Differential addition chains

Daniel J. Bernstein [⋆]

`djb@cr.yp.to`

Abstract. Differential addition chains (also known as strong addition chains, Lucas chains, and Chebyshev chains) are addition chains in which every sum is already accompanied by a difference. Low-cost differential addition chains are used to efficiently exponentiate in groups where the operation $a, b, a/b \mapsto ab$ is fast: in particular, to perform x-coordinate scalar multiplication $P \mapsto mP$ on an elliptic curve $y^2 = x^3 + Ax^2 + x$. Similarly, low-cost *two-dimensional* differential addition chains are used to efficiently compute the function $P, Q, P - Q \mapsto mP + nQ$ on an elliptic curve. This paper presents two new constructive upper bounds on the costs of two-dimensional differential addition chains. The paper's new "binary" chain is very easy to compute and uses 3 additions (14 field multiplications in the elliptic-curve context) per exponent bit, with a uniform structure that helps protect against side-channel attacks. The paper's new "extended-gcd" chain takes more time to compute, does not have the uniform structure, and is not easy to analyze, but experiments show that it takes only about 1.77 additions (9.97 field multiplications) per exponent bit.

1 What is a differential addition chain?

A **differential addition chain** is an addition chain in which each sum is already accompanied by a difference: i.e., whenever a new chain element $P + Q$ is formed by adding P and Q, the difference $P - Q$ was already in the chain. Here is an example of a one-dimensional differential addition chain starting from 0 and 1:

$$\begin{array}{rcl}
0 & & \\
1 & & \\
2 = 1 + 1 & \text{with difference} & 1 - 1 = 0 \\
3 = 2 + 1 & \text{with difference} & 2 - 1 = 1 \\
4 = 2 + 2 & \text{with difference} & 2 - 2 = 0 \\
7 = 4 + 3 & \text{with difference} & 4 - 3 = 1 \\
11 = 7 + 4 & \text{with difference} & 7 - 4 = 3 \\
18 = 11 + 7 & \text{with difference} & 11 - 7 = 4 \\
29 = 18 + 11 & \text{with difference} & 18 - 11 = 7 \\
40 = 29 + 11 & \text{with difference} & 29 - 11 = 18 \\
51 = 40 + 11 & \text{with difference} & 40 - 11 = 29 \\
91 = 51 + 40 & \text{with difference} & 51 - 40 = 11
\end{array}$$

[⋆] This work was carried out while the author was visiting Danmarks Tekniske Universitet. Date of this document: 2006.02.19. Permanent ID of this document: `9620b81ea01f66b2a782be234dade959`.

Here is an example of a two-dimensional differential addition chain starting from $(0,0)$, $(1,0)$, $(0,1)$, and $(1,-1)$:

$(0,0)$						
$(1,0)$						
$(0,1)$						
$(1,-1)$						
$(1,1)$	$=$	$(1,0)$	$+$	$(0,1)$	with	$(1,-1)$
$(1,2)$	$=$	$(1,1)$	$+$	$(0,1)$	with	$(1,0)$
$(1,3)$	$=$	$(1,2)$	$+$	$(0,1)$	with	$(1,1)$
$(2,5)$	$=$	$(1,3)$	$+$	$(1,2)$	with	$(0,1)$
$(3,8)$	$=$	$(2,5)$	$+$	$(1,3)$	with	$(1,2)$
$(5,13)$	$=$	$(3,8)$	$+$	$(2,5)$	with	$(1,3)$
$(7,18)$	$=$	$(5,13)$	$+$	$(2,5)$	with	$(3,8)$
$(12,31)$	$=$	$(7,18)$	$+$	$(5,13)$	with	$(2,5)$
$(19,49)$	$=$	$(12,31)$	$+$	$(7,18)$	with	$(5,13)$
$(26,67)$	$=$	$(19,49)$	$+$	$(7,18)$	with	$(12,31)$
$(33,85)$	$=$	$(26,67)$	$+$	$(7,18)$	with	$(19,49)$
$(40,103)$	$=$	$(33,85)$	$+$	$(7,18)$	with	$(26,67)$
$(47,121)$	$=$	$(40,103)$	$+$	$(7,18)$	with	$(33,85)$
$(54,139)$	$=$	$(47,121)$	$+$	$(7,18)$	with	$(40,103)$
$(94,242)$	$=$	$(47,121)$	$+$	$(47,121)$	with	$(0,0)$
$(141,363)$	$=$	$(94,242)$	$+$	$(47,121)$	with	$(47,121)$
$(148,381)$	$=$	$(94,242)$	$+$	$(54,139)$	with	$(40,103)$
$(289,744)$	$=$	$(148,381)$	$+$	$(141,363)$	with	$(7,18)$
$(296,762)$	$=$	$(148,381)$	$+$	$(148,381)$	with	$(0,0)$
$(585,1506)$	$=$	$(296,762)$	$+$	$(289,744)$	with	$(7,18)$
$(874,2250)$	$=$	$(585,1506)$	$+$	$(289,744)$	with	$(296,762)$
$(1459,3756)$	$=$	$(874,2250)$	$+$	$(585,1506)$	with	$(289,744)$
$(2333,6006)$	$=$	$(1459,3756)$	$+$	$(874,2250)$	with	$(585,1506)$
$(2918,7512)$	$=$	$(1459,3756)$	$+$	$(1459,3756)$	with	$(0,0)$
$(5251,13518)$	$=$	$(2918,7512)$	$+$	$(2333,6006)$	with	$(585,1506)$
$(8169,21030)$	$=$	$(5251,13518)$	$+$	$(2918,7512)$	with	$(2333,6006)$
$(10502,27036)$	$=$	$(5251,13518)$	$+$	$(5251,13518)$	with	$(0,0)$
$(18671,48066)$	$=$	$(10502,27036)$	$+$	$(8169,21030)$	with	$(2333,6006)$

