# THE PARALLELIZED POLLARD KANGAROO METHOD IN REAL QUADRATIC FUNCTION FIELDS

ANDREAS STEIN AND EDLYN TESKE

Abstract. We show how to use the parallelized kangaroo method for computing invariants in real quadratic function fields. Specifically, we show how to apply the kangaroo method to the infrastructure in these fields. We also show how to speed up the computation by using heuristics on the distribution of the divisor class number, and by using the relatively inexpensive baby steps in the real quadratic model of a hyperelliptic function field. Furthermore, we provide examples for regulators and class numbers of hyperelliptic function fields of genus 3 that are larger than those ever reported before.

## 1. Introduction

In this paper we generalize the parallelized kangaroo method for computing invariants in real quadratic function fields. Basic such invariants of a real quadratic function field are the regulator and the divisor class number which play an important role in cryptosystems based on hyperelliptic function fields. For example, in the key-exchange protocol by Scheidler, Stein and Williams [SSW96], the regulator provides a measure for the key space; moreover, computation of the regulator is an instance of solving the discrete logarithm problem in real quadratic function fields.

The Pollard kangaroo method [Pol78], also called the lambda method, was originally developed to compute discrete logarithms in $\mathbb{Z}/p\mathbb{Z}$ and has been canonically generalized to solve the discrete logarithm problem in any finite abelian group. The key ingredient for the kangaroo method is that we know that the discrete logarithm lies in a given interval $[a, b[$; then the expected running time is $O(\sqrt{b-a})$ group operations. Van Oorschot and Wiener [vOW99] have shown that the kangaroo method can be parallelized with linear speed-up, which makes the method attractive for distributed attacks. It is important to note that the serial version of the kangaroo method is very space efficient, and that the space requirements of the parallelized version can be adjusted to the constraints of the machines. This is a big advantage over square-root attacks based on Shanks' baby step-giant step method [Sha71].

The objects with which we deal in real quadratic function fields are reduced principal ideals, which do *not* constitute a group. However, the baby step-giant step method could be efficiently adapted to this setting by defining analogues of baby steps and giant steps that make use of the ideal arithmetic (see [SZ91]). The disadvantage of this method is that the space restrictions of the machines constitute a bound on how large regulators can be computed. Currently, this bound is at about 25 digits ([SW98, ST99b]). It is therefore natural to ask whether such a space efficient method as the kangaroo method can be employed to real

*Date*: June 7, 2000.
1991 *Mathematics Subject Classification.* Primary 11Y16, 11Y40, 11R29. Secondary 11R58, 14H05.

quadratic function fields. In this paper we show that this indeed can be done. This allows us to push the range for regulator and class number computation much further. In fact, we give experimental results for the computation of a 29-digit class number and regulator with the parallelized Pollard kangaroo method. We used the computer algebra system SIMATH [Zim97], on 16 variably fast machines (including SGI Challenge Workstations and Sun Ultra Enterprise 450s), and found two matching distinguished points after 109 hours. We estimate that on a network of 16 Sun Ultra Enterprise 450 under Solaris 2.6 that computation of a 29-digit class number and regulator would have taken about 73 hours.

In the following, we first give an overview of the kangaroo method and its parallelization, where in the parallelized case we deal with both the variants of van Oorschot and Wiener [vOW99] and Pollard [Pol]. We keep this exposition as general as possible. Since no experimental results with the parallelized kangaroo method (not to mix with the parallelized rho method!) have been published so far, we also include some statistics about experiments with elliptic curve groups showing that the practical performance matches the theoretical predictions. Then, in Section 3, we give the basic definitions and facts needed about real quadratic function fields. In Section 4 we specialize the kangaroo method for the problem of regulator and class number computation in function fields and give experimental results.

## 2. The Parallelized Pollard Kangaroo Method

Let $G$ be a finite cyclic group generated by the group element $g$, and let $h \in G$. Then there exists a least positive number $x$ such that $h = g^x$. This number $x$ is usually called the *discrete logarithm*, or *index*, *of* $h$ *to the base* $g$. In the special case that $h = 1$, we call $x$ the *order* of $g$. We want to compute $x$. Let us assume that we know integers $a$ and $b$ such that $a \leq x < b$. This is the setting for which Pollard [Pol78] has designed his kangaroo method: The idea is to define two kangaroos, a tame kangaroo $T$ with starting point $t_0 = g^b$ and a wild kangaroo $W$ with starting point $w_0 = h$. In terms of the exponents of $g$, $T$ starts at the upper end of the interval $[a, b[$, while $W$ starts at an unknown spot $x$. Let $\delta_0(T) = b$, the initial distance of $T$ from the origin, and $\delta_0(W) = 0$, the initial distance of $W$ from $h$. Let $S = \{g^{s_1}, \dots, g^{s_r}\}$ $(s_i > 0)$ be a set of jumps, and let $v : G \to \{1, \dots, r\}$ be a hash function. We think of the exponents $s_i$ as travel distances, which should be small in comparison with the length $b - a$ of the interval; Pollard [Pol78] suggested that the powers of two (starting with $2^0$) up to a certain size to be specified below might be a good choice. Now we let the tame kangaroo travel through the group, following the path

$$t_{j+1} = t_j * g^{s_{v(t_j)}} , \qquad j \in \mathbb{N}_0 . \tag{2.1}$$

While computing the path $(t_j)$, we keep track of the distances $\delta_j(T)$,

$$\delta_{j+1}(T) = \delta_j(T) + s_{v(t_j)} , \qquad j \in \mathbb{N}_0 . \tag{2.2}$$

Hence, $t_j = g^{\delta_j(T)}$, for each $j \in \mathbb{N}_0$. After a certain number of jumps, the tame kangaroo stops and installs a trap at its final spot, say $t_M$. Then the wild kangaroo is set off, following the path

$$w_{j+1} = w_j * g^{s_{v(w_j)}} , \qquad j \in \mathbb{N}_0 , \tag{2.3}$$

with the distances $\delta_j(W)$ given by

$$\delta_{j+1}(W) = \delta_j(W) + s_{v(w_j)} \;, \qquad j \in \mathbb{N}_0 \;. \tag{2.4}$$

Then $w_j = h * g^{\delta_j(W)}$ ($j \in \mathbb{N}_0$). Hence, in terms of the exponents of $g$, at each stage we know the exact position of $T$, but we do not know the position of $W$ since its starting spot $x$ is unknown; this is why $W$ is wild. After each jump it is checked whether $W$ has fallen into the trap. If this is the case, say at $w_N$, the equation $t_M = w_N$ immediately yields a solution of $g^x = h$, namely $x = \delta_M(T) - \delta_N(W)$. Otherwise, after a certain number of jumps the wild kangaroo is halted, and a new wild kangaroo with starting point $w_0 = h * g^z$ and initial distance $\delta_0(W) = z$ ($z$ small) is set off. If $W$ falls into $T$'s trap, this means that the paths of the two kangaroos must have met earlier during the travel, from which point on they have been identical. In this case, if we draw the two paths on a piece of paper starting at the bottom left and the bottom right, respectively, we obtain the Greek letter lambda, which explains why the method is also called the lambda method. Van Oorschot and Wiener [vOW99] have analyzed this method and found out that the running time is expected to be minimal if the mean value of $S$ is approximately $(\sqrt{b-a})/2$, and if $T$ makes approximately $0.7\sqrt{b-a}$ jumps before placing the trap. Then after about $2.7\sqrt{b-a}$ jumps of the wild kangaroo, $W$ either must have fallen into the trap, which happens with probability $\approx 0.75$, or it must have safely passed the trap and should be halted. The expected total running time amounts to $3.28\sqrt{b-a}$ group operations. The algorithm has to store only the set $S$ and the current positions of the two kangaroos; since $|S| = O(\log(b-a))$ if the exponents $s_i$ are powers of two, it is very space efficient.

2.1. **The van Oorschot-Wiener parallelization.** In the same paper [vOW99], van Oorschot and Wiener improved on the above result by applying a *distinguished point method*. Their technique suits both for a better serial application of the kangaroo method and for its efficient parallelization. The idea is to call a group element (="point") *distinguished* if it satisfies some easily checkable property. For example, one could require that certain bits of the binary representation of a distinguished point must be zero. Whenever a kangaroo hits a distinguished point, say $t_j$ or $w_j$, it is stored, together with the corresponding distance $\delta_j$. In the parallel setting, this information is sent to a central server; in this case, the name of the corresponding kangaroo should be included to be able to deal with collisions between members of the same herd. As soon as the same distinguished point has been found by both a tame and a wild kangaroo, the discrete logarithm can be derived from the corresponding distances. We describe this method in more detail. Let $m$ denote the number of processors. Let us first assume that $m$ is even. We work with $u = m/2$ tame and $v = m/2$ wild kangaroos. The tame kangaroos $T_0, \ldots, T_{u-1}$ start at and shortly after the middle of the interval, at positions $t_0(T_k) = g^{(a+b)/2+k\nu}$, $0 \le k < u$, with some small constant $\nu$. ( Note that $\nu$ should be chosen with some care to reduce the possibility of collisions between kangaroos of the same herd. For example, we should have $\nu \neq s_i$, $i = 1, \ldots, r$.) The wild kangaroos $W_0, \ldots, W_{v-1}$ start at $w_0(W_k) = h * g^{k\nu}$, $0 \le k < v$. Then the kangaroos follow paths as in (2.1) and (2.3), where again we keep track of the distances (2.2) and (2.4), just as in the original kangaroo method described above. Let $d$ denote the distance between the herds of tame and wild kangaroos, that is, $d = |(a+b)/2 - x|$, and let $\beta$ be the mean value of the exponents $s_i$ in $S$. Then the expected running time is

$$d/\beta + (2/m)^2\beta + 1/\Theta \;,$$

where $\Theta$ denotes the proportion of points that satisfy the distinguishing property. Since $d$ is unknown, the analysis [vOW99] assumes an expected value for $d$ of $(b-a)/4$. The expected running time until two different kangaroos hit the same distinguished point is minimal if $\beta$ is approximately $m(\sqrt{b-a})/4$ which gives an expected running time of

$$(1+\tfrac{4d}{b-a})(\sqrt{b-a})/m + 1/\Theta \tag{2.5}$$

jumps on each processor. Observe that $1 \leq 1 + 4d/(b-a) \leq 3$. If $d$ is approximately $(b-a)/4$, that is $x \approx (a+b)/2 \pm (b-a)/4$, it takes an expected number of approximately

$$2(\sqrt{b-a})/m + 1/\Theta$$

jumps on each processor. This is, on average, true for a randomly chosen $x$, but in general does not hold for a specific $x$.

