COMPUTING DISCRETE LOGARITHMS IN AN INTERVAL

STEVEN D. GALBRAITH, JOHN M. POLLARD, AND RAMINDER S. RUPRAI

Abstract. The discrete logarithm problem in an interval of size N in a group G is: Given $g, h \in G$ and an integer N to find an integer $0 \leq n \leq N$, if it exists, such that $h = g^n$. Previously the best low-storage algorithm to solve this problem was the van Oorschot and Wiener version of the Pollard kangaroo method. The heuristic average case running time of this method is $(2 + o(1))\sqrt{N}$ group operations.

We present two new low-storage algorithms for the discrete logarithm problem in an interval of size N. The first algorithm is based on the Pollard kangaroo method, but uses 4 kangaroos instead of the usual two. We explain why this algorithm has heuristic average case expected running time of $(1.715 + o(1))\sqrt{N}$ group operations. The second algorithm is based on the Gaudry-Schost algorithm and the ideas of our first algorithm. We explain why this algorithm has heuristic average case expected running time of $(1.661 + o(1))\sqrt{N}$ group operations. We give experimental results that show that the methods do work close to that predicted by the theoretical analysis.

This is a revised version since the published paper that contains a corrected proof of Theorem 6 (the statement of Theorem 6 is unchanged). We thank Ravi Montenegro for pointing out the errors.

Keywords: discrete logarithm problem (DLP), random walks.

MSC2000: 11Y16 , 68W20 , 11T71.

1. Introduction

The discrete logarithm problem (DLP) in an interval is the problem: Given g, h in a group G and $N \in \mathbb{Z}_{>0}$ such that $h = g^n$ for some $0 \leq n \leq N$ (where N is less than the order of g), to compute n. This is a fundamental computational problem that arises naturally in a number of contexts, for example the DLP with c-bit exponents (c-DLSE) [7, 13, 20], decrypting in the Boneh-Goh-Nissim homomorphic encryption scheme [1], counting points on curves or abelian varieties over finite fields [6], the analysis of the strong Diffie-Hellman problem [2, 9], and side-channel or small subgroup attacks [8, 10]. One can solve the DLP in an interval using the baby-step-giant-step algorithm; this requires $O(\sqrt{N})$ group operations and storage of $O(\sqrt{N})$ group elements. The fastest variant (given in Section 3 of Pollard [16]) has average case running time of $\frac{4}{3}\sqrt{N}$ group operations.

The Pollard kangaroo method [15] was designed to solve the DLP in an interval using constant number of group elements of storage. This is a randomised algorithm, which was originally given a heuristic analysis. Recently, Montenegro and Tetali [11] have given a more rigorous analysis of the kangaroo method.

The state-of-the-art for solving the DLP in an interval since the early 1990s was the distributed kangaroo algorithm as developed by van Oorschot and Wiener [19, 21]. The algorithm of van Oorschot and Wiener solves the DLP in an interval of size N with heuristic average case expected running time of $(2 + o(1))\sqrt{N}$ group operations and polynomial storage. This record has stood for over 15 years so it might have been natural to believe that $(2 + o(1))\sqrt{N}$ group operations was the best possible result for algorithms of this type. However, we provide two new algorithms, both of which beat this record. The first algorithm is a simple modification of the kangaroo method. The trick is to translate the discrete logarithm problem to the interval $[-N/2, N/2]$ and then start wild kangaroos at both h and h^{-1}. The second algorithm is a variant of an algorithm due to Gaudry and Schost [6].

We remark that the Pollard rho algorithm is a randomised algorithm to solve the DLP in a cyclic group. If the group has order N then the heuristic average case expected number of group operations performed by the algorithm (again, using distinguished points as done by van Oorschot and Wiener [19, 21]) is $\sqrt{\pi N/2} \approx 1.25\sqrt{N}$. Since the DLP in a group is a special case of the DLP in an interval, one does not expect any variant of the kangaroo method to have average case expected running time lower than $1.25\sqrt{N}$ group operations.

This work supported by EPSRC grant EP/D069904/1.

In groups for which inversion can be computed much more quickly than a general group operation (for example, elliptic curves) then an improvement was recently given by Galbraith and Ruprai, using an algorithm due to Gaudry and Schost. The main result of [4] is an algorithm for the DLP in an interval of size N with heuristic average case expected running time of approximately $(1.36 + o(1))\sqrt{N}$ group operations.

2. Background on the Pollard Kangaroo Method

We first briefly recall the Pollard kangaroo method using distinguished points as described by van Oorschot and Wiener [21] and Pollard [16]. To fix notation: We are given g, h and N (throughout the paper we assume N is even) and asked to find $0 \leq n \leq N$ such that $h = g^n$.

For $a, b \in \mathbb{R}$ we use the notation $[a, b]$ to denote $\{n \in \mathbb{Z} : a \leq n \leq b\}$. One pictures the kangaroo method in terms of the exponents in the group. In other words, we identify a group element g^n with the integer $n \in [0, N]$.

As with the rho method, the kangaroo method relies on a pseudorandom walk, however steps in the kangaroo walk correspond to known small increments in the exponent (in other words, kangaroos make small jumps). The tame kangaroo starts in the middle of the interval (i.e., at $g^{N/2}$) and jumps towards the right. The wild kangaroo starts at the group element h (with unknown discrete log) and jumps to the right using the same pseudorandom walk. For both walks, one stores the current group element z and also some information about its discrete logarithm: for tame walks one stores y such that $z = g^y$ and for wild walks one stores y such that $z = hg^y$. On a serial computer one alternately jumps the tame and wild kangaroos. Every now and then a tame or wild kangaroo lands on a distinguished group element and this is stored in a sorted list, binary tree or hash table together with a flag indicating whether the kangaroo was tame or wild and the corresponding value y as above. Once the same group element is visited twice, by kangaroos of different types, the DLP is solved (since we have an equation of the form $g^y = hg^{y'}$). Van Oorschot and Wiener [21] and Pollard [16] discuss how to parallelise this algorithm; the idea is to use herds of kangaroos with longer jumps.

