows errors, both from the input and from the internals of the algorithm itself. So it's not necessary for the top bits to *always* match; a high correlation should be good enough.

Presumably the error in $\rho_{\mathrm{perm}_B(r),b,y}$ will be almost exactly $(N/2^T-1)/2$ on average, in effect increasing top_T by $(1-2^T/N)/2$. Taking $\Delta \approx Z(1-2^T/N)/2$ accounts for this. There will still be random variations, but presumably those are usually on the scale of $\sqrt{Z}$, so taking T slightly above $(1/2)\lg Z$ should suffice. (It might even make sense to take T *below* $(1/2)\lg Z$; the question here is the cost vs. the correlation obtained.)

For any particular gap $C-T$, reducing T by 1 means that C can be decreased by 1, reducing the number of samples in half. Taking, e.g., $C = \lfloor(\lg n)/2\rfloor + 6$ and $T = \lfloor(\lg n)/2\rfloor + 3$ and $Z \approx 4n$ would use only $\Theta(n^{1.5}\lg n)$ samples, and I don't see how any of the intuition stated in [1] wouldn't apply. For experiments aimed at collecting evidence for or against the intuition, it seems reasonable to try even smaller parameters.

References

[1] Daniel R. Simon, *A polynomial-time quantum algorithm for the dihedral coset problem* (2026), accessed 20260813. URL: https://eprint.iacr.org/2026/1591. Cited in §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1.