If $m$ is odd, we also can simulate $2m$ half-speed processors by alternating between jumps of one tame and one wild kangaroo. This takes an expected number of $2(\sqrt{b-a})/m + 2/\Theta$ jumps on each (real) processor.

Notice that if we have more information about the distribution of $d$ in the interval $[a, b[$ – as it is the case for real quadratic function fields – it may be useful to alter $\beta$ and thus obtain a different expected running time. For example, if $E(d)$ is the expected value for $d$ then the optimal choice for $\beta$ is

$$\beta = m\sqrt{E(d)}/2 \ , \tag{2.6}$$

which gives an expected running time of $4\sqrt{E(d)}/m + 1/\Theta$ jumps on each processor.

On comparing the expected number of jumps (= group operations) for the kangaroo method using distinguished points with the expected number of $(\sqrt{\pi \cdot \operatorname{ord} g/2})/m + 1/\Theta$ group operations for the parallelized rho method (cf. [vOW99]), we see that the kangaroo method is expected to be faster than the rho method if $b - a < \pi/8 \cdot \operatorname{ord} g$. ($\pi/8 \approx 0.39$.)

The parallelization method of van Oorschot and Wiener does not exclude collisions between kangaroos of the same herd. Such collisions do not reveal any information about the value $x$ to be computed. Since the kangaroos travel on the same paths once they have collided, computing power is wasted if not one of the colliding kangaroos is halted. Halting a kangaroo (with the purpose to restart it at a new starting point) requires a message from the central server to the corresponding machine, an effort which one would like to avoid.

2.2. **Pollard's parallelization.** Pollard [Pol] found a method of parallelization such that collisions between kangaroos of the same herd cannot occur. His method works as follows. Let $m$ be the number of processors and let $u$ and $v$ denote the numbers of tame and wild kangaroos, respectively. We choose $u$ and $v$ to be coprime, nearly equal and such that $u+v \leq m$. The powers of $g$ in the set of jumps are multiples of $uv$: $S = \{g^{s_1 uv}, g^{s_2 uv}, \ldots, g^{s_r uv}\}$. As before, the tame kangaroos $T_0, \ldots, T_{u-1}$ are set off at and shortly after the middle of the interval, now at positions $t_0(T_i) = g^{(a+b)/2+iv}$, $0 \leq i < u$, and the wild kangaroos $W_0, \ldots, W_{v-1}$ start at positions $w_0(W_k) = h * g^{ku}$, $0 \leq k < v$. Then they travel through the group just as before. Since the equation

$$(a+b)/2 + iv = x + ku \mod uv \qquad (x = \log_g h)$$

has a unique solution in $i \pmod u$ and $k \pmod v$, there is exactly one pair $(T_i, W_k)$ of kangaroos that travel in the same residue class modulo $uv$; this is the only pair which can meet. No two tame and no two wild kangaroos can meet each other. Hence, as soon as any two kangaroos have met and a distinguished point $t_j(T_i) = w_{j'}(W_k)$ has been found, the unknown $x$ can be determined from the starting points $t_0(T_i)$ and $w_0(W_k)$ and distances $\delta_j(T_i)$ and $\delta_{j'}(W_k)$. Pollard's analysis of this method shows that the optimal choice for $S$ is such that the mean value of the $s_i$ is close to $(\sqrt{(b-a)/uv})/2$. Then it takes an expected number of approximately $(\sqrt{(b-a)/(uv)})B(S) + 1/\Theta$ jumps on each processor. Here, $B(S)$ is a correction factor depending on the choice of $S$; Pollard [Pol] gives experimental evidence that $B(S) \approx 1$ if the exponents $s_i$ in $S$ are the consecutive powers of 2 or 4 (starting with $2^0$ resp. $4^0$) and $|S| \geq 6$. Since both $u$ and $v$ are close to $m/2$, this then gives essentially the same expected running time as for the method by van Oorschot and Wiener.

2.3. **The parallelized Pollard kangaroo method in practice.** There are several aspects to consider when it comes to actually implementing a parallelized Pollard kangaroo attack. We begin with some typical experimental data, shown in Tables 1 – 3, to illustrate our discussion. Using the computer algebra system LiDIA [LiD97], we implemented both the van Oorschot-Wiener and the Pollard variants of parallelization for groups of points of elliptic curves over finite fields, to compute the group order. We simulated the parallelization on a serial computer; this agrees with the assumption in the running-time analysis that all machines are equally fast.

In our first example we work with the curve $E_{7,13} : y^2 = x^3 + 7x + 13$ defined over the finite field $\mathbb{F}_p$ with the 15-digit prime $p = 100000005000197$; the point group $E_{7,13}(\mathbb{F}_p)$ is cyclic. A theorem due to Hasse says that its order $n$ satisfies the inequality $|n - p - 1| \leq 2\sqrt{p}$; in fact $n = 100000015380435$. We choose a random point $P$ on the curve and we use the parallelized kangaroo method to find a multiple of its order in the interval $[p + 1 - 2\sqrt{p}, p + 2 + 2\sqrt{p}[$. Note that if $\operatorname{ord} P > 4\sqrt{p}$, then the unique multiple of $\operatorname{ord} P$ in the interval must be the group order. We have $d = |n - p - 1| = 1.038...\sqrt{p}$. This is the distance between the herd of tame kangaroos, set off near $(p + 1) \cdot P$, and the herd of wild kangaroos, set off near the neutral element $\mathcal{O}$ of $E_{7,13}(\mathbb{F}_p)$, and it is approximately the distance on which the running time analysis in [vOW99] and [Pol] is based. Using (2.5), we then find that $\alpha = 2.038...$, and the expected running time is $12889/m + 1/\Theta$ jumps/kangaroo. We work with $m = 4$, $m = 8$ and $m = 20$ kangaroos.

In this and the following examples, we fix the remaining parameters as follows: We use various sizes for the sets of distinguished points, from $\Theta = 1, 2^{-1}, \ldots$ to $2^{-12}$. As for the jump distances, for the van Oorschot/Wiener parallelization we let $s_i = 2^{i-1}$ $(1 \leq i < r)$ and choose $0 < s_r \leq 2^{r-1}$ such that the mean value of the $s_i$ equals the nearest integer of $mp^{1/4}/2$ – notice that $r$ is uniquely determined by these conditions. Correspondingly, for the Pollard parallelization we work with jump distances $s_i uv$ where $s_i = 2^{i-1}$ for $1 \leq i < r$ and $s_r$ is such that the mean value of the $s_i$ equals $\lfloor p^{1/4}/\sqrt{uv} \rfloor$, where $u = m/2 - 1$ and $v = m/2 + 1$ are the numbers of tame and wild kangaroos, respectively. Finally, in the van Oorschot-Wiener variant we work with $\nu = 13$ as distance between the starting points of two consecutive kangaroos of the same herd; whenever two tame or two wild kangaroos meet (which is signalled by their finding the same distinguished point), one of these two kangaroos is halted and set off at a new starting point.

| $\Theta$ | 1 | | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ | $2^{-5}$ | $2^{-6}$ | ... | $2^{-12}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 kangaroos. Expected: $3222 + 1/\Theta$ jumps/kangaroo | | | | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | | | | |
| dist. points | 13861 | | 6932 | 3467 | 1735 | 870 | 437 | 220 | ... | 6 |
| jumps/kang. | 3465 | (2.191) | 3467 | 3469 | 3475 | 3486 | 3505 | 3544 | ... | 7664 |
| Pollard's variant | | | | | | | | | | |
| dist. points | 15690 | | 7851 | 3924 | 1963 | 983 | 493 | 248 | ... | 7 |
| jumps/kang. | 3923 | (2.481) | 3926 | 3928 | 3932 | 3941 | 3957 | 3989 | ... | 7939 |
| 8 kangaroos. Expected: $1611 + 1/\Theta$ jumps/kangaroo | | | | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | | | | |
| dist. points | 13917 | | 6965 | 3484 | 1744 | 876 | 442 | 224 | ... | 7 |
| jumps/kang. | 1740 | (2.201) | 1741 | 1743 | 1748 | 1758 | 1774 | 1811 | ... | 4971 |
| Pollard's variant | | | | | | | | | | |
| dist. points | 13727 | | 6864 | 3436 | 1722 | 865 | 436 | 222 | ... | 11 |
| jumps/kang. | 1716 | (2.17) | 1717 | 1719 | 1723 | 1731 | 1747 | 1779 | ... | 5992 |
| 20 kangaroos. Expected: $644 + 1/\Theta$ jumps/kangaroo | | | | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | | | | |
| dist. points | 13389 | | 6702 | 3361 | 1690 | 855 | 436 | 227 | ... | 9 |
| jumps/kang. | 669 | (2.115) | 670 | 672 | 677 | 685 | 701 | 733 | ... | 2443 |
| Pollard's variant | | | | | | | | | | |
| dist. points | 13769 | | 6891 | 3456 | 1735 | 876 | 448 | 233 | ... | 23 |
| jumps/kang. | 688 | (2.175) | 689 | 691 | 695 | 702 | 719 | 750 | ... | 4812 |

TABLE 1. $E_{7,13}(\mathbb{F}_p)$ with $p = 100000005000197$. Order computation.

Now for each order computation, we counted the number of jumps of each kangaroo until any pair of tame and wild kangaroos found the same distinguished point, and the total number of distinguished points found by all kangaroos. Table 1 shows the average values taken over 1000 such runs; here we considered only those runs with points $P$ whose order exceeded $4\sqrt{p}$. The number in brackets in the second column shows the average ratio of the number of jumps and $(\sqrt{b-a})/m$, for which the theoretical value is $\alpha = 2.038$. Note that in the van Oorschot-Wiener variant, on average at most one collision occurred between kangaroos of the same herd when $m = 4$ and $m = 8$, and at most two such collisions when $m = 20$.

In our second example, we work with the curve $E_{5,17} : y^2 = x^3 + 5x + 17$ defined over $\mathbb{F}_p$, where $p = 100000000000000500053$ (21 digits). $E_{5,17}(\mathbb{F}_p)$ is cyclic of group order $n = 99999999991517342872$. Hence, $d = |n - p - 1| = 0.848...\sqrt{p}$, $\alpha = 1.848$, and we expect an average running time of about $369663/m + 1/\Theta$ jumps for each kangaroo. With $m = 8$ and $m = 20$ kangaroos, we conduct the same computations as in the first example. The average values of the results, this time taken over 100 computations with $\operatorname{ord} P > 4\sqrt{p}$, are shown in Table 2. Also, on average at most one collision occurred between kangaroos of the same herd, both for $m = 8$ and $m = 20$ (in the van Oorschot-Wiener variant).