More generally, a **differential addition-subtraction chain** is an addition-subtraction chain where each sum is already accompanied by a difference and each difference is already accompanied by a sum. A typical example starts from $(0,0)$ $(1,0)$ $(0,1)$ $(1,-1)$ and reaches $(26967,48215)$ via $(1,1)$ $(1,2)$ $(2,3)$ $(3,5)$ $(4,7)$ $(5,9)$ $(9,16)$ $(14,25)$ $(19,34)$ $(33,59)$ $(38,68)$ $(66,118)$ $(71,127)$ $(61,109)$ $(132,236)$ $(203,363)$ $(264,472)$ $(325,581)$ $(528,944)$ $(731,1307)$ $(1259,2251)$ $(1787,3195)$ $(2518,4502)$ $(3249,5809)$ $(5036,9004)$ $(6823,12199)$ $(10072,18008)$ $(16895,30207)$ $(26967,48215)$; here $(61,109)$ is computed as $(66,118)-(5,9)$, with the sum $(66,118)+(5,9)=(71,127)$ already in the chain.

Notes on terminology

Let C be a differential addition chain that starts from $0, 1$, and let C' be the same addition chain without the initial 0. [10] calls C a "Lucas chain" if C strictly increases. A "STRONGCHAIN" program published online by Knuth refers to C' as "a strong addition chain—aka a Lucas chain or a Chebyshev chain" if C strictly increases. [12] calls C' a "Lucas chain" whether or not C increases. [8] calls C' a "Lucas chain" (and any positive integer multiple of C' a "Lucas prechain") if C strictly increases. [11, Definition 3.1] calls C a "strong addition chain" whether or not it increases. [11, Definition 3.2] uses "Lucas chain" for any differential addition-subtraction chain that starts from $0, 1$. [11, Definition 3.5] uses "vectorial Lucas chain" for any differential addition-subtraction chain that starts from $(0, 0), (1, 0), (0, 1), (1, -1)$.

There are several reasons that I'm avoiding the "Lucas chain" terminology. I want to talk about addition chains and about addition-subtraction chains; "Lucas chains" allow subtractions in [11] but not in [10], [12], or [8]. Furthermore, many new readers will confuse "Lucas chains" with Lucas sequences, which are a quite different concept. Furthermore, I see no evidence that Lucas deserves any credit for the concept of a "Lucas chain." Furthermore, the terminology gives no hint that a "Lucas chain" is an addition chain satisfying an extra condition. "Strong addition chain" avoids this problem, but "differential addition chain" does a better job of helping the reader remember what the condition is.

2 What are the contributions of this paper?

The point of this paper is to present two new constructive upper bounds on the costs of two-dimensional differential addition-subtraction chains. Assume, for example, that m and n are 256-bit integers. The new "binary" chain for (m, n) has the following features:

- It starts from $(0, 0)$, $(1, 0)$, $(0, 1)$, and $(1, -1)$.
- It has 768 additions (and no subtractions).
- For each addition of P and Q, the difference $P - Q$ is either $(0, 0)$ or $(1, 0)$ or $(0, 1)$ or $(1, 1)$ or $(1, -1)$.
- 256 of the additions are doublings, i.e., have difference $(0, 0)$. The doublings appear in a uniform pattern: add, double, add; add, double, add; etc.