The kangaroo method is not analysed using the birthday paradox but using the mean step size m of the pseudorandom walks. Once the rear kangaroo reaches the starting point of the front kangaroo it is jumping over a region where roughly one in m group elements have been visited by the front kangaroo. Heuristically, there is therefore a $1/m$ probability at each step that the back kangaroo lands on a footprint of the front kangaroo. Hence, the walks join after an expected m steps.

One obtains the heuristic average case expected running time of approximately $(2 + o(1))\sqrt{N}$ group operations as follows: The rear kangaroo is, on average, distance $N/4$ from the front kangaroo. The rear kangaroo therefore performs an expected $N/(4m)$ jumps to reach the starting point of the front kangaroo, followed by an expected m more steps until the walks collide (and then a small number more steps until a distinguished point is hit; see analysis in the next paragraph). Since there are two kangaroos in action the total running time is roughly $2(N/(4m) + m)$ group operations. This expression is minimised by taking $m = \sqrt{N}/2$, which leads to a count of $2\sqrt{N}$ group operations.

Let θ be the probability that a group element is distinguished. Since the algorithm does not detect the collision until a distinguished point has been visited by the second walk, it follows that the overall heuristic average case expected running time is $2\sqrt{N} + 1/\theta$ group operations. Taking $\theta = c\log(N)/\sqrt{N}$ for some constant $c > 0$ means the expected number of group elements storage is $c\log(N)$ (i.e., polynomial storage) and the average case expected running time of the algorithm is $(2 + 1/(c\log(N))\sqrt{N} = (2 + o(1))\sqrt{N}$ group operations.

3. Using More Than Two Kangaroos

We start with three kangaroos. We now assume that g has odd order r (in practice r is usually prime; in general one reduces to prime order groups using the Pohlig-Hellman reduction [14]) and that the width N of the interval is such that $N < r$. It is convenient to assume that $10 \mid N$.

We shift the problem (multiplying the instance by $g^{-N/2}$ which requires an inversion in the group) so that it is of the form

$$h = g^n , \quad -N/2 \leq n \leq N/2.$$

The idea of the algorithm is to start two wild kangaroos, W_1 and W_2, from h and h^{-1} respectively (hence, one more inversion in the group is needed). Start a tame kangaroo T from the point $3N/10$. This is the middle of the interval $I = [N/10, N/2]$. Either one wild kangaroo is in the interval I, or both are in the interval $J = [-N/10, N/10]$. In either case there is a pair whose distance apart is not more than $N/5$. Effectively, this pair are performing the original method in an interval of reduced width $2N/5$.

As with the kangaroo method, we store the current group element together with some information about the DLP: for the tame kangaroo we store (z, y) such that $z = g^y$; for W_1 we store (z, y) such that $z = hg^y$; for W_2 we store (z, y) such that $z = h^{-1}g^y$. Whenever z is distinguished we store z, y and a flag indicating the type of kangaroo (T, W_1 or W_2). The crucial idea is that a collision between any pair will solve the DLP: if $g^y = hg^{y'}$ or $g^y = h^{-1}g^{y'}$ we can compute $\log_g(h)$; also if $hg^y = h^{-1}g^{y'}$ then, since g has odd order, we can solve $\log_g(h) = (y' - y)2^{-1} \pmod r$.

This method only requires two inversions in the group and so is applicable even in groups where inversions are much slower than a multiplication (such as $\mathbb{F}_p^*$). The method can be parallelised, giving a linear speedup, using the methods of van Oorschot and Wiener [21] or Pollard [16].

3.1. **Analysis of the Running Time.** We now give a heuristic analysis of the running time of this algorithm. As with the usual kangaroo method, we first determine the expected number of steps for a rear kangaroo to pass the starting point of a kangaroo in front of it. Then we determine the expected number of steps until the rear kangaroo visits a group element already visited by the front kangaroo.

Let the DLP be $h = g^{xN/2}$ where $0 \leq x \leq 1$ (without loss of generality we can assume it is in the right hand side of the interval). Let $0 \leq a(x) \leq b(x) \leq 1$ be such that the closest pair of kangaroos start a distance $a(x)N/2$ apart, and the second closest pair of kangaroos start a distance $b(x)N/2$ apart.

One has the following values for $a(x)$ and $b(x)$.

x	$a(x)$	$b(x)$
$0 \leq x \leq \frac{1}{5}$	$2x$	$\frac{3}{5} - x$
$\frac{1}{5} \leq x \leq \frac{3}{5}$	$\frac{3}{5} - x$	$2x$
$\frac{3}{5} \leq x \leq 1$	$x - \frac{3}{5}$	$x + \frac{3}{5}$

Let $m = \alpha\sqrt{N}$ be the mean step size. The naive and pessimistic assumption is that the closest pair of kangaroos will meet. The theory of the preceding section (applied to an interval of length $2N/5$) shows that we should take $\alpha = \sqrt{1/10}$. The existing results on the kangaroo method imply the heuristic average case expected running time would be at most $(3/2)(2 + o(1))\sqrt{2N/5} = (1.898 + o(1))\sqrt{N}$ group operations. The following, more precise, analysis allows for collisions involving the third kangaroo.

Theorem 1. *Suppose the three-kangaroo algorithm above is performed, to solve the DLP in an interval of size N, with a truly random walk comprising steps of mean size $0.375\sqrt{N}$. Let $\theta = \log(N)/\sqrt{N}$ be the probability of a group element being distinguished, and assume that distinguished points are uniformly distributed in the group. Then the average case expected number of group operations is $(1.819 + o(1))\sqrt{N}$.*

Proof. We ignore the cost of the inversions in the group, and the cost of the group operations to compute the initial value $T = g^{3N/10}$.

We first consider a fixed instance $h = g^{xN/2}$ of the DLP. The expected number of steps for the rear kangaroo to pass the starting point of the front kangaroo for the closest pair of kangaroos is $A(x) = a(x)N/(2m)$ where $m = \alpha\sqrt{N}$ is the mean step size and the optimality of the choice $\alpha = 0.375$ will be justified later. Similarly, for the second closest pair it is $B(x) = b(x)N/(2m)$. Finally, the expected time for the rear kangaroo to pass the start point of the front kangaroo for the most distant pair of kangaroos is $C(x) = A(x) + B(x)$. The actual numbers of steps are normally distributed random variables whose standard deviations are of lower order than their means (the standard deviations are $O(\sqrt{m}\log(m))$, compared to means $O(m)$). For the remainder of the proof we replace these random variables by their mean value. It can be shown that, when $m = \alpha\sqrt{N}$, the resulting error terms are all $o(1)\sqrt{N}$, as stated in the Theorem.