In our last example, the curve is $E_{5,17} : y^2 = x^3 + 13x + 5$ over $\mathbb{F}_p$ with the 16-digit prime $p = 1000000000000037$. The group $E_{5,17}(\mathbb{F}_p)$ is cyclic and has order $n = 1000000058247036$.

| $\Theta$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ | $2^{-5}$ | $2^{-6}$ | ... | $2^{-12}$ |
|---|---|---|---|---|---|---|---|
| 8 kangaroos. Expected: $46208 + 1/\Theta$ jumps/kangaroo | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | |
| dist. points | 97746 | 48861 | 24443 | 12232 | 6120 | ... | 104 |
| jumps/kang. | 48871 (1.954) | 48876 | 48885 | 48903 | 48934 | ... | 54035 |
| Pollard's variant | | | | | | | |
| dist. points | 107388 | 53681 | 26824 | 13416 | 6703 | ... | 112 |
| jumps/kang. | 53677 (2.147) | 53681 | 53690 | 53706 | 53735 | ... | 57577 |
| 20 kangaroos. Expected: $18483 + 1/\Theta$ jumps/kangaroo | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | |
| dist. points | 87349 | 43684 | 21850 | 10943 | 5487 | ... | 102 |
| jumps/kang. | 17478 (1.747) | 17482 | 17492 | 17508 | 17550 | ... | 21141 |
| Pollard's variant | | | | | | | |
| dist. points | 89903 | 44954 | 22480 | 11239 | 5632 | ... | 106 |
| jumps/kang. | 17975 (1.797) | 17978 | 17985 | 17998 | 18029 | ... | 22005 |

TABLE 2. $E_{5,17}(\mathbb{F}_p)$ with $p = 1000000000000000500053$. Order computation.

| $\Theta$ | 1 | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ | $2^{-5}$ | $2^{-6}$ | ... | $2^{-12}$ |
|---|---|---|---|---|---|---|---|---|---|
| 8 kangaroos. Expected: $3995 + 1/\Theta$ jumps/kangaroo | | | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | | | |
| dist. points | 32566 | 16276 | 8139 | 4071 | 2040 | 1025 | 514 | ... | 12 |
| jumps/kang. | 4071 (2.895) | 4072 | 4074 | 4079 | 4091 | 4112 | 4149 | ... | 7640 |
| Pollard's variant | | | | | | | | | |
| dist. points | 30364 | 15184 | 7585 | 3790 | 1903 | 954 | 480 | ... | 14 |
| jumps/kang. | 3795 (2.699) | 3797 | 3799 | 3802 | 3811 | 3825 | 3855 | ... | 7332 |
| 20 kangaroos. Expected: $1598 + 1/\Theta$ jumps/kangaroo | | | | | | | | | |
| van Oorschot-Wiener variant | | | | | | | | | |
| dist. points | 29418 | 14724 | 7371 | 3689 | 1860 | 937 | 481 | ... | 15 |
| jumps/kang. | 1471 (2.616) | 1472 | 1475 | 1479 | 1489 | 1504 | 1543 | ... | 3666 |
| Pollard's variant | | | | | | | | | |
| dist. points | 32974 | 16510 | 8265 | 4146 | 2080 | 1048 | 531 | ... | 27 |
| jumps/kang. | 1649 (2.932) | 1650 | 1651 | 1655 | 1662 | 1677 | 1706 | ... | 5536 |

TABLE 3. $E_{13,5}(\mathbb{F}_p)$ with $p = 1000000000000037$. Order computation.

This time the distance between the herds of tame and wild kangaroos is larger than $\sqrt{p}$. Indeed, we have $d = |n - p - 1| \approx 1.842$, and $\alpha \approx 2.842$. We then expect, on average, about $31963/m + 1/\Theta$ jumps for each kangaroo. Our results, this time for $m = 8$ and $m = 20$ kangaroos, are shown in Table 3. All averages are taken over 1000 computations. In the van Oorschot-Wiener variant, we observed on average at most one collision between members of the same herd when $m = 8$ and at most two collisions when $m = 20$.

2.3.1. *Which variant: van Oorschot-Wiener or Pollard?* Our experimental data suggest that with both variants, we achieve an almost linear speed-up. The Pollard variant of parallelization has the advantage that collisions between kangaroos of the same herd are impossible. In the van Oorschot-Wiener variant we do have to deal with such collisions, which requires communication from the central server to the processors. In our experiments, the number of collision was quite small, but this effect may be more significant when dealing with much larger herds of kangaroos. On the other hand, the van Oorschot-Wiener variant allows that additional kangaroos (=processors) can join a parallelized kangaroo attack at any stage and contribute with each single distinguished point they find. Thus, it mainly depends on the application which variant suits better.

2.3.2. *The distinguished point set.* A simple way to implement the distinguished point set is to define a point to be distinguished if the $F$ lowest bits in the representation of the point as a binary string are zero. Then the proportion of distinguished points is $\Theta = 1/2^F$. In our experiments, we worked with $F = 0, \ldots, 6, 12$. Our results fully match the theoretical predictions how $\Theta$ effects the running time. In general, $F$ should be chosen small enough such that $1/\Theta = 2^F$ is small compared with $\sqrt{b-a}$, but not too small since the algorithm has to store an expected number of $2\Theta\sqrt{b-a}$ distinguished points.

2.3.3. *The set of jumps.* Our experimental results suggest that choosing the exponents $s_i$ in the set of jumps as powers of two in principal works fine in the sense that the theoretically predicted running times are almost met. Sample experiments with choosing the $s_i$ uniformly at random from the interval $[0, 2\mu]$ ($\mu$ denoting the theoretically optimal mean value of the $s_i$) yielded even slightly better results.

2.3.4. *Choice of the spacings $\nu$ in the van Oorschot-Wiener variant.* In the experiments shown in our tables, we worked only with $\nu = 13$. Working with other choices of $\nu$ such as the primes 31, 1031, 1000031 yielded almost identical results. It is subject to further investigation whether the number of useless collisions can be reduced by using another strategy to choose the spacings, for example, by introducing any kind of randomization.

## 3. A Brief Introduction into Real Quadratic Function Fields

3.1. **Basic definitions.** For details about function fields we refer to [Sti93] and for the arithmetic of real quadratic function fields we mention [Art24, SSW96, PR99]. Let $k = \mathbb{F}_q$ be a finite field of odd characteristic with $q$ elements. A *hyperelliptic function field* over the constant field $k$ of genus $g$ is a quadratic extension of the rational function field over $k$ in one variable, i.e. $K = k(X)(\sqrt{D})$, where $D = D(X)$ is a squarefree polynomial of degree $2g+1$ or $2g+2$. If $D$ is a monic, squarefree polynomial of degree $2g+2$, then $K$ is called a *real quadratic function field over* $k$, and otherwise *imaginary quadratic*.

Let $D$ be a monic, squarefree polynomial of degree $2g+2$ so that $K = k(X)(\sqrt{D})$ is a real quadratic function field over $k$ with respect to the real quadratic order $\mathcal{O}(X) = k[X][\sqrt{D}]$. In this case, the infinite place $\infty$ of $k(X)$ splits into two extensions $\infty_1$ and $\infty_2$ in $K$. We know that $\mathcal{O}(X)^* = k^* \times \langle\epsilon\rangle$, where $\epsilon \in K$ is a fundamental unit. We know that $K \leq \mathbb{F}_q((1/X))$, where $\mathbb{F}_q((1/X))$ denotes the field of Puiseux series. We then consider elements of $K$ as Puiseux series at $\infty_1$ in $1/X$. Now, let $\alpha \in \mathbb{F}_q((1/X))$ be a non-zero element. Then $\alpha = \sum_{i=-\infty}^{d} c_i X^i$ with $c_d \neq 0$. We denote by $\deg(\alpha) = d$ the *degree* of $\alpha$. We set $\deg(0) = -\infty$. We define the *regulator* $R_X$ of $K$ over $k$ with respect to $\mathcal{O}(X)$ as $R_X := \deg(\epsilon)$. We then have that $h = R_X h_X$ , where $h_X$ denotes the *ideal class number* of $K$ with respect to $\mathcal{O}(X)$ and $h$ the *divisor class number* of $K$. [1]

3.2. **Ideals.** The ring $\mathcal{O}(X)$ is a Dedekind domain. Any (non-zero integral $\mathcal{O}(X)$-)ideal can be uniquely represented in the form $\mathfrak{a} = SQ\, k[X] + (SP + S\sqrt{D})\, k[X]$ , where $S,\ P,\ Q \in k[X]$ with $Q|(D - P^2)$, $\operatorname{sgn}(S) = \operatorname{sgn}(Q) = 1$, and $\deg(P) < \deg(Q)$; here, $\operatorname{sgn}(S)$ denotes the highest coefficient of $S$. The set $\{SQ, SP + S\sqrt{D}\}$ is called a $k[X]$-*basis* of $\mathfrak{a}$. If $\mathfrak{a}$ is given in the representation above, which we also call standard representation, then the *norm* of $\mathfrak{a}$ is defined by $N(\mathfrak{a}) = Q\, S^2 \in k[X]$. An ideal is called *primitive* if $S = 1$. If $\mathfrak{a} = (\alpha) = \alpha\mathcal{O}(X)$ with $\alpha \in \mathcal{O}(X)$, we call $\mathfrak{a}$ a *principal ideal.* We say that two integral ideals $\mathfrak{a}$ and $\mathfrak{b}$ are *equivalent*, written $\mathfrak{a} \sim \mathfrak{b}$, if there exist some non-zero elements $\alpha_1, \alpha_2 \in \mathcal{O}(X)$ such that $(\alpha_1)\mathfrak{a} = (\alpha_2)\mathfrak{b}$.

A primitive ideal $\mathfrak{a}$ is called *reduced,* if $\deg N(\mathfrak{a}) \leq g$. Hence, each reduced ideal $\mathfrak{a}$ can be uniquely represented by a pair $(Q, P)$ of polynomials in $k[X]$, where $Q|(D-P^2)$, $\operatorname{sgn}(Q) = 1$, i.e. $Q = N(\mathfrak{a})$, and $\deg(P) < \deg(Q) \leq g$. For instance, $\mathcal{O}(X)$ is a reduced principal ideal with representation (1, 0). It is well-known that each ideal class contains exactly one cycle of reduced ideals. So does the principal ideal class. Throughout our computations, we will only be performing arithmetic on reduced principal ideals, which will be represented by their $k[x]$-bases $\{Q, P + \sqrt{D}\}$. We denote by $\mathcal{R}$ the set of reduced principal ideals. The continued fraction expansion of ideals starting at $\mathfrak{r}_1 = \mathcal{O}(X)$ produces all elements of $\mathcal{R}$. This infinite process is ultimately periodic with period $p$, and thus $\mathcal{R} = \{\mathfrak{r}_1, \ldots, \mathfrak{r}_p\}$, where $\mathfrak{r}_{p+i} = \mathfrak{r}_i$ for $i \in \mathbb{N}$.