The new "extended-gcd" chain for (m, n) has the following features:

- It starts from $(0, 0)$, $(1, 0)$, $(0, 1)$, and $(1, -1)$.
- It has approximately 455.0 (standard deviation 5.9) additions/subtractions for an average coprime pair (m, n). This figure comes from experiments with 1000 pairs; the actual average is almost certainly in $[454, 456]$.
- Out of the additions/subtractions, approximately 88.8 on average (standard deviation 10.5) are doublings.

The extended-gcd chain has several parameters that can be tweaked. Tweaking parameters can replace $(455.0, 88.8)$ by $(453.1, 77.5)$, for example.

Applications

See [11] for a survey of applications and closely related problems such as XTR exponentiation. I will focus on the application I find most interesting, namely computing $mP + nQ$ where P and Q are points on an elliptic curve.

Consider an elliptic curve of the form $y^2 = x^3+Ax^2+x$. If P and Q are points on the curve then one can efficiently compute the x-coordinate of $P+Q$ from the x-coordinates of P, Q, and $P - Q$. Specifically, if x-coordinates are represented as fractions, then the computation of $P+Q$ takes just 6 field multiplications, as pointed out by Montgomery in [9].

By iterating this computation one can reach any combination $mP + nQ$ starting from $P, Q, P - Q$. This takes just $6k$ field multiplications if (m, n) has a differential addition chain with k additions starting from $(0, 0)$, $(1, 0)$, $(0, 1)$, $(1, -1)$. Consider, for example, the chain $(0, 0)$, $(1, 0)$, $(0, 1)$, $(1, -1)$, $(1, 1)$, $(1, 2)$, $(2, 3)$, $(3, 5)$. One can efficiently compute

- the x-coordinate of $P + Q$ from the x-coordinates of $P, Q, P - Q$; and then
- the x-coordinate of $P + 2Q$ from the x-coordinates of $P + Q, Q, P$; and then
- the x-coordinate of $2P + 3Q$ from the x-coordinates of $P + 2Q, P + Q, Q$; and then
- the x-coordinate of $3P+5Q$ from the x-coordinates of $2P+3Q, P+2Q, P+Q$.

This uses 24 field multiplications. Differential addition-subtraction chains can be used in the same way.

Some additions in this context take fewer than 6 field multiplications, as pointed out by Montgomery:

- Computing the x-coordinate of $P+Q$ from the x-coordinates of $P, Q, P-Q$ takes only 5 field multiplications if the x-coordinate of $P-Q$ has denominator 1. This observation motivates considering differential addition chains with a limited set of differences $P - Q$, allowing the denominators to be replaced by 1 at the cost of a limited number of field divisions.
- A doubling $P \mapsto P+P$ takes only 4 field multiplications. (Some authors say 5 to account for a field multiplication by $(A - 2)/4$; but normally $(A - 2)/4$ is a small integer constant, allowing this multiplication to be eliminated.) This observation motivates separately counting the number of doublings in a differential addition chain.

Computing $mP+nQ$ with the new binary chain takes $256 \cdot 4+(768-256) \cdot 5 = 3584$ field multiplications and a few field divisions. Computing $mP+nQ$ with the new extended-gcd chain takes approximately $88.8 \cdot 4 + (455.0 - 88.8) \cdot 6 = 2552.4$ field multiplications for an average pair (m, n).

There are many more ways to measure the cost of a differential addition chain. Some papers count a field squaring as 0.8 field multiplications, replacing the weights $(6, 5, 4) = 4(1.5, 1.25, 1)$ with $(5.6, 4.6, 3.6) \approx 3.6(1.556, 1.278, 1)$. More useful is to count CPU cycles or other low-level operations. For example, [3, Section 4] reports the following speeds for arithmetic in a field having $2^{255}-19$ elements:

- 10 floating-point operations for a field addition.
- 55 floating-point operations for a field multiplication by $(A-2)/4$.
- 162 floating-point operations for a field squaring.
- 243 floating-point operations for a field multiplication.