Suppose a kangaroo has walked across an interval of length l. Then we expect l/m group elements to have been visited, and can think of all elements as being roughly of distance m apart. If another kangaroo walks along the same interval then, at each step, the probability that it lands on a group element visited by the first kangaroo is (under our assumption that the random walk is uniformly chosen from the set of all

possible random walks of the correct type) $1/m$. Hence, the probability to make k steps and avoid the steps of the first kangaroo is $(1-1/m)^k \leq e^{-k/m}$. It follows that the probability of a collision after $k+1$ steps is at most $\frac{1}{m}e^{-k/m}$.

Suppose the algorithm is organised so that, at each time step, all three kangaroos make one jump. The algorithm stops when there is a collision: namely two kangaroos of different type visit the same group element (here we ignore the fact that collisions are only detected once a distinguished point is visited). Let $X(x)$ be the random variable on $\mathbb{N}$ such that $\Pr(X(x)=k)$ is the probability (for a fixed instance of the DLP) that the first collision takes place at time step k. For ease of notation we now drop the explicit mention of x in $A(x), B(x), C(x)$ and $X(x)$. Then, for $k \geq 0$, we have $\Pr(X=(k+1)) \leq f(k+1)$ where

$$f(k) = \begin{cases} 0 & 1 \leq k \leq A \\ \frac{1}{m}e^{-(k-1-A)/m} & A+1 \leq k \leq B \\ \frac{2}{m}e^{-(k-1-A)/m}e^{-(k-1-B)/m} & B+1 \leq k \leq C \\ \frac{3}{m}e^{-(k-1-A)/m}e^{-(k-1-B)/m}e^{-(k-1-C)/m} & C+1 \leq k. \end{cases}$$

The expected value of X is

$$\sum_{k=1}^{\infty} k \Pr(X=k) \leq \int_1^{\infty} k \Pr\big(X=\lfloor k \rfloor\big)dk \leq 1 + \int_1^{\infty} kf(k)dk.$$

Using

$$\int_u^v ze^{-a(z-b)}dz = -\left[\left(\tfrac{z}{a}+\tfrac{1}{a^2}\right)e^{-a(z-b)}\right]_u^v$$

one computes

$$\begin{aligned} \int_1^{\infty} kf(k)dk &= \tfrac{1}{m}\int_{A+1}^{B} ze^{-(z-1-A)/m}dz \ + \ \tfrac{2}{m}\int_{B+1}^{C} ze^{-(2z-2-C)/m}dz \ + \ \tfrac{3}{m}\int_{C+1}^{\infty} ze^{-(3z-3-2C)/m}dz \\ &\approx A + m - \tfrac{m}{2}e^{-(B-A)/m} - \tfrac{m}{6}e^{-C/m}. \end{aligned}$$

The approximation error is a multiplicative factor of $1+o(1)$ when $m = \alpha\sqrt{N}$.

The expected number of group operations performed by the algorithm on a specific instance (parameterised by x) up to the first collision is therefore

$$(3+o(1))\left(A(x) + m - \tfrac{m}{2}e^{-(B(x)-A(x))/m} - \tfrac{m}{6}e^{-C(x)/m}\right).$$

One also has to perform an expected $1/\theta$ time steps to hit a distinguished point (and so detect the collision) which adds $3/\theta$ to the running time. The expected running time of the algorithm is therefore $(c+o(1))\sqrt{N} + 3/\theta$ where

$$c = 3\left(a(x)/(2\alpha) + \alpha - \tfrac{\alpha}{2}e^{(a(x)-b(x))/(2\alpha^2)} - \tfrac{\alpha}{6}e^{-(a(x)+b(x))/(2\alpha^2)}\right).$$

Taking $\theta = \log(N)/\sqrt{N}$ gives $(c+o(1))\sqrt{N}$ group operations.

Now we determine the average case running time (averaging over all $0 \leq x \leq 1$). Inserting the values for $a(x)$ and $b(x)$ and integrating over $0 \leq x \leq 1$ gives

$$c = 3\left(\tfrac{1}{10\alpha} + \alpha - \tfrac{2\alpha^3}{3} + \left(\tfrac{5\alpha^3}{6} - \tfrac{\alpha}{5}\right)e^{-3/(5\alpha^2)} - \tfrac{\alpha^3}{6}e^{-1/\alpha^2}\right).$$

One computes the optimal value of α to be $0.3752120113\ldots$ giving $c = 1.8182026\ldots$, which we round up to 1.819. The naive choice $\alpha = \sqrt{1/10}$ gives $c \approx 1.834$. □

Our conjecture is that when the random walk is replaced by a pseudorandom walk of the standard type, assuming sufficiently many partitions and that distinguished points are sufficiently common and evenly distributed, then one can solve the DLP in an interval in $(1.819+o(1))\sqrt{N}$ group operations. This is a noticeable improvement over the heuristic $(2+o(1))\sqrt{N}$ group operations of the van Oorschot and Wiener method.

3.2. **Four Kangaroo Method.** We can make a further improvement. The starting points of W_1 and W_2, namely x and $-x$, are integers of the same parity. Hence, to obtain wild-wild collisions more quickly one should take walks whose jumps are all *even* length. However, now there is the possibility that the wild walks will never collide with the tame walk. The solution is to have two tame kangaroos, T_1 and T_2, starting on adjacent integers (one even and one odd). There is no possibility of a useless tame-tame collision, but there is a chance of tame-wild collisions, as desired (though with only one of the tame kangaroos).

The overall effect is to effectively halve the width of the interval but increase from 3 to 4 kangaroos. The algorithm itself is essentially identical to the three-kangaroo case. We run 4 kangaroos and store 4 lists of numbers. The jumps are even and the mean step size m' is $m' = \sqrt{2} \cdot \alpha\sqrt{N} \approx 0.53\sqrt{N}$ (i.e., the value m used before is multiplied by $\sqrt{2}$).