3.3. **Distance.** With each reduced principal ideal $\mathfrak{a} = (\alpha)$, we associate a *distance*

$$\delta(\mathfrak{a}) = \delta(\mathfrak{a}, \mathcal{O}(X)) := \deg(\alpha)\ . \tag{3.1}$$

¿From [SSW96] we know that $\delta$ is unique modulo $R_X$, and defines an order

$$\mathfrak{r}_1 = \mathcal{O}(X) < \mathfrak{r}_2 < \ldots < \mathfrak{r}_p < \mathfrak{r}_{p+1} < \ldots$$

on the reduced principal ideals (and in particular on $\mathcal{R}$). Furthermore, $\delta(\mathfrak{r}_i)$ is strictly increasing with $i$; more exactly, we have that $1 \leq \delta(\mathfrak{r}_{i+1}) - \delta(\mathfrak{r}_i) \leq g + 1$. Note that $\delta(\mathfrak{r}_2) = g + 1$ and $R_X = \delta(\mathfrak{r}_{p+1})$. Two reduced principal ideals $\mathfrak{a}$ and $\mathfrak{b}$ are equal if and only if

$$\delta(\mathfrak{a}) \equiv \delta(\mathfrak{b}) \pmod{R_X}\ .$$

Given any positive integer $y$, there certainly exists an index $j \geq 1$ such that

$$\delta(\mathfrak{r}_j) \leq y < \delta(\mathfrak{r}_{j+1})\ .$$

[1] If $C$ denotes the non-singular projective model of the function field $K$, then $C$ is an irreducible hyperelliptic curve of genus $g$. Thus, the function field $k(C)$ of $C$ is equal to $K = k(C) = k(X)(\sqrt{D}) = k(X)(Y)$, where $Y^2 = D(X)$. Thus, the divisor class number of $K$ is the cardinality of the Jacobian of the hyperelliptic curve $C$.

We call $\delta(\mathfrak{r}_j)$ the *closest ideal to* (the left of) $y$ and denote it by $\mathbf{Nearest}(y)$. In this situation, we put

$$n(y) = y - \delta(\mathbf{Nearest}(y)) = y - \delta(\mathfrak{r}_j) \ .$$

Following [PR99], we identify any integer $y$ between 0 and $R_X - 1$ with a tuple $(\mathbf{Nearest}(y), n(y))$, where $y = \delta(\mathbf{Nearest}(y)) + n(y)$ and $0 \le n(y) \le g$. That means there exists a unique representation for any integer $y$ between 0 and $R_X - 1$ in terms of an element $\mathbf{Nearest}(y) \in \mathcal{R}$ and an integer $n(y)$ which is the distance between $\mathbf{Nearest}(y)$ and $y$. Using that $\mathfrak{r}_{mp+j} = \mathfrak{r}_j$ and $\delta(\mathfrak{r}_{mp+j}) = mR_X + \delta(\mathfrak{r}_j)$ for $m \in \mathbb{N}$ and $j \ge 1$, we then in the obvious way extend this representation to all non-negative integers $y$; then also $n(s + mR_X) = n(s)$ for $m, s \in \mathbb{N}$. Possibly $n(y) = 0$, then there exists a reduced principal ideal with precise distance $y$. But, it might happen that $n(y) \neq 0$ in which case there is a "hole" in the cycle. For instance $\delta(\mathfrak{r}_1) = 0$ and $\delta(\mathfrak{r}_2) = g + 1$ so that $\mathbf{Nearest}(j) = \mathfrak{r}_1 = \mathcal{O}(X)$ for $j = 0, \dots, g$. Then the integers $0, 1, \dots, g, g+1$ are represented by the tuples $(\mathcal{O}(X), 0), (\mathcal{O}(X), 1), \dots, (\mathcal{O}(X), g), (\mathfrak{r}_2, 0)$, respectively. Note that $n(y) = g$ if and only if $y = g + mR_X$ for some $m \in \mathbb{N}$.

In our application we mainly consider the case that $q$ is large and $g$ is small. By picking random integers $y$ between $g + 1$ and $R_X$, we found experimental evidence that $n(y) = 0$ with probability $1 - O(1/q)$, and, in general, $n(y) = j$ for $1 \le j \le g - 1$ with probability $O(1/q^j)$.

We call one step in the continued fraction expansion of reduced principal ideals, i.e. the computation of $\mathfrak{r}_{i+1}$ from $\mathfrak{r}_i$, a *baby step (forwards)*. Similarly, the computation of $\mathfrak{r}_i$ from $\mathfrak{r}_{i+1}$ is said to be a *baby step (backwards)*. Generally, a baby step can be performed by using $4g + O(1)$ operations in the finite field $k$ (see [Ste99]).

3.4. **Infrastructure.** Shanks' infrastructure idea [Sha72] also applies to the set of reduced principal ideals in a real quadratic function field. We define an operation $\star$ (a *giant step* or an *infrastructure operation*) on $\mathcal{R}$ as follows: Let $\mathfrak{a} = (Q_a, P_a)$ and $\mathfrak{b} = (Q_b, P_b)$ be two reduced principal ideals. Firstly, compute the ideal product $\mathfrak{a}\mathfrak{b} = (S)\mathfrak{c}'$, where $\mathfrak{c}'$ is a primitive principal ideal. Secondly, reduce $\mathfrak{c}'$ to obtain an equivalent reduced principal ideal $\mathfrak{c}$. Put $\mathfrak{c} = \mathfrak{a} \star \mathfrak{b}$. If one knows the distances $\delta(\mathfrak{a})$ and $\delta(\mathfrak{b})$, then $\delta(\mathfrak{c})$ can be computed as well, and

$$\delta(\mathfrak{c}) = \delta(\mathfrak{a}) + \delta(\mathfrak{b}) + f \ , \tag{3.2}$$

where $-2g \le f \le 0$. This operation can be performed efficiently in an expected number of $17\, g^2 + O(g)$ operations in the finite field $k$ (see [Ste99]).

We remark here that we found experimental evidence (for $1 \le g \le 10$) that for randomly chosen reduced ideals $\mathfrak{a}$ and $\mathfrak{b}$ we have $f = -\lfloor g/2 \rfloor$ with probability $1 - O(1/q)$.

Furthermore, given any non-negative integer $y$, we may compute $\mathbf{Nearest}(y)$ effectively in $O(\log n)$ giant steps by using an adaption of the square-and-multiply strategy.

In the kangaroo method, the infrastructure will be applied as follows: For jump distances $s_i$, we compute the corresponding ideals $\mathbf{Nearest}(s_i)$. Then jumps of the kangaroos will be giant steps of the form $\mathfrak{z}_j \star \mathbf{Nearest}(s_i)$, with $\mathfrak{z}_j$ denoting the position of the kangaroo after $j$ jumps.

Notice that (3.2) means that kangaroos experience a slight headwind, as a consequence of which kangaroos tend to jump a bit too short. We can help the kangaroo by altering the set of jumps a bit: given the jump distances $s_i$, we work with the ideals $\mathbf{Nearest}(s_i + \lfloor g/2 \rfloor)$, such that jumps of the kangaroos are of the form $\mathfrak{z}_j \star \mathbf{Nearest}(s_i + \lfloor g/2 \rfloor)$. Then, with probability $1 - O(1/q)$ we have for $\mathfrak{z}_{j+1} = \mathfrak{z}_j \star \mathbf{Nearest}(s_i + \lfloor g/2 \rfloor)$ that $\delta(\mathfrak{z}_{j+1}) = \delta(\mathfrak{z}_j) + s_i$. In particular, this means that the expected running time of the algorithm is not affected by the headwind.

3.5. **An estimate of $h$.** Our main application of the kangaroo method for real quadratic function fields is to compute $R_X$ and $h$. For this, we first need to know integers $a$ and $b$ such that $a \le h = h_X R_X \le b$. Of course, $b - a$ should be as small as possible. Luckily, a reasonable interval for $h$ can be determined by evaluating the formulas in [ST99b]. The idea is to determine integers $E$ and $L$ such that

$$|h - E| < L^2 \ .$$

If we put $a = E - L^2 + 1$ and $b = E + L^2$, then $b - a = 2L^2 - 1$. Of course, we want that $L$ be as small as possible. For $g \le 2$, we find appropriate $E$ and $L$ using the inequality

$$(\sqrt{q} - 1)^{2g} \le h \le (\sqrt{q} + 1)^{2g}$$

which is a consequence of the Theorem of Hasse-Weil (see e.g. [Sti93, Theorem V.1.15 and V.2.1]). To compute $E$ and $L$ for $g \ge 3$, we make use of the analogue of the analytic class number formula for function fields and proceed with approximating $h$ by truncated Euler products. This works as follows. We know that

$$h = \frac{q^{g+1}}{q-1} \prod_P \frac{1}{1 - \chi(P) q^{-\deg(P)}} = \frac{q^{g+1}}{q-1} \prod_{\nu=1}^{\infty} \prod_{\substack{P \\ \deg(P)=\nu}} \frac{1}{1 - \chi(P) q^{-\nu}} \ ,$$

where $P$ runs through all monic prime polynomials of $k[X]$ and $\chi(P) = [D/P]$ denotes the Legendre symbol for polynomials of $D$ over $P$. Following the reasoning in [ST99b], we let

$$\lambda = \begin{cases} \lfloor (2g-1)/5 \rfloor & \text{if } g \equiv 2 \pmod 5 \ , \\ \operatorname{round}((2g-1)/5) & \text{otherwise} \ . \end{cases}$$

(round$(y)$ is the unique integer such that $-\frac{1}{2} < y - \operatorname{round}(y) \le \frac{1}{2}$.) We now put $E = \operatorname{round}(E')$, where

$$E' = \frac{q^{g+1}}{q-1} \prod_{\nu=1}^{\lambda} \prod_{\substack{P \\ \deg(P)=\nu}} \frac{1}{1 - \chi(P) q^{-\nu}} \ .$$

Let

$$\psi = \frac{(2g + \epsilon(\lambda)) q^{-\frac{\lambda}{2}}}{(\lambda+1)(\sqrt{q} - 1)} + \frac{(2g+2) q^{-\frac{\lambda-1}{2}}}{(\lambda+2)(\sqrt{q}-1)^3} \ ,$$

where $\epsilon(\lambda) = 0$ if $\lambda$ is even and $\epsilon(\lambda) = 1$ if $\lambda$ is odd. Then we define the integer $L$ by

$$L = \left\lceil \sqrt{E'(e^{\psi} - 1) + \tfrac{1}{2}} \right\rceil \ ,$$

and we know from [ST99b, Theorem 4.3] that

$$|h - E| < L^2 \qquad \text{and} \qquad L = O(q^{\frac{g}{2} - \frac{\lambda+1}{4}}) .$$

Note that for $g \geq 3$ the approximation $E$ can be determined efficiently, in $O(q^\lambda)$ operations. It is described in detail in [SW98] how to evaluate $E$ efficiently if $\lambda = 1$ or $\lambda = 2$.