A general elliptic-curve addition involves 4 field additions, 2 squarings, and 4 more multiplications, for a total of $4 \cdot 10 + 2 \cdot 162 + 4 \cdot 243 = 1336$ operations. An addition with denominator 1 involves $1336 - 243 = 1093$ operations. An elliptic-curve doubling involves 4 field additions, 1 multiplication by $(A-2)/4$, 2 squarings, and 2 more multiplications, for a total of $4 \cdot 10 + 1 \cdot 55 + 2 \cdot 162 + 2 \cdot 243 = 905$ operations. These figures suggest measuring differential addition chains using weights $(1336, 1093, 905) \approx 905(1.476, 1.208, 1)$.

Beware that the extended-gcd chain takes more effort to compute than the binary chain. Reducing the computation time—perhaps by fast-gcd techniques, or by a more detailed analysis of the parameter space—is outside the scope of this paper. At this point I expect the extended-gcd chain to be used mainly in applications where chains can be precomputed. For example, in the context of elliptic-curve public-key signature verification, a short chain can be computed once by the signer and then quickly checked by each verifier.

Perhaps 3-dimensional versions of the ideas in this paper will also save time in the recent elliptic-curve-signature-verification algorithm of [2]. I will leave this exploration to future research.

3 How does this compare to previous work?

dim	method	additions per bit, 128-bit exponents	additions per bit, 256-bit exponents	field mults per bit, 128-bit exponents	field mults per bit, 256-bit exponents	uniform
2	easy	4.000	4.000	19.000	19.000	yes
2	Schoenmakers	3.500	3.500	17.250	17.250	no
2	Akishita	3.000	3.000	14.250	14.250	no
2	new binary	3.000	3.000	14.000	14.000	yes
2	PRAC	1.820	1.824	10.261	10.283	no
2	new extended gcd	1.757	1.770	9.918	9.970	no
1	easy	2.000	2.000	9.000	9.000	yes
1	standard	1.533	1.560	8.885	8.983	no
1	lower bound	1.440	1.440	? 8.642	? 8.642	

Comparison to the one-dimensional small-difference case

It is easy to write down a *one-dimensional* "binary" differential addition chain that uses 2 small-difference additions per bit: e.g., 512 additions for a 256-bit exponent. The new two-dimensional binary chain in this paper uses 3 small-difference additions per bit: e.g., 768 additions for 256-bit exponents. The new two-dimensional binary chain is considerably more expensive than the one-dimensional binary chain, but not twice as expensive.

Counting field multiplications in the context of elliptic curves, rather than simply additions, produces similar conclusions. The standard one-dimensional binary chain has, for each bit, one addition having difference 1, and one doubling, for a total of 9 field multiplications per bit. The new two-dimensional binary chain has 14 field multiplications per bit.

The standard one-dimensional binary chain has a simple, uniform structure, alternating between a difference-0 addition and a difference-1 addition. In the cryptographic context, this structure reduces the costs of protecting a secret exponent against side-channel attacks; see, e.g., [6, Section 4]. The new two-dimensional binary chain in this paper has the same virtue.

Comparison to the one-dimensional large-difference case

Montgomery pointed out two decades ago, in the one-dimensional case, that it is usually possible to find differential addition chains considerably shorter than the obvious binary chain—although the question of how much shorter is still open!

My experiments with standard methods produced one-dimensional chains with 196.224 additions on average for 1000 random 128-bit primes (1.53300 per bit, with standard deviation 0.02028 per bit), and one-dimensional chains with 399.286 additions on average for 1000 random 256-bit primes (1.55971 per bit, with standard deviation 0.00946 per bit). The new two-dimensional extended-gcd chain in this paper is somewhat more expensive than these one-dimensional chains, although the gap is not as large as in the binary case.

Counting field multiplications produces similar conclusions. For example, my experiments found chains involving 1137.236 field multiplications on average for the same 128-bit primes (8.88466 per bit, with standard deviation 0.11134 per bit). There is again a gap between the one-dimensional case and the two-dimensional case.

Montgomery proved in [10, Theorem 7] that a one-dimensional differential addition chain with k additions, starting from $0, 1$, cannot reach a prime larger than the $(k+2)$nd Fibonacci number. This means that the worst-case chains (and typical-case chains; see [8, Section 7]) have at least $(\log 2)/\log((1+\sqrt{5})/2) \approx 1.44042$ additions per bit. The same proof strategy should be able to produce lower bounds on weighted measures of chain cost, but as far as I know nobody has worked out the details; I don't know whether there's a lower bound of 8.64252 field multiplications per bit, for example.