We now estimate the complexity of the method. As before, we organise the algorithm so that at each time step, each of the 4 kangaroos takes one jump. The number of steps for a kangaroo to travel distance d is $\frac{d}{m'} = \frac{1}{\sqrt{2}}\frac{d}{m}$ and, once a rear kangaroo passes the start of a front kangaroo, the probability of standing on a footstep of the front kangaroo is $2/m'$ (since the steps are even). Hence, the expected number of steps for a collision is $\frac{m'}{2} = \frac{\sqrt{2}m}{2} = \frac{1}{\sqrt{2}}m$. By ignoring the tame kangaroo for which collisions cannot occur, the algorithm is equivalent to applying the 3 kangaroo algorithm in an interval of size $N/2$. Theorem 1 therefore implies that the total number of steps to be performed by the idealised version of the algorithm is $(1.819 + o(1))\sqrt{N/2}$. In other words, the algorithm takes roughly $1/\sqrt{2}$ of the number of steps of the three-kangaroo algorithm. But we now do 4 jumps in each time step instead of 3, so we must multiply the running time by 4/3. In other words, the expected number of group operations of the four-kangaroo algorithm is the number from the three-kangaroo case multiplied by $4/(3\sqrt{2}) = \sqrt{8/9} \approx 0.9428$.

Hence, using 4 kangaroos gives an algorithm for the DLP in an interval whose heuristic average case expected running time is $(1.715 + o(1))\sqrt{N}$ group operations. This is a significant further improvement over the heuristic $(2 + o(1))\sqrt{N}$ group operations of the van Oorschot and Wiener method.

3.3. **Why are 4 Kangaroos Optimal?** One might also consider using one tame kangaroo and four wild kangaroos starting at h, h^{-1}, h^2 and h^{-2} respectively and a tame kangaroo g^t starting near the right hand side of the interval. Any collision between walks allows us to solve the discrete logarithm problem. This idea is tempting since the "most likely" collision can come from h being close to h^2 (this is the case when $h = g^x$ and x is small), or from h^2 being close to g^t (i.e., when $x \approx t/2$) or from h being close to g^t (i.e., when $x \approx t$).

However, the fact that one is now running 5 walks and that only 2 or 3 are likely to lead to a collision in any situation seems to offset the improved collision probability. We have given a rough analysis. We note that the closest kangaroos have distance x apart when $0 \leq x \leq t/4$, distance $|t - 2x|$ when $t/4 \leq x \leq 2t/3$ and distance $|t - x|$ when $2t/3 \leq x \leq 1$. A rough calculation suggests the overall running time for optimal choices of t and α is still at least $2\sqrt{N}$.

4. Gaudry-Schost Variant

Gaudry and Schost [6] developed a different approach to algorithms for solving the DLP. Their method involves pseudorandom walks of different types (typically, "tame" walks and "wild" walks) in subsets of the group. One applies a version of the birthday paradox in the regions of overlap of the subsets. A collision between walks of two different types leads to a solution to the DLP. Galbraith and Ruprai [3, 4] have shown that the Gaudry-Schost method can have some advantages over the Pollard kangaroo method. In particular, it can be used to efficiently solve the DLP in an interval when using equivalence classes under inversion.

We now explore versions of the Gaudry-Schost algorithm motivated by the 3 and 4 kangaroo algorithms in the previous section.

4.1. **Basic Three-Set Gaudry-Schost Algorithm.** This section presents a version of the Gaudry-Schost algorithm for the DLP in an interval. The algorithm in this section is what you get when you apply the standard Gaudry-Schost philosophy to the three-kangaroo method of Section 3. We will give an improved algorithm in Section 4.3.

Let N be the width of the interval. It is convenient to assume that $10 \mid N$. As before, let $h = g^n$ where $-N/2 \leq n \leq N/2$.

Recall that the three-kangaroo method starts random walks at n, $-n$ and $3N/10$ (the latter being the middle of the interval $[N/10, N/2]$). For $a, b \in \mathbb{R}$ we use the notation $[a, b]$ to denote $\{n \in \mathbb{Z} : a \leq n \leq b\}$. The Gaudry-Schost philosophy is therefore to run pseudorandom walks in 3 sets. The "tame" set is centered on $3N/10$ and has length $4N/10$ and the two "wild" sets also have length $4N/10$ and are centered on n and $-n$ respectively. We give formal definitions now.

$$\begin{array}{ll} \text{Tame set} & T = \frac{3}{10}N + [-2N/10, 2N/10] = [N/10, N/2], \\ \text{Wild set 1} & W_1 = n + [-2N/10, 2N/10], \\ \text{Wild set 2} & W_2 = -n + [-2N/10, 2N/10]. \end{array}$$

Throughout the paper we approximate the number of integers in an interval $[a, b]$ by its length $b - a$. Hence we write $|T| = |W_1| = |W_2| \approx 4N/10 = 2N/5$ rather than $|T| = 2N/5 + 1$

The algorithm starts three walks, each at a random point in the sets T, W_1 and W_2 respectively. At each time step the walks advance to a new point by applying a deterministic function, just as in the kangaroo method. If two walks collide then they follow the same path. Collisions are detected using distinguished points. Unlike the kangaroo method, there will occasionally be useless collisions between walks of the same type. A collision between walks of any two different types will lead to a solution of the DLP.

Unlike the kangaroo method, pseudorandom walks in the Gaudry-Schost method do not travel long distances. Instead, we typically use walks with relatively "local" behaviour (though not too local, otherwise they are "not random enough"). More precisely, let θ be the probability that a group element is distinguished (e.g., $\theta = \log(N)/\sqrt{N}$). Then walks have length $1/\theta$ on average, and we assume that after this many steps a walk has travelled significantly less than the width of the set in which it starts (our experiments suggest that each walk should cover around 0.1 to 1 percent of the size of the tame set, assuming that the number of individual walks performed is at least 1000). Unlike the kangaroo method, instead of continuing the walk when we hit a distinguished point, we start a new walk at a fresh, uniformly chosen, element of the interval (computing this new group element requires some group operations; the cost can be somewhat mitigated by precomputation and other tricks). Care must be taken when starting walks near the boundaries of the set so that the probability of a walk going outside the set is low. For more discussion of these issues we refer to [3, 6, 17].

Running random walks in this way, the DLP is solved when there is a collision between walks of different types. Hence, the expected running time of the algorithm depends on the sizes of the intersections $T \cap W_1$, $T \cap W_2$ and $W_1 \cap W_2$. We first determine the sizes of these intersections.