3.6. **The computation of $R_X$ and $h$.** The complete algorithm for computing $R_X$ and $h$ consists of three steps. First, we compute an approximation $E$ of $h$ such that $|h - E| < L^2$ for some integer $L$, as explained above. In the second step, we compute a multiple $h_0 = h^* R$ of $R_X$ in the interval $]E - L^2, E + L^2[$. This can be done either by Shanks' baby step-giant step method as described in [SW98], or by applying Pollard's kangaroo method to function fields (see below). In the final step, one determines $h^*$ by factoring $h_0$ and using the fact that a prime divisor $r$ of $h_0$ divides $h^*$ if and only if $\mathbf{Nearest}(h_0/r) = \mathcal{O}(X)$. Once having computed $h^*$, one knows $R = h_0/h^*$. In general, we expect $R_X$ to be greater than $2L^2 - 2$, in which case $h_0 = h$ and $h^* = h_X$. If $R \leq 2L^2 - 2$, some additional steps will produce the values of $h$ and $h_X$.

The complexity of this algorithm is mainly determined by the first and the second step. The running time for the approximation is $O(q^\lambda)$. The multiple of the regulator can be computed in $O(L)$ operations by the baby step-giant-step method or in expected running time $O(L)$ by Pollard's kangaroo method. For $g \geq 3$, this gives a total running time of

$$O(q^{\mathrm{round}((2g-1)/5)+\eta}) \qquad , g \geq 3 ,$$

where $\eta = 0$ if $g \equiv 0$ or $g \equiv 3 \pmod 5$, $\eta = 1/4$ if $g \equiv 1 \pmod 5$, $\eta = -1/4$ if $g \equiv 2 \pmod 5$ and $\eta = 1/2$ if $g \equiv 4 \pmod 5$. If $g = 1$ or 2, the total running time is $O(q^{1/4})$ or $O(q^{3/4})$, respectively.

## 4. The Parallelized Kangaroo Method in Real Quadratic Function Fields

In this section we apply the parallelized kangaroo method to compute the the regulator $R_X$, and hence the class numbers $h$ and $h'$, in a real quadratic function field.

Let $k = k(X)(\sqrt{D})$ be a real quadratic function field. We let $E$ and $L$ as in Section 3.5. Then, $h$ lies in the interval $[E - L^2 + 1, E + L^2[$ of length $2L^2 - 1$. This estimate for $h$ and the distance (3.1) represent the necessary ingredients such that the kangaroo method can be applied to compute a multiple of the regulator $R_X$ in expected running time $O(L)$. We explain how this works. We first define $S$, the set of jumps. Here, essential parameters are $l$, the number of elements in $S$, and $\beta$, the mean value of the jump distances. In practice, we will work with $l = 50$, and $\beta$ will be chosen according to Section 2. Given positive integers $\beta$ and $l$, we first generate $l$ positive integers $s_1, s_2, \ldots, s_l$ whose mean value is $\beta$. In fact, since $\mathbf{Nearest}(m) = \mathcal{O}(X)$ for $m = 1, \ldots, g$, we even require that $s_i \geq g+1$, $i = 1, \ldots, l$ to guarantee that $\mathcal{O}(X)$, the neutral element with respect to the giant step operation, is not included in $S$. So let $s_1$ be a randomly chosen integer in $[g+1, 2\beta]$. Then, for $i = 2, \ldots, l-1$, randomly select $s_i$ in the interval $[g+1, 2\beta]$ with the additional property that

$$(i-1)\beta < \sum_{j=1}^{i} s_j \leq (i+1)\beta - (g+1) .$$

Finally, we put

$$s_l = l\beta - \sum_{i=1}^{l-1} s_i \ .$$

Note that $s_l \in [g+1, 2\beta]$ and $\sum_{i=1}^{l} s_i/l = \beta$. We then put

$$\mathfrak{b}_i = \mathbf{Nearest}(s_i + \lfloor g/2 \rfloor) \ , \qquad 1 \le i \le l,$$

and define the set of jumps: $S = \{\mathfrak{b}_1, \mathfrak{b}_2, \ldots, \mathfrak{b}_l\}$. The term $\lfloor g/2 \rfloor$ is added to compensate the headwind (cf. Section 3.4). Having precomputed the reduced principal ideals $\mathfrak{b}_i$ and their distances $\delta(\mathfrak{b}_i)$, we store them as triples $(Q_i, P_i, \delta(\mathfrak{b}_i))$, where $(Q_i, P_i)$ refers to the standard representation of $\mathfrak{b}_i$. $i = 1, \ldots, l$. Next, let $v : \mathcal{R} \to \{1, \ldots, l\}$ be a hash function. For example, if the reduced principal ideal $\mathfrak{a}$ is represented by $(Q, P)$, then we can put $v(\mathfrak{a}) = 1+$ the last coefficient of the polynomial $Q$ modulo $l$.

A kangaroo $Z$ (tame or wild) with starting point $\mathfrak{z}_0$ then follows the path $(\mathfrak{z}_j)$ of reduced principal ideals such that

$$\mathfrak{z}_{j+1} = \mathfrak{z}_j \star \mathfrak{b}_{v(\mathfrak{z}_j)} \ , \qquad j \in \mathbb{N}_0 \ . \tag{4.1}$$

In the beginning, we let $\delta_0(Z) = \delta(\mathfrak{z}_0)$ and, in general, we define $\delta_j(Z) = \delta(\mathfrak{z}_j)$ for $j \ge 1$. By the results mentioned in Section 3.4, we have

$$\delta_{j+1}(Z) = \delta_j(Z) + \delta(\mathfrak{b}_{v(\mathfrak{z}_j)}) + f_{j+1}(Z) \ , \qquad j \in \mathbb{N}_0 \ , \tag{4.2}$$

where $f_{j+1}(Z)$ is a computable integer with $-2g \le f_{j+1}(Z) \le 0$. With that definition, the distance $\delta_j(Z)$ of Section 2 coincides with the distance of Section 3.3 in real quadratic function fields.

Having introduced the notation of kangaroos in real quadratic function fields, we can now follow precisely the lines of Section 2 and define tame and wild kangaroos. We just have to show how to find a multiple of the regulator.

In the original setting of Pollard, we define a tame kangaroo $T$ following the path $(\mathfrak{t}_j)$ with starting point $\mathfrak{t}_0 = \mathbf{Nearest}(E + L^2)$ and a wild kangaroo $W$ travelling along the path $(\mathfrak{w}_j)$ with starting point $\mathfrak{w}_0 = \mathcal{O}(X) = \mathbf{Nearest}(h)$. Both paths are computed using (4.1). We keep track of the distances $\delta_j(T)$ and $\delta_j(W)$ as in (4.2), where $E + L^2 - \delta_0(T) \le g$ and $\delta_0(W) = 0$. The tame kangaroo jumps $M$ steps and then stops at the final resting spot $\mathfrak{t}_M$, where it installs the trap. If it happens that the wild kangaroo $W$ falls into the trap, then there exists an index $N$ such that $\mathfrak{t}_M = \mathfrak{w}_N$. Thus, $\delta_M(T) \equiv \delta_N(W) \pmod{R}$, and $\delta_M(T) - \delta_N(W)$ is a (most likely non-trivial) multiple of the regulator $R_X$. Otherwise, the wild kangaroo is halted after a certain number of steps, and a new wild kangaroo is started at $\mathbf{Nearest}(z)$, with a small integer $z$. As in the general setting, we expect this process to terminate after a total number of $3.28\sqrt{2L^2 - 1}$ jumps, if the mean value $\beta$ of the distances of the ideals in the set of jumps is $(\sqrt{2L^2 - 1})/2$.

The distinguished point method can be employed in an analogous way. We call a reduced principal ideal a *distinguished point* if it satisfies some easily checkable property. For instance, we can define that $\mathfrak{a}$ is a distinguished point if and only if the last coefficient of $N(\mathfrak{a})$ has a specified number, say $F$, of zero bits; then $\Theta = 1/2^F$ represents the approximate proportion of distinguished points. Whenever a kangaroo reaches a distinguished point, we store the ideal together with its distance.

In the parallel setting of van Oorschot and Wiener, we proceed as in Section 2.1. Let $m$ denote an even number of processors, and let $u = v = m/2$. We then define two herds of kangaroos, namely $m/2$ tame kangaroos $T_0, \dots, T_{u-1}$ and $m/2$ wild kangaroos $W_0, \dots, W_{v-1}$. The tame kangaroos are set off around $E$ at starting points $\mathfrak{t}_0(T_k) = \mathbf{Nearest}(E + k\nu)$, $0 \le k < u$, with some small constant $\nu > g+1$. The wild kangaroos start at $\mathfrak{w}_0(W_k) = \mathbf{Nearest}(k\nu)$, $0 \le k < v$. Both herds now follow the paths and keep track of the distances as in (4.1) and (4.2). When it happens that two members of different herds reach the same distinguished point, we can compute a multiple of the regulator from these kangaroos' starting positions and the distances they travelled. This takes an expected number of $2\sqrt{2L^2-1}/m + 1/\Theta$ jumps for each kangaroo.