Note that, when the differential requirement is dropped, addition chains have no asymptotic gap between lower bounds and upper bounds. Every chain needs at least 1 addition per bit; and it is easy to construct two-dimensional addition chains with just $1 + o(1)$ additions per bit. The differential condition doesn't merely replace 1 by a larger constant; it separates the known lower bounds, the known one-dimensional upper bounds, and the known two-dimensional upper bounds. Perhaps future research will manage to close these gaps.

Previous records for the two-dimensional small-difference case

It is easy to write down a reasonably short differential addition chain for (m, n) with all differences in $\{(0, 0), (1, 0), (0, 1), (1, 1), (1, -1)\}$. If m, n each have b bits then the obvious chain has approximately $4b$ additions, including b doublings. In the elliptic-curve context, this chain costs 19 field multiplications per bit.

This paper's new binary chain has only $3b$ additions, including b doublings, for a total of 14 field multiplications per bit. There have been two previous attempts to achieve the same result:

- A 2000 algorithm by Schoenmakers, published in 2003 in [11, Section 3.2.3], often succeeds in handling a bit with 3 small-difference additions, including 1 doubling. Unfortunately, whenever a bit is set at the same position in the binary expansions of m and n, this algorithm uses 2 large-difference additions and 3 small-difference additions, none of which are doublings. Consequently, for an average pair (m, n), this algorithm uses 3.5 additions per bit, including 3 small-difference additions per bit, including 0.75 doublings per bit, for a total of $3.5 \cdot 6 - 3 - 0.75 = 17.25$ field multiplications per bit.
- An algorithm by Akishita in [1] eliminates the large-difference additions in the Schoenmakers algorithm. Consequently, for an average pair (m, n), Akishita's algorithm uses 3 small-difference additions per bit, including 0.75 doublings per bit, for a total of $3 \cdot 5 - 0.75 = 14.25$ field multiplications per bit.

This paper's new binary algorithm pushes the same ideas to their apparent limits, eliminating all of the bad cases in the previous algorithms. It is faster than the algorithms of Schoenmakers and Akishita, and has the advantage of a uniform add-double-add structure, analogous to the uniform double-add structure in the one-dimensional case.

Previous records for the two-dimensional large-difference case

Stam in [11, Conjecture 3.29] reported computer experiments showing that Montgomery's "PRAC" two-dimensional chain in [10, Section 7] takes about 1.82 additions per bit, including 0.33 doublings per bit. My experiments with PRAC produced similar results: chains with 232.913 additions on average for 1000 random coprime 128-bit pairs (1.81963 per bit, with standard deviation 0.05538 per bit), and chains with 467.028 additions on average for 1000 random coprime 256-bit pairs (1.82432 per bit, with standard deviation 0.03857 per bit).

The new extended-gcd chain in this paper has 224.867 additions on average for the same 128-bit pairs (1.75677 per bit, with standard deviation 0.02984 per bit), and 453.090 additions on average for the same 256-bit pairs (1.76988 per bit, with standard deviation 0.02064 per bit).

Counting field multiplications shows similar speedups. In my experiments, PRAC used 1313.408 field multiplications for 128 bits (10.261 per bit, with standard deviation 0.239 per bit) and 2632.532 field multiplications for 256 bits (10.283 per bit, with standard deviation 0.166 per bit). The new chain uses

1269.564 field multiplications for 128 bits (9.918 per bit, with standard deviation 0.163 per bit) and 2552.394 field multiplications for 256 bits (9.970 per bit, with standard deviation 0.111 per bit).

These experiments show that the new chain closes about 20% of the gap between previous two-dimensional chains and one-dimensional chains. Perhaps more important is that we are no longer stuck with PRAC: the new chain opens up a large parameter space for further exploration. One can reasonably hope that future research will produce even shorter chains.