Lemma 2. *Let notation be as above. Write $h = g^{xN/2}$ where, without loss of generality $0 \leq x \leq 1$. Then the sizes of $T \cap W_1$, $T \cap W_2$ and $W_1 \cap W_2$ are given, up to an error $O(1/N)$, in the following table.*

x	$\lvert T \cap W_2\rvert/(N/2)$	$\lvert W_1 \cap W_2\rvert/(N/2)$	$\lvert T \cap W_1\rvert/(N/2)$
$0 \leq x \leq 1/5$	$1/5 - x$	$4/5 - 2x$	$1/5 + x$
$1/5 \leq x \leq 2/5$	0	$4/5 - 2x$	$1/5 + x$
$2/5 \leq x \leq 3/5$	0	0	$1/5 + x$
$3/5 \leq x \leq 1$	0	0	$7/5 - x$

Proof. This is easily verified. For example, to compute $|W_1 \cap W_2|$, the upper limit of W_2 is $2N/10 - xN/2 = (2/5 - x)(N/2)$ and the lower limit of W_1 is $-2N/10 + xN/2 = (-2/5 + x)(N/2)$. Hence, as long as the sets overlap, the length of the overlap is $(2/5 - x - (-2/5 + x))(N/2) = (4/5 - 2x)(N/2)$. The other cases are similar. □

For the analysis of the algorithm we need a generalisation of the birthday paradox that applies when seeking collisions between objects of different "types". A basic example of this is the following: Suppose we sample uniformly at random from a set of size M and write the element in one of two lists (either write elements alternately on the lists, or flip an unbiased coin to determine the list for each draw). It is known (e.g., see [18]) that the expected number of trials until the two lists have an element in common is $\sqrt{\pi M}$.

Theorem 3. *Let $n \in \mathbb{N}$ (we only require $n = 2$ or $n = 3$). Let $N, m \in \mathbb{N}$ (these will be assumed to be large and with $m < N$). Let $S_1, \ldots, S_n$ be subsets of $\{1, \ldots, N\}$ of size m and $L_1, \ldots, L_n$ lists that are initially empty.*

Suppose elements are repeatedly and independently sampled by the following method: choose $i \in \{1, \ldots, n\}$ uniformly and then choose $x \in S_i$ uniformly and store x in list L_i.

Let $M = \sum_{1 \leq i < j \leq n} |S_i \cap S_j|$ and suppose $M/N > c$ for some constant independent of N. Then the expected number of draws until the same element is sampled twice but put into different lists is

$$\tfrac{nm}{2}\sqrt{\pi/M} + O(N^{1/4}).$$

Proof. This is a special case of Theorem 1 of Galbraith and Holmes [5]. It can also be deduced (with a less precise error term) from the result of Selivanov [18]. □

Theorem 4. *Let notation be as above. Suppose the three-set Gaudry-Schost algorithm described above is run with a truly random walk and assume that distinguished points are uniformly distributed in the group. Then the average case expected running time is $(1.812 + o(1))\sqrt{N}$ group operations. (Note that the expectation is over all choices for the random walk.)*

Proof. Write $h = g^{xN/2}$ where $0 \leq x \leq 1$. We will compute the expected running time of the algorithm (over all choices for the random walk) for the specific value x. Then we will average over all possible values for $0 \leq x \leq 1$.

The total number of group operations is equal to the sum of: the set up cost; the number of steps taken in pseudorandom walks to the first useful collision; the number of "wasted" steps due to collisions between walks of the same type; the number of group operations used to restart walks; the number of steps after the first useful collision until a distinguished point is hit. Taking $\theta = \log(N)/\sqrt{N}$ means that $c\log(N)$ walks in total will be performed (so the number of group operations in restarting walks is bounded by $\log(N)^2$) and the number of steps after collision to next distinguished point is $o(1)\sqrt{N}$. Similarly, the set up cost is polynomial time if the number of partitions is polynomial. Adapting the analysis below (determining the number of steps before the first good collision) shows that the number of wasted steps in walks from collisions of the same type is a negligible proportion of the total number of steps (essentially, one only expects a small constant number of such collisions). Hence, the total number of group operations is $(c + o(1))\sqrt{N}$ where $c\sqrt{N}$ is the number of steps to the first collision. Hence, the rest of the proof is to compute this value.

We will apply Theorem 3 where N is the width of the interval and the sets S_1, S_2 and S_3 are T, W_1 and W_2. We have $m = |S_1| = 4N/10$ and will write $M = F(x)N/2$. By Lemma 2 we have

$$F(x) = \begin{cases} 6/5 - 2x & 0 \leq x \leq 1/5 \\ 1 - x & 1/5 \leq x \leq 2/5 \\ 1/5 + x & 2/5 \leq x \leq 3/5 \\ 7/5 - x & 3/5 \leq x \leq 1. \end{cases}$$

Theorem 3 implies that the expected number of steps is

$$\tfrac{3}{5}\sqrt{2\pi N}\; F(x)^{-1/2} \;+\; O(N^{1/4}).$$

To get an average time complexity we need to compute

$$\int_0^1 F(x)^{-1/2} dx = 5\sqrt{4/5} + \sqrt{6/5} - 4\sqrt{3/5} - 2\sqrt{2/5} \approx 1.204283.$$

The result follows from $\frac{3}{5}\sqrt{2\pi} \cdot 1.204283 \approx 1.811214$, which we round up to 1.812. □

Under the heuristic assumptions that the pseudorandom walk is sufficiently close to random (i.e., there are enough partitions, and those partitions are determined by a good enough hash function) and that $1/\theta$ is sufficiently large (so that we can safely ignore the group operations from re-starting walks) then we conclude that there is an algorithm for the DLP in an interval that performs $(1.812 + o(1))\sqrt{N}$ group operations.

4.2. **Basic Four-Set Gaudry-Schost Algorithm.** In the same way as the four-kangaroo algorithm is derived from the three-kangaroo algorithm, we can improve the three-set Gaudry-Schost algorithm by using steps of even length and having two tame sets – one consisting of elements with even DLP and one consisting of elements with odd DLP. There is no need to modify the random walks any further, since there is no "mean step size" in the Gaudry-Schost method.

To analyse the algorithm it is not necessary to prove a new version Theorem 4. The point is that one of the tame sets is completely wasted (collisions between tame and wild walks will only occur with respect to the tame set involving elements whose discrete logarithm has the same parity as n). Hence, we can apply

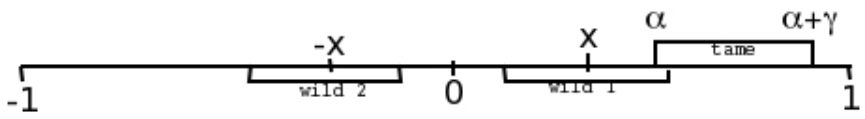

FIGURE 1. Tame and Wild sets for the improved Gaudry-Schost algorithm.

the same argument as that in Section 3.2. The algorithm is basically the same as working in an interval of size $N/2$, but we have to do $4/3$ of the work. Hence, one expects the four-set algorithm to run in $\sqrt{8/9}$ the time of the three-set algorithm.

In other words, one makes the conjecture that the four-set algorithm has average case expected running time of

$$(1.708 + o(1))\sqrt{N}$$

group operations.

This already gives a small improvement in theory over what was achieved using kangaroos (namely, $(1.715 + o(1))\sqrt{N}$). One caveat is that there are a number of reasons why the Gaudry-Schost algorithm does not necessarily work quite as well as the theory predicts. The main problem is that it is difficult to design a pseudorandom walk that behaves close to a truly random walk in the precise regions as stated in the description of the algorithm. For example, it is difficult to design pseudorandom walks that cover the sets uniformly without also sometimes stepping outside the regions. Extensive discussion of this issue is given in [3, 17]. A second problem is that extra group operations are needed when re-starting the random walks at new group elements.

For these reasons, it is unclear whether, in practice, this algorithm would be faster than the four kangaroo method. In the next section we obtain a further improvement which could mean that the Gaudry-Schost approach really is faster in practice.

4.3. **Improved Three-Set Gaudry-Schost Algorithm.** One powerful feature of the Gaudry-Schost philosophy is that there are many variations one can make to optimise the running time. In particular, the tame and wild sets can be different shapes and sizes and one is not even obliged to sample uniformly within them.

We now explain one variant of this algorithm and derive an improved running time. We write $h = g^{xN/2}$ with $-1 \leq x \leq 1$ and, without loss of generality, $x \geq 0$. We take the widths of the tame and wild intervals to be $\gamma N/2$ where γ is a variable whose value will be chosen later. We also start the tame set at a variable point, namely the tame set is $T = [\alpha N/2, (\alpha + \gamma)N/2]$ where $\alpha \geq \gamma/2$ is a value to be determined. The wild sets are $W_1 = [(x - \gamma/2)N/2, (x + \gamma/2)N/2]$ and $W_2 = [(-x - \gamma/2)N/2, (-x + \gamma/2)N/2]$. The (normalised) sets are drawn in Figure 1.

Lemma 5. *Let notation be as above. Write $h = g^{xN/2}$ where, without loss of generality $0 \leq x \leq 1$. Suppose $\gamma/2 \leq \alpha \leq \gamma$ and $\alpha + 3\gamma/2 \geq 1$. Then the sizes of $T \cap W_1$, $T \cap W_2$ and $W_1 \cap W_2$ are given in the following table.*

x	$\lvert T \cap W_2\rvert/(N/2)$	$\lvert W_1 \cap W_2\rvert/(N/2)$	$\lvert T \cap W_1\rvert/(N/2)$
$0 \leq x \leq \alpha - \gamma/2$	0	$\gamma - 2x$	0
$\alpha - \gamma/2 \leq x \leq \gamma/2$	0	$\gamma - 2x$	$x - (\alpha - \gamma/2)$
$\gamma/2 \leq x \leq \alpha + \gamma/2$	0	0	$x - (\alpha - \gamma/2)$
$\alpha + \gamma/2 \leq x \leq 1$	0	0	$(\alpha + 3\gamma/2) - x$

Proof. This is straightforward. □

Theorem 6. *The improved three-set Gaudry-Schost algorithm in this section, with truly random walks and assuming that distinguished points are uniformly distributed in the group, has average case expected running time of $(1.761 + o(1))\sqrt{N}$ group operations.*

Proof. As argued in the proof of Theorem 4 the total number of group operations is $(c + o(1))\sqrt{N}$ where $c\sqrt{N}$ is the number of elements to be sampled until the first collision between walks of different types. Since

$T \cap W_1 \cap W_2 = \emptyset$, in any region there can only be a collision between two types of walk. Write $M = F(x)N/2$ for the size of the overlap set $(T \cap W_1) \cup (T \cap W_2) \cup (W_1 \cap W_2)$. By earlier arguments (Theorem 3 with $n = 3$ and $m = \gamma N/2$, and the proof of Theorem 4), the average case expected running time is

$$\tfrac{3\gamma}{2}\sqrt{\pi N/2} \int_0^1 F(x)^{-1/2} dx.$$

By Lemma 5 we have

$$F(x) = \begin{cases} \gamma - 2x & 0 \leq x \leq (\alpha - \gamma/2) \\ 3\gamma/2 - x - \alpha & (\alpha - \gamma/2) \leq x \leq \gamma/2 \\ x - \alpha + \gamma/2 & \gamma/2 \leq x \leq \alpha + \gamma/2 \\ \alpha + 3\gamma/2 - x & \alpha + \gamma/2 \leq x \leq 1. \end{cases}$$

One easily computes $\int_0^1 F(x)^{-1/2} dx$ as a function of γ and α. A crude optimisation to minimise the integral gives the values $\gamma = 0.588$ and $\alpha = \gamma/2$. The integral is then approximately 1.593 and the average case expected complexity is $(1.761 + o(1))\sqrt{N}$ group operations. □

We therefore conjecture that by taking $1/\theta$ is sufficiently large and by using good enough random walks, there is an algorithm for the DLP in an interval that performs $(1.761 + o(1))\sqrt{N}$ group operations.

4.4. **Improved Four-Set Gaudry-Schost Algorithm.** One can of course apply the four-set trick. This multiplies the complexity by $\sqrt{8/9}$, giving an algorithm with conjectured complexity of $(1.661 + o(1))\sqrt{N}$ group operations. This slightly beats the value obtained from the kangaroo method.

4.5. **Further Improvement.** We have considered a more general setting where the tame interval and the wild intervals do not have the same length. This does not seem to lead to an algorithm with better complexity. We have also considered sampling from sub-regions of the tame and wild sets as was done in Sections 4.1.5 and 4.3.3 of [17]. Again, these ideas do not seem to give any further improvement to our main result.

5. Experimental Results

As remarked, all the results in this paper are heuristic, and there are several good reasons why the methods may not work quite as well in practice as the theory predicts. It is therefore essential to give some experimental results to support the theoretical calculations.

For simplicity we implemented the algorithm in the additive group $\mathbb{Z}$. There is no reason to think results would be any different for elliptic curve groups, assuming that the interval length is significantly smaller than the order of the group (if the interval is the same size as the order of the group then it is better to use the Pollard rho method).

5.1. **Four-Kangaroo Method.** Implementing the four-kangaroo method is quite straightforward. We took an interval of size $N = 2^{40}$ and chose jumps of even length so that they have the correct mean step size. Our implementation partitioned the set into 32 pieces using a simple non-linear function on $\mathbb{Z}$ reduced modulo 32. We took $\theta = 1/500$ (which is actually much bigger than $\log(N)/\sqrt{N}$ in this case; in other words, walks are rather short and the storage is rather large) so that each of the 4 lists contains around 900 group elements.

We performed over 1000 experiments and recorded the average running time of the algorithm. Each experiment is for a fresh value of the discrete logarithm (uniformly chosen in $[0, N/2]$) and a fresh choice of jumps in the random walk (uniformly chosen in the interval $[1, 2m]$). Recall that the theoretical prediction is $1.715\sqrt{N}$ group operations.

Indeed, we computed a confidence interval so that, with 95% confidence, the true value of the expected average case running time divided by $\sqrt{N}$ lies in 1.733 ± 0.044 (in other words, the interval $[1.689, 1.777]$). The theoretical value 1.715 does lie in this interval, so our experiments do not dispute the theoretical running time. These results are also given in Table 1.

For completeness, we also gathered some data about the three-kangaroo method under the same conditions. We computed a confidence interval so that, with 95% confidence, the true value of the expected average case running time divided by $\sqrt{N}$ lies in 1.851 ± 0.067 (in other words, the interval $[1.784, 1.918]$). The theoretical value of 1.812 lies in this interval.

Method	Theoretical value	Sample mean	95% confidence interval
Three-kangaroo	1.819	1.851	$[1.784, 1.918]$
Four-kangaroo	1.715	1.733	$[1.689, 1.777]$
Three-set Gaudry-Schost ($N = 2^{40}$)	1.761	1.814	$[1.796, 1.833]$
Four-set Gaudry-Schost ($N = 2^{40}$)	1.661	1.719	$[1.698, 1.739]$
Four-set Gaudry-Schost ($N \approx 2^{47}$)	1.661	1.686	$[1.624, 1.747]$

TABLE 1. Experimental results. The numbers are total number of group operations to solve the DLP, divided by $\sqrt{N}$.

The results confirm our conjecture about the performance of the four-kangaroo method. Note that the group operation counts in Table 1 exclude the group operations that are part of the set-up of the algorithm (i.e., those required to compute the set of jumps for the walks and to compute the starting values of the tame and wild walks). These steps are negligible as $N \to \infty$.

5.2. **Four-Set Gaudry-Schost Method.** The first set of experiments that we ran on the (improved) four-set Gaudry-Schost method was for an interval of size $N = 2^{40}$. We implemented the algorithm using jumps of even length. Recall that the theoretical prediction is $1.661\sqrt{N}$ group operations. To get accurate data, we performed 10000 experiments (i.e., more than were performed for the kangaroo method; since it is better understood and easier to analyse).

We took $\theta = 1/500$ so that each of the 4 lists contains around 900 group elements. We used walks as in the kangaroo method (i.e., jumps only to the right rather than "side-to-side" walks). The mean step size of the jumps was taken to be around $0.001 \cdot N \cdot \theta$ so that a typical walk covers a distance of around $0.001N$. We used 32 partitions of the group, using the same hash function to determine partitions. As in [17] we disallowed walks to start in the extreme right hand end of the regions (we omitted an interval of length $0.00001N$).

The results are given in Table 1. We also give results for the three-set algorithm. Note that the theoretical prediction 1.661 for the constant does not lie in the confidence interval (similarly, for the three-set method, the value 1.761 does not lie in the confidence interval). In both cases, the constants from the experimental results are around 0.05 to 0.06 greater than the theoretical prediction. This indicates that our implementation was not able to achieve the theoretical running time.

Finding the best way to implement the Gaudry-Schost algorithm is a significant challenge compared with other methods. In particular, if the walks are too "local" or if they overflow the regions "too often" then the performance can be a lot worse than desired.

However, following some suggestions by Montenegro [12] we ran a further 1000 experiments for a larger interval of size $N \approx 2^{47}$. We took $\theta = 1/4000$ so that each of the 4 lists contains around 1340 group elements. We used walks as in the kangaroo method (i.e., jumps only to the right rather than "side-to-side" walks). The mean step size of the jumps was taken to be around $0.001 \cdot N \cdot \theta$ so that a typical walk covers a distance of around $0.001N$. Critically we used 128 partitions of the group instead of only 32 partitions as in the earlier experiment, using a similar hash function to determine partitions. As in [17] we disallow walks to start in the extreme right hand end of the regions (we omitted an interval of length $0.00001N$).

The average constant of these results was 1.686 which is significantly closer to the theoretical prediction 1.661. We believe this is due, both, to running the algorithm over a larger interval size as well as partitioning the group into 128 parts rather than 32. As before, we computed a confidence interval so that, with 95% confidence, the true value of the expected average case running time divided by $\sqrt{N}$ lies in 1.686 ± 0.062 (in other words, the interval $[1.624, 1.747]$). The theoretical value 1.661 does lie in this interval, so these 'refined' experiments do not dispute the theoretical running time. These results are also given in Table 1.