In the parallel setting of Pollard as outlined in Section 2.2, the kangaroos travel in fixed equivalence classes modulo $uv$ for some coprime integers $u$ and $v$. If we allow the kangaroos to alter the lengths of the jumps as in (4.2), they change equivalence classes modulo $uv$ and Pollard's idea does not work. We resolve this problem by slightly modifying the definition of a jump. For this, we follow the idea of Section 3.3 of identifying positive integers $y$ with tuples $(\mathbf{Nearest}(y), n(y))$, and we let the kangaroos travel on tuples of that form such that for each kangaroo, the sums $\delta(\mathbf{Nearest}(y)) + n(y)$ are invariant modulo $uv$. This works as follows. As before, let $m$ denote the number of processors, and let $u$ and $v$ be coprime integers less than $m$ denoting the numbers of tame and wild kangaroos, respectively. Given a set $\{s_1, s_2, \dots, s_l\}$ of jump distances, we put for $1 \le i \le l$

$$\mathfrak{b}'_i = \mathbf{Nearest}(uvs_i + \lfloor g/2 \rfloor) \quad \text{and} \quad b_i = n(uvs_i + \lfloor g/2 \rfloor) = uvs_i + \lfloor g/2 \rfloor - \delta(\mathfrak{b}'_i)$$

and define the set of jumps by $S' = \{(\mathfrak{b}'_1, b_1), (\mathfrak{b}'_2, b_2), \dots, (\mathfrak{b}'_l, b_l)\}$. Let $v : \mathcal{R} \to \{1, \dots, l\}$ be a hash function. Given a non-negative integer $\mu_0$, we set off a kangaroo $Z'$ at $\mu_0$ by letting its starting point be the tuple $(\mathfrak{z}'_0, z_0)$, where

$$\mathfrak{z}'_0 = \mathbf{Nearest}(\mu_0) \quad \text{and} \quad z_0 = n(\mu_0) = \mu_0 - \delta(\mathfrak{z}'_0) \ .$$

Then $Z'$ follows the path $((\mathfrak{z}'_j, z_j))$ of reduced principal ideals $\mathfrak{z}'_j$ (together with the differences $z_j$) such that for $j \ge 0$

$$\mathfrak{z}'_j = \mathbf{Nearest}(\mu_0 + uv\lambda_j) \quad \text{and} \quad z_j = n(\mu_0 + uv\lambda_j) = \mu_0 + uv\lambda_j - \delta(\mathfrak{z}'_j) \ , \tag{4.3}$$

where $\lambda_0 = 0$ and $\lambda_j = \lambda_{j-1} + s_{v(\mathfrak{z}'_{j-1})}$. If we put $\delta_j(Z') = \delta(\mathfrak{z}'_j)$, then

$$\begin{aligned} \delta_j(Z') + z_j &= \mu_0 + uv\lambda_j \\ &\equiv \mu_0 \qquad \pmod{uv}, \qquad j \ge 0 \ . \end{aligned} \tag{4.4}$$

Given $(\mathfrak{z}'_j, z_j)$ the tuple $(\mathfrak{z}'_{j+1}, z_{j+1})$ can be efficiently computed as follows. First, we perform a giant step operation and determine $\mathfrak{z}_{j+1} = \mathfrak{z}'_j \star \mathfrak{b}'_{v(\mathfrak{z}'_j)}$ such that

$$\begin{aligned} \delta(\mathfrak{z}_{j+1}) &= \delta(\mathfrak{z}'_j) + \delta(\mathfrak{b}'_{v(\mathfrak{z}'_j)}) + f_{j+1}(Z') \\ &= \mu_0 + uv\lambda_{j+1} - z_j + \lfloor g/2 \rfloor - b_{v(\mathfrak{z}'_j)} + f_{j+1}(Z') \ , \end{aligned}$$

where $-2g \le f_{j+1}(Z') \le 0$. Notice that

$$-4g + \lfloor g/2 \rfloor \le \delta(\mathfrak{z}_{j+1}) - (\mu_0 + uv\lambda_{j+1}) \le \lfloor g/2 \rfloor \ .$$

Starting at $\mathfrak{z}_{j+1}$, we then perform at most $4g - \lfloor g/2 \rfloor$ baby steps forwards or $\lfloor g/2 \rfloor$ baby steps backwards to compute $\mathfrak{z}'_{j+1} = \mathbf{Nearest}(\mu_0 + uv\lambda_{j+1})$. Then, we put $z_{j+1} = \mu_0 + uv\lambda_{j+1} - \delta(\mathfrak{z}'_{j+1})$. Notice that with probability $1 - O(1/q)$ we have $z_j = 0 = b_{v(\mathfrak{z}'_j)}$ and $f_{j+1}(Z') = -\lfloor g/2 \rfloor$, in which case $\mathfrak{z}'_{j+1} = \mathfrak{z}_{j+1}$ and a jump is just the same as in the van-Oorschot Wiener parallelization. Only if $\mathfrak{z}'_{j+1} \neq \mathfrak{z}_{j+1}$ we actually obtain a different interpretation of a jump.

Now we define two herds of kangaroos. The tame kangaroos $T_0, \ldots, T_{u-1}$ are set off at $E + iv$ with starting points $(\mathfrak{t}_0(T_i), t_0(T_i)) = (\mathbf{Nearest}(E + iv), n(E + iv))$, $0 \le i < u$. The wild kangaroos $W_0, \ldots, W_{v-1}$ are set off at $ku$ with starting points $(\mathfrak{w}_0(W_k), w_0(W_k)) = (\mathbf{Nearest}(ku), n(ku))$, $0 \le k < v$. Both herds now follow the paths and keep track of the distances as in (4.3) and (4.4). Let $mR_X$ be the multiple of $R_X$ that is closest to $E$ in the interval $[E - L^2 + 1, E + L^2]$. Then the equation

$$E + iv = mR_X + ku \pmod{uv}$$

has a unique solution in $i \pmod u$ and $k \pmod v$. Now recall that $\mathbf{Nearest}(ku) = \mathbf{Nearest}(mR_X + ku)$ and $n(ku) = n(mR_X + ku)$ for $m \in \mathbb{N}$. Therefore, there exist unique integers $i$ and $k$ such that

$$\delta_j(T_i) + t_j(T_i) = \delta_{j'}(W_k) + w_{j'}(W_k) \tag{4.5}$$

for some integers $j$ and $j'$, where

$$\delta_j(T_i) + t_j(T_i) = E + iv + \sigma \ , \qquad \delta_{j'}(W_k) + w_{j'}(W_k) = mR_X + ku + \tau \ ,$$

and $\sigma$ and $\tau$ are multiples of $uv$ that can be evaluated along the paths. The event (4.5) can be detected if $\delta_j(T_i) = \delta_{j'}(W_k)$, which is the case with probability $1 - O(1/q)$, since then the corresponding reduced principal ideals $\mathfrak{t}_j(T_i)$ and $\mathfrak{w}_{j'}(W_k)$ coincide *and* the kangaroos $T_i$ and $W_k$ travel on the same paths towards the next distinguished point. It follows immediately that

$$mR_X = E + iv + \sigma - (ku + \tau)$$

(or, if the travel continues after (4.5) until the next distinguished point, with correspondingly modified $\sigma'$ and $\tau'$), so that we have found a multiple of the regulator. This takes an expected number of approximately $\sqrt{(2L^2 - 1)/(uv)} + 1/\Theta$ jumps on each processor; in the very unlikely event that (4.5) is not detected because $\mathfrak{t}_j(T_i) \neq \mathfrak{w}_{j'}(W_k)$ or $\mathfrak{t}_j(T_i) = \mathfrak{w}_{j'}(W_k)$ but $t_j(T_i) \neq w_{j'}(W_k)$ (where the latter may cause that (4.5) does not survive until the next distinguished point), the algorithm continues until the next event of the form (4.5).

4.1. **Improvements.** There are two ways to improve the kangaroo method in function fields. Firstly, the above complexity analysis and parameter choice were made under the assumption that $h$ was distributed at random in the interval $[E - L^2 + 1, E + L^2[$. But this is not the case for function fields. In fact, results in [ST99b, Section 6] imply that for large values of $q$, the expected mean value of $|h - E|/L^2$ is a real number $\alpha(g)$ much smaller than $1/2$. For instance, $\alpha(3) \approx 0.1634$ and $\alpha(4) \approx 0.1253$. If we start the tame kangaroos at around $E$ and the wild kangaroos at around $h$, then the expected initial difference between the two herds of kangaroos is $\alpha(g) \cdot L^2$ (and not $L^2/2$). With this knowledge we can improve the choice of the set of jumps as discussed in Section 2.1. (See also below.)

Secondly, we make use of the different running times for baby steps and giant steps in real quadratic function fields to obtain a speed-up of size $\Theta(\sqrt{n})$, where

$$n = \text{running time for one giant step/running time for one baby step} \ . \tag{4.6}$$

Because of the running times mentioned in Sections 3.3 and 3.4, we expect that $n \approx 4g$ for large values of $q$.

The idea is to build "classes" with respect to the baby steps, since one giant step roughly corresponds to $n$ baby steps. Here, we consider the van Oorschot and Wiener setting. Let $m$ denote an even number of processors and let $\beta$ denote the mean value for the distances of the ideals in the set of jumps $S = \{\mathfrak{b}_1, \mathfrak{b}_2, \dots, \mathfrak{b}_l\}$.

Let $\mathcal{R}$ denote the set of reduced principal ideals. Any $\mathfrak{a} \in \mathcal{R}$ is given as $\mathfrak{a} = (Q, P)$, where $Q, P \in k[X]$ such that $Q|(D - P^2)$, $\deg(P) \leq \deg(Q)$, and $Q$ is monic. Let $\mathrm{last}(Q)$ denote the lowest coefficient of $Q$. For a positive integer $\rho$ we define $\mathcal{Q}_\rho \subset \mathcal{R}$ by

$$\mathcal{Q}_\rho = \{\mathfrak{a} = (Q, P) \ ; \ \mathrm{last}(Q) = 0 \pmod{\rho}\} \ ,$$

and we let the kangaroos operate on $Q_\rho$. Define the map $\Psi : \mathcal{R} \to \mathcal{Q}_\rho$ by

$$\Psi(\mathfrak{c}) = \text{the first } \mathfrak{a} \in \mathcal{Q}_\rho \text{ that is reached from } \mathfrak{c} \text{ via baby steps} \ .$$

By considering the pre-images of $\Psi$, we have that $\Psi$ divides $\mathcal{R}$ into approximately $|\mathcal{R}|/\rho$ equivalence classes of size $\rho$ each. Furthermore, the expected running time to evaluate $\Psi$ for a given $\mathfrak{c} \in \mathcal{R}$ is $(\rho - 1)$ baby steps. Let $v : \mathcal{Q}_\rho \to \{1, \dots, l\}$ be a hash function and $f : \mathcal{Q}_\rho \to \mathcal{R}$ given by

$$f(\mathfrak{a}) = \mathfrak{a} \star \mathfrak{b}_{v(\alpha)} \ .$$

Then one jump of a kangaroo consists of computing $\Psi(f(\mathfrak{a}))$. That is, we define the jumps on the set $\mathcal{Q}_\rho$ rather than on $\mathcal{R}$. Observe that the expected cost of one jump is one giant step plus $(\rho - 1)$ baby steps, hence

$$1 + (\rho - 1)/n \qquad \text{giant steps} \ .$$

We now determine the optimal choices for $\beta$ and $\rho$, given that we know $m$, $\alpha(g)$, $n$, and $L$. Our analysis is a generalization of the generic case; if baby steps and giant steps have the same complexity, then $n = \rho = 1$. As in Section 2.1, let $E(d)$ denote the expected initial distance between the herds of tame and wild kangaroos. Then $E(d) = E(|h - E|) = \alpha(g)L^2$. Now we denote by $t$ the expected running time for one kangaroo when the set of jumps is chosen such that the mean value $\beta$ of the distances is $\beta \approx \sqrt{2L^2 - 1}m/4$, i.e., under the assumption that the solution is uniformly distributed in the interval $]E - L^2, E + L^2[$. Then

$$t = E(d)/\beta + (2/m)^2\beta + 1/\Theta = (2\alpha(g) + 1)\sqrt{2L^2 - 1}/m + 1/\Theta + O(1)$$

giant steps. In the following, we neglect the term $1/\Theta$, assuming that it is small compared to the total running time. Let $t_g = t_g(\beta)$ be the expected running time for one kangaroo when choosing the jump distances according to the heuristics in [ST99b], and let $t_{g,n} = t_{g,n}(\beta, \rho)$ be the expected running time for one kangaroo when using the method with baby steps as explained above and the heuristics.