4 How does the new binary chain work?

Here is an example of the new binary chain:

$$\begin{tikzcd}
(0,0) & (1,0) \arrow[dl] \arrow[d, Rightarrow] \arrow[dr] & (0,1) \arrow[dll] & (1,-1) \\
(1,1) \arrow[rr, bend left=60] \arrow[d] \arrow[dr, Rightarrow] \arrow[drr] & (2,0) \arrow[dl] & (2,1) \arrow[d] & \\
(3,1) \arrow[d] & (2,2) \arrow[dl] \arrow[d, Rightarrow] \arrow[dr] & (3,2) \arrow[d] & \\
(5,3) \arrow[d] & (4,4) \arrow[dl] \arrow[dr] & (5,4) \arrow[dl, Rightarrow] \arrow[d] & \\
(9,7) \arrow[d] \arrow[dr, Rightarrow] \arrow[drr] & (10,8) \arrow[dl] & (9,8) \arrow[d] & \\
(19,15) \arrow[d] \arrow[drr] & (18,14) \arrow[dl] & (18,15) \arrow[dl, Rightarrow] \arrow[d] & \\
(37,29) \arrow[d] \arrow[dr, Rightarrow] \arrow[drr] & (36,30) \arrow[dl] & (37,30) \arrow[d] & \\
(73,59) & (74,58) & (74,59) &
\end{tikzcd}$$

Each line after the first has three of the four pairs (a, b), $(a + 1, b)$, $(a, b + 1)$, $(a + 1, b + 1)$ for a unique (a, b). The missing element of $(a + \{0, 1\}, b + \{0, 1\})$ is always chosen as either $(\text{even}, \text{odd})$ or $(\text{odd}, \text{even})$, where the choice is related to the (A, B) for the next line:

- If $(a + A, b + B)$ is $(\text{even}, \text{odd})$ then the choice is $(\text{odd}, \text{even})$.
- If $(a + A, b + B)$ is $(\text{odd}, \text{even})$ then the choice is $(\text{even}, \text{odd})$.
- If $(a + A, b + B)$ is $(\text{even}, \text{even})$ then the lines have the same choices.
- If $(a + A, b + B)$ is (odd, odd) then the lines have opposite choices.

The pair (a, b) is also related to (A, B): it is simply $(\lfloor A/2\rfloor, \lfloor B/2\rfloor)$.

For comparison: The obvious binary chain uses all four pairs (a, b), $(a+1, b)$, $(a, b+1)$, $(a+1, b+1)$. The Schoenmakers chain in [11, Section 3.2.3] omits $(a+1, b+1)$. Akishita's chain in [1] omits $(a+1-(A \bmod 2), b+1-(B \bmod 2))$. This paper's new binary chain omits $(a+(a+d+1 \bmod 2), b+(b+d \bmod 2))$ where d has a relatively complicated definition.

Recursive definition of the new binary chain

Define $C_D(A, B)$ recursively, for all nonnegative integers A and B and for all $D \in \{0, 1\}$, as $C_d(a, b)$ followed by the three pairs

$$\begin{aligned}&(A+(A+1 \bmod 2), B+(B+1 \bmod 2)),\\ &(A+(A \bmod 2), B+(B \bmod 2)),\\ &(A+(A+D \bmod 2), B+(B+D+1 \bmod 2)),\end{aligned}$$

where $a = \lfloor A/2\rfloor$, $b = \lfloor B/2\rfloor$, and

$$d = \begin{cases} 0 & \text{if } (a+A, b+B) \bmod 2 = (0,1)\\ 1 & \text{if } (a+A, b+B) \bmod 2 = (1,0)\\ D & \text{if } (a+A, b+B) \bmod 2 = (0,0)\\ 1-D & \text{if } (a+A, b+B) \bmod 2 = (1,1). \end{cases}$$

Exception: $C_D(0, 0)$ is defined as $(0, 0), (1, 0), (0, 1), (1, -1)$.

Here are the first few examples of this chain $C_D(A, B)$:

$C_0(0, 0)$ is $(0, 0), (1, 0), (0, 1), (1, -1)$.
$C_1(0, 0)$ is $(0, 0), (1, 0), (0, 1), (1, -1)$.
$C_0(1, 0)$ is $(0, 0), (1, 0), (0, 1), (1, -1), (1, 1), (2, 0), (2, 1)$.
$C_1(1, 0)$ is $(0, 0), (1, 0), (0, 1), (1, -1), (1, 1), (2, 0), (1, 0)$.
$C_0(0, 1)$ is $(0, 0), (1, 0), (0, 1), (1, -1), (1, 1), (0, 2), (0, 1)$.
$C_1(0, 1)$ is $(0, 0), (1, 0), (0, 1), (1, -1), (1, 1), (0, 2), (1, 2)$.
$C_0(1, 1)$ is $(0, 0), (1, 0), (0, 1), (1, -1), (1, 1), (2, 2), (2, 1)$.
$C_1(1, 1)$ is $(