As in the four-kangaroo case, we have ignored the set-up costs. Further, and more serious, we did not count the full cost of the group operations needed to restart the walks after hitting a distinguished point. Such costs can be reduced by using precomputation and storage of certain powers of g, and other strategies for fast exponentiation. In any case, these costs become negligible if the average length $1/\theta$ of the walks is long, but they would have been significant in our case.

The running times can probably be slightly improved by taking different choices of parameters, as was shown by our refined experiment. To conclude, though the four-set Gaudry-Schost algorithm is, asymptotically and in theory, superior to the four-kangaroo method, it may be more difficult to realise this improvement in practice for relatively small groups.

6. Conclusions

We have given two methods that improve the solution of the DLP in an interval in an arbitrary group. At most two inversions in the group are needed, so our methods are applicable in groups such as $\mathbb{F}_p^*$ where inversion is slow. For the discrete logarithm problem in elliptic curve groups one would prefer the methods of [4].

Our experimental results confirm that the algorithms work well in practice and that one can solve the discrete logarithm problem in an interval of size N in approximately $1.686\sqrt{N}$ group operations. This is a significant improvement over the previous methods, which required $2\sqrt{N}$ group operations. The kangaroo method is straightforward to implement and so is probably the preferred choice in practice for smaller intervals. There are some subtleties when parallelising the kangaroo method (see [16] for discussion) whereas the Gaudry-Schost method is easy to parallelise (see [6, 17]). Thus the Gaudry-Schost methods seem the preferred methods when parallelising and/or for larger intervals.

References

[1] D. Boneh, E.-J. Goh and K. Nissim, Evaluating 2-DNF Formulas on Ciphertexts, in J. Kilian (ed.), TCC 2005, Springer LNCS 3378 (2005) 325–341.

[2] J. H. Cheon, Security Analysis of the Strong Diffie-Hellman Problem, in S. Vaudenay (ed.), EUROCRYPT 2006, Springer LNCS 4004 (2006) 1–11.

[3] S. D. Galbraith and R. S. Ruprai, An improvement to the Gaudry-Schost algorithm for multidimensional discrete logarithm problems, in M. Parker (ed.), Twelfth IMA International Conference on Cryptography and Coding, Cirencester, Springer LNCS 5921 (2009) 368–382.

[4] S. D. Galbraith and R. S. Ruprai, Using Equivalence Classes to Accelerate Solving the Discrete Logarithm Problem in a Short Interval, in P. Q. Nguyen and D. Pointcheval (eds.), PKC 2010, Springer LNCS 6056 (2010) 368–383.

[5] S. D. Galbraith and M. Holmes, A non-uniform birthday problem with applications to discrete logarithms, Preprint (2010).

[6] P. Gaudry and E. Schost, A low-memory parallel version of Matsuo, Chao and Tsujii's algorithm, in D. A. Buell (ed.), ANTS VI, Springer LNCS 3076 (2004) 208–222.

[7] R. Gennaro, An Improved Pseudo-random Generator Based on Discrete Log, in M. Bellare (ed.), CRYPTO 2000, Springer LNCS 1880 (2000) 469–481.

[8] K. Gopalakrishnan, N. Thériault, and C. Z. Yao, Solving Discrete Logarithms from Partial Knowledge of the Key, in K. Srinathan, C. P. Rangan, and M. Yung, (eds.), INDOCRYPT 2007, Springer LNCS 4859 (2007) 224–237.

[9] D. Jao and K. Yoshida, Boneh-Boyen signatures and the Strong Diffie-Hellman problem, in H. Shacham and B. Waters (eds.), *Pairing 2009*, Springer LNCS 5671 (2009) 1–16.

[10] C. H. Lim and P. J. Lee, A Key Recovery Attack on Discrete Log-based Schemes Using a Prime Order Subgroup, in B. S. Kaliski Jr. (ed.), CRYPTO 1997, Springer LNCS 1294 (1997) 249–263.

[11] R. Montenegro and P. Tetali, How long does it take to catch a wild kangaroo? STOC 2009, ACM (2009) 553–560.

[12] R. Montenegro, Remarks about random walks in the Gaudry-Schost algorithm, Emails 4 January and 5 January 2011.

[13] S. Patel and G. Sundaram, An Efficient Discrete Log Pseudo Random Generator, in H. Krawczyk (ed.), CRYPTO 1998, Springer LNCS 1462 (1998) 304–317.

[14] S. Pohlig and M. Hellman, An improved algorithm for computing logarithms over GF(p) and its cryptographic significance, *IEEE Trans. Inf. Theory*, **24** (1978) 106–110.

[15] J. M. Pollard, Monte Carlo methods for index computation mod p, *Mathematics of Computation*, **32**, No. 143 (1978) 918–924.

[16] J. M. Pollard, Kangaroos, Monopoly and discrete logarithms, *Journal of Cryptology*, **13** (2000) 437–447.

[17] R. S. Ruprai, Improvements to the Gaudry-Schost Algorithm for Multidimensional discrete logarithm problems and Applications, PhD thesis, Royal Holloway University of London, 2009.

[18] B. I. Selivanov, On waiting time in the scheme of random allocation of coloured particles, *Discrete Math. Appl.*, **5**, No. 1 (1995) 73–82.

[19] P. C. van Oorschot and M. J. Wiener, Parallel collision search with application to hash functions and discrete logarithms, In *ACM Conference on Computer and Communications Security*, (1994) 210–218.

[20] P. C. van Oorschot and M. J. Wiener, On Diffie-Hellman Key Agreement with Short Exponents, in U. M. Maurer (ed.), EUROCRYPT 1996, Springer LNCS 1070 (1996) 332–343.

[21] P. C. van Oorschot and M. J. Wiener, Parallel collision search with cryptanalytic applications, *Journal of Cryptology*, **12** (1999) 1–28.

E-mail address: S.Galbraith@math.auckland.ac.nz

Mathematics Department, The University of Auckland, Private Bag 92019, Auckland 1142, New Zealand. Phone: (+64 9) 9238778 FAX: (+64 9) 3737457

E-mail address: jmptidcott@googlemail.com

Tidmarsh Cottage, Manor Farm Lane, Tidmarsh, Reading, Berkshire RG8 8EX, UK.

E-mail address: raminder@email.com

Mathematics Department, Royal Holloway University of London, Egham, Surrey TW20 0EX, UK.