With the expected initial distance between the two herds being $\alpha(g)L^2$, it takes an expected number of $\alpha(g)L^2/\beta$ jumps to cover this initial distance. Now we assume that for each jump (after the kangaroo that travels behind has covered the initial distance), the probability for a hit is, on average,

$$\rho/\beta$$

($\rho < \beta$). Then it takes an expected number of $4\beta/(\rho m^2)$ jumps for the $m/2$ kangaroos following behind to produce a hit on a spot previously occupied by one of the $m/2$ leading kangaroos. Hence, the expected total running time per kangaroo is

$$\alpha(g)L^2/\beta + 4\beta/(\rho m^2) \tag{4.7}$$

jumps. Thus, the expected running time per kangaroo totals

$$t_{g,n}(\beta,\rho) = \left(\alpha(g)L^2/\beta + 4\beta/(\rho m^2)\right)(1 + (\rho-1)/n)$$

giant steps. Taking the first derivative with respect to $\beta$ yields that $t_{g,n}(\beta,\rho)$ becomes minimal for

$$\beta = mL\,\frac{\sqrt{\alpha(g)}\,\sqrt{\rho}}{2}\ .$$

Plugging this into $t_{g,n}(\beta,\rho)$ and taking the first derivative with respect to $\rho$ yields that $t_{g,n}(\beta,\rho)$ becomes minimal if

$$\rho = \frac{\sqrt{n-1}}{\sqrt{\alpha(g)}}\,\frac{2\beta}{mL} = \sqrt{n-1}\,\sqrt{\rho}\ ,$$

i.e. $\rho = n-1$, and

$$\beta = \frac{m\,L}{2}\,\sqrt{\alpha(g)}\,\sqrt{n-1}\ . \tag{4.8}$$

This gives a total expected running time of

$$t_{g,n} = \frac{8\sqrt{n-1}\,\sqrt{\alpha(g)}}{n}\,\frac{L}{m}$$

giant steps for each kangaroo. If we do not use baby steps, then we have $n = \rho = 1$ and the optimal choice of $\beta$ would be

$$\beta = \frac{m\,L}{2}\,\sqrt{\alpha(g)} \tag{4.9}$$

(in accordance with (2.6)), yielding a total expected running time of

$$t_g = 4\sqrt{\alpha(g)}\,\frac{L}{m}$$

giant steps for each kangaroo. Thus, using baby-step equivalence classes gives a speed-up of

$$\frac{t_g}{t_{g,n}} = \frac{4\sqrt{\alpha(g)}\,n}{8\sqrt{n-1}\,\sqrt{\alpha(g)}} = \frac{n}{2\sqrt{n-1}} \sim \sqrt{g}$$

for large values of $q$, where we use that $n$ is approximately $4g$. Also notice that we obtain a speed-up of the original method without heuristics and baby steps by a factor of about

$$\frac{t}{t_{g,n}} = \frac{(2\alpha(g)+1)\,n\,\sqrt{2L^2-1}}{8\sqrt{n-1}\,\sqrt{\alpha(g)}\,L} \sim \frac{(2\alpha(g)+1)\sqrt{2}\,\sqrt{n}}{8\sqrt{\alpha(g)}} \sim \frac{\sqrt{g}(2\alpha(g)+1)}{2\sqrt{2}\,\sqrt{\alpha(g)}}\ .$$

For example, for $g = 3$ and $\alpha(g) \approx 0.1634$, this gives a speed-up factor of about 2, while for $g = 4$ and $\alpha(g) \approx 0.1254$, the speed-up factor is about 2.5. By exploiting only the heuristics but not using the baby-step equivalence classes, we still can speed up the original van Oorschot and Wiener parallelization by the factor $t/t_g = (2\alpha(g)+1)/(2\sqrt{2\alpha(g)})$.

4.2. **Experimental Results.** For our computations we used several Suns and SGIs, and the Computer Algebra System SIMATH [Zim97]. The reference machine was a Sun Ultra Enterprise 450 under Solaris 2.6. We computed the regulator $R_X$ and the class numbers $h$, $h_X$ of a real quadratic function field $K = k(X)(\sqrt{D})$ over $k = \mathbb{F}_q$, where $q$ is an odd prime. The discriminants $D \in k[X]$ were random monic squarefree polynomials of even degree. Hereby, we mainly used irreducible polynomials so that we could not use 2-parts of the ideal class number $h_X$.

We implemented the parallelized Pollard kangaroo method successfully in the variant of van Oorschot and Wiener as described in Section 4 together with the aforementioned improvements. We now provide examples for 2 and 16 kangaroos. We used a set of jumps $S$ with $l = 50$ elements; we believe that arguments similar to the ones given in [Tes98] will show that a choice of $l \geq 20$ elements is sufficient. We define a distinguished point to be a reduced ideal $\mathfrak{a} = (Q, P) \in \mathcal{Q}_\rho$ for which the lowest $F$ bits of $\text{last}(Q)/\rho$ are 0 (In the non-baby step setting we let $\rho = 1$.). For example, we can choose $F = \lfloor \log_2 L \rfloor /2$ such that $\Theta = 1/2^F \approx 1/\sqrt{L}$, and the expected number of distinguished points to be stored is $2\Theta\sqrt{2L^2} \approx 2\sqrt{2L}$. Note that this is only about the square root of the space requirement of the baby step-giant step algorithm [SW98].

In the first application, we simulated two parallel processors by letting the kangaroos jump alternately. There were only two kangaroos, a tame kangaroo $T$ with starting point $\mathfrak{t}_0(T) =$ **Nearest**$(E)$ and a wild kangaroo $W$ with starting point $\mathfrak{w}_0(W) = \mathcal{O}(X)$. Our aim was to compare in practice the (average) running time of the method using baby steps with the (average) running time of the original method, with varying mean values for the jump distances. We concentrated on the case that $g = 3$, which means $\deg(D) = 8$, and $q = 10009$. We used such a small value of $q$ because we needed to work with a large sample space in order to produce meaningful averages.

In Table 4, we list examples for the original method with different mean values $\beta$ for the set of jumps. We worked with $\Theta = 2^{-4}$, so that about every 16-th ideal encountered was stored as distinguished point. We randomly generated 1000 examples, with

$$\beta = \frac{\gamma}{10} L ,$$

where $\gamma$ is an integer between 2 and 23. Recall that for the original method the theoretically optimal $\beta$ is $\beta = L/\sqrt{2}$ which means that $\gamma$ should be around 7. When taking into account the heuristics [ST99b], we expect that (using (4.9)) $\gamma = 10\sqrt{\alpha(3)} \approx 4$ would be optimal. (Recall that $\alpha(3) \approx 0.1634$.) We then expect a speed-up of $t/t_3 = (2\cdot 0.1634+1)/(2\sqrt{2 \cdot 0.1634}) \approx 1.16$. Indeed, on dividing the time for $\gamma = 7$ by the time for $\gamma = 4$ we find a speed-up by a factor 1.18.

In Table 5, we list examples for the improved method (i.e., using baby-step classes) with different mean values $\beta$ for the set of jumps. We worked with $\Theta = 2^{-4}$. As for the parameter for the baby-step classes, we worked with $r = 9$. (This choice for $r$ is taking into account

| $\gamma$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| | $7.73s$ | $6.42s$ | $6.23s$ | $6.16s$ | $6.86s$ | $7.35s$ | $7.92s$ | $8.64s$ |

| $\gamma$ | 10 | 11 | 12 | 13 | 14 | 16 | 20 | 23 |
|---|---|---|---|---|---|---|---|---|
| | $8.81s$ | $9.73s$ | $10.16s$ | $10.85s$ | $11.46s$ | $13.31s$ | $16.41s$ | $18.55s$ |

TABLE 4. Average running times for regulator computation (no use of baby steps) for $g = 3$ and $q = 10009$.

| $\gamma$ | 3 | 4 | 6 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|
| | $9.97s$ | $7.70s$ | $5.38s$ | $4.45s$ | $4.27s$ | $4.03s$ | $3.83s$ | $3.73s$ | $3.67s$ | $3.49s$ |

| $\gamma$ | 15 | 16 | 17 | 18 | 20 | 22 | 24 | 26 | 28 | 30 |
|---|---|---|---|---|---|---|---|---|---|---|
| | $3.52s$ | $3.57s$ | $3.58s$ | $3.51s$ | $3.60s$ | $3.74s$ | $3.88s$ | $4.04s$ | $4.17s$ | $4.26s$ |

TABLE 5. Average running time for regulator computation (method using baby steps) for $g = 3$ and $q = 10009$.

that for small values of $q$, the ratio (4.6) is smaller than $4g$. See also Table 8 in [Ste99].) Again, we randomly generated 1000 examples, with $\beta = \gamma L/10$, where now $\gamma$ is an integer between 3 and 30. When applying the method with baby-step classes we expect that (using (4.8) with $n = 10$) $\gamma = 10\sqrt{\alpha(3)}\sqrt{9}$, i.e. $\gamma = 12$, yields optimal performance. We then expect that the speed-up factor of the new method compared to the method with heuristics is $t_3/t_{3,10} = 1.5$. Again, this is reflected by our experimental results: on comparing the entries for $\gamma = 3, 4$ in Table 4 with the corresponding entries for $\gamma = 12$ we find a speed-up by a factor 1.7. Notice that for larger values of $q$ and $g = 3$, when $n \approx 12$, the expected speed-up goes up to $t_3/t_{3,n} = \sqrt{3} \approx 1.73$.

Finally, we give experimental data about two of our largest regulator computations. In both examples, we worked with 8 tame and 8 wild kangaroos. We used $m = 16$ processors, including SGI Challenge Workstations running IRIX V6.5 and SunUltra Enterprise 450 machines under Solaris 2.6. Distinguished points were collected on the individual machines, and then manually transferred to a central file for comparison.

In the first example, we took $k = \mathbb{F}_q$ with $q = 1000000097$ (10 digits) and

$$\begin{aligned} D(X) =& X^8 + 438841888X^7 + 582096993X^6 + 508378916X^5 + 745512041X^4 \\ &+ 485998124X^3 + 614914799X^2 + 46942763X + 162950126 \ , \end{aligned}$$

a prime polynomial modulo $q$. We computed $E = 1000037518200164985425069904$ and $L = 1870915821$, and used the kangaroo method to find the divisor class number $h$ in the interval $]E-L^2, E+L^2[$. For this, we used the variant with baby-step classes, with parameter

$\rho = 10$. We worked with a set $S$ of $l = 50$ jumps for which the mean value $\beta$ of the jump distances was $16 \cdot L$. (This choice of $\beta$ is slightly different from the theoretically optimal choice (4.8), which would be $10.2 \cdot L$. Our choice corresponds to $\gamma = 10\beta/(Lm/2) = 20$ of Table 5 – while $\beta = 10.2L$ corresponds to $\gamma = 12$ – , which, in earlier versions of Table 5, produced the best average running times in practice.) We defined the jumps only on ideals represented by $(Q, P)$, such that $\text{last}(Q) \equiv 0 \pmod{10}$. Such an ideal was a distinguished point if, in addition, the last 19 bits of $\text{last}(Q)/10$ were zero. The 8 tame kangaroos were set off at the ideals closest to $E + 10009i$, $i = 0, \ldots, 7$, and the 8 wild kangaroos were set off at the ideals closest to $10009i$, $i = 0, \ldots, 7$. Altogether, 1908 distinguished points were found and centrally stored until, after 19 hours of (parallel) computing time, two matching points had been discovered, where one stemmed from a wild and one from a tame kangaroo. (Before that, we had observed one useless collision between two wild kangaroos.) This match allowed us to compute

$$h = 1000037518790912387296311384 \ ,$$

a divisor class number of 28 digits. For the regulator, we found that $R_X = h$. Notice that $|h - E|/L^2 = 0.168769\ldots$. Therefore, using (4.7), we find that the theoretically expected number of jumps for the 16 kangaroos altogether is 1064118920. On the other hand, since it takes on average $2^{19}$ jumps to find a distinguished point, we conclude that the total number of jumps performed by all kangaroos was about 1000341504, and this agrees with our predictions.

To estimate the running time for this computation had it been done on a single SunUltra Enterprise 450, we observe that it took, on average, 10 1/4 minutes on such a machine to find a distinguished point. Thus, the estimated total running time is 19557 minutes, or 13 days and 14 hours.

For our second example we chose $k = \mathbb{F}_q$ with $q = 2155000013$ and

$$\begin{aligned} D(X) = &X^8 + 583595124X^7 + 1467293974X^6 + 670334099X^5 + 1512249128X^4 \\ &+ 146202392X^3 + 1574723323X^2 + 434418859X + 1335383000 \ . \end{aligned}$$

Here, $q^3 = 1.000787\ldots \cdot 10^{28}$. Moreover, we evaluated $E = 1000814641563842458722346704$ and $L = 4031767232$, so that $E - L^2 > 1.0008\ldots \cdot 10^{28}$. This means that the real quadratic function field $k(X)\sqrt{D}$ must have a divisor class number $h$ with 29 digits, whence the choice of $q$ and $D$. Again, we used the variant with baby-steps classes, with parameter $\rho = 10$. This time we worked with 50 elements in the set of jumps whose jump distances had the mean value $0.85 \cdot 16 \cdot L$. The kangaroos operated on ideals such that $\text{last}(Q) \equiv 0 \pmod{10}$. For an ideal to be distinguished we requested, in addition, that the last 20 bits of $\text{last}(Q)/10$ be zero. The tame kangaroos were set off at the ideals closest to $E + 100003i$, $i = 0, \ldots, 7$, and the 8 wild kangaroos were set off at the ideals closest to $100003i$, $i = 0, \ldots, 7$. Altogether 1101 distinguished points have been collected until a useful collision was detected after almost 110 hours. (As before, we also observed one useless collision, again between two wild kangaroos.) We then computed

$$h = R_X = 1000814641788539503344512486 8 \ .$$

Thus, $|h - E|/L^2 = 0.138231\ldots$, so that the expected number of jumps of all 16 kangaroos is 2026461075. Considering that 1101 distinguished points were found and it took on average

$2^{20}$ jumps to find one such point, we estimate that altogether about 1154482176 jumps have been performed. (In fact, an explicit count of the jumps gave the value 1150422076.) This is much less than we would expect, which is most likely due to the fact that in the second stage of the algorithm, i.e., when both herds travel in the same region, the algorithm experiences performance variances as they are typical for birthday paradoxon algorithms (cf. [Tes00] for some details).

Since, on average, it took 1 hour and 12 1/4 minutes to find a distinguished point on a SunUltra Enterprise 450 running Solaris 2.6, we estimate that the whole computation would have taken altogether about 79547 hours, or 55 days and 6 hours, on a single such machine. For genus $g = 3$, this is the largest such class number and regulator computation ever reported.

## 5. Outlook

Pollard's kangaroo method also applies to *imaginary* quadratic function fields, which we consider to be quadratic function fields $k(X)(\sqrt{D(X)})$ where $D$ is a monic, squarefree polynomial of degree $2g+1$. (In the case that $D(X)$ is a squarefree polynomial of degree $2g+2$ and the leading coefficient is a non-square in $k$, a constant field extension of degree 2 will lead to a real quadratic function field.) In the imaginary case, the corresponding operation on reduced ideals as described in Section 3.4 is a group operation (see [Can87, PR99, Ste99]) so that the parallelized kangaroo methods as suggested by van Oorschot and Wiener and by Pollard immediately generalize to imaginary quadratic function fields (see also [GH]). However, since one only has one operation in the imaginary case, baby steps and giant steps have the same complexity and the idea of speeding up with baby steps does not work.

In Section 4, we only considered the computation of the regulator and the class number, since we could determine an interval for the divisor class number. We now define the *discrete logarithm problem for real quadratic function fields* as follows: For any $\mathfrak{a} \in \mathcal{R}$, find $\delta(\mathfrak{a})$, $0 \le \delta(\mathfrak{a}) < R$. Then, of course, Pollard's kangaroo method also applies to solving the discrete logarithm problem in real quadratic function fields, if one knows that the discrete logarithm lies in a given interval.

A generalization of Pollard's *rho* method and its parallelized versions (see [Pol78, Tes98, vOW99, Pol]) to function fields can be employed as well by applying similar ideas as in Section 4. This method is preferable if one has no additional information on the size of the regulator or the class numbers.

**Acknowledgments:** We are grateful to John M. Pollard for fruitful comments on [ST99a].

## References

[Art24] E. Artin. Quadratische Körper im Gebiete der höheren Kongruenzen I, II. *Math. Zeitschr.*, 19:153–206, 1924.

[Can87] D. G. Cantor. Computing in the jacobian of a hyperelliptic curve. *Mathematics of Computation*, 48:95–101, 1987.

[GH] P. Gaudry and R. Harley. Counting points on hyperelliptic curves over finite fields. To appear in the Proceedings of ANTS IV, LNCS, Springer.

[LiD97] LiDIA Group, Technische Universität Darmstadt, Darmstadt, Germany. *LiDIA - A library for computational number theory, Version 1.3*, 1997.

[Pol] J. M. Pollard. Kangaroos, Monopoly and discrete logarithms. To appear in *Journal of Cryptology*.

[Pol78] J. M. Pollard. Monte Carlo methods for index computation (mod $p$). *Mathematics of Computation*, 32(143):918–924, 1978.

[PR99] S. Paulus and H.-G. Rück. Real and imaginary quadratic representations of hyperelliptic function fields. *Mathematics of Computation*, 68:1233–1241, 1999.

[Sha71] D. Shanks. Class number, a theory of factorization and genera. In *Proc. Symp. Pure Math. 20*, pages 415–440. AMS, Providence, R.I., 1971.

[Sha72] D. Shanks. The infrastructure of a real quadratic field and its applications. In *Proc. 1972 Number Th. Conf., Boulder, Col.*, pages 217–224, 1972.

[SSW96] R. Scheidler, A. Stein, and H. C. Williams. Key-exchange in real quadratic congruence function fields. *Designs, Codes and Cryptography*, 7:153–174, 1996.

[ST99a] A. Stein and E. Teske. Catching kangaroos in function fields. Technical Report CORR 99-09, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, 1999. 19 pages.

[ST99b] A. Stein and E. Teske. Explicit bounds and heuristics on class numbers in hyperelliptic function fields. Technical Report CORR 99-26, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, 1999. 30 pages.

[Ste99] A. Stein. Sharp upper bounds for arithmetics in hyperelliptic function fields. Technical Report CORR 99-23, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, 1999. 68 pages.

[Sti93] H. Stichtenoth. *Algebraic Function Fields and Codes*. Springer, Berlin, 1993.

[SW98] A. Stein and H. C. Williams. An improved method of computing the regulator of a real quadratic function field. In *Algorithmic Number Theory Seminar ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 607–620. Springer, 1998.

[SZ91] A. Stein and H. G. Zimmer. An algorithm for determining the regulator and the fundamental unit of a hyperelliptic congruence function field. In *Proc. 1991 Int. Symp. on Symbolic and Algebraic Computation, ISAAC, Bonn, July 15-17*, pages 183–184. ACM Press, 1991.

[Tes98] E. Teske. Speeding up Pollard's rho method for computing discrete logarithms. In *Algorithmic Number Theory Seminar ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 541–554. Springer, 1998.

[Tes00] E. Teske. On random walks for Pollard's rho method. *Mathematics of Computation*, posted on February 18, 2000, PII: S 0025-5718(00)01213-8 (to appear in print), 2000.

[vOW99] P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12:1–28, 1999.

[Zim97] H. G. Zimmer *et al.* Simath manual, 1997. Universität des Saarlandes, Saarbrücken, Germany.

University of Waterloo, Department of Combinatorics and Optimization, Waterloo, Ontario, Canada N2L 3G1

*E-mail address*: {astein,eteske}@cacr.math.uwaterloo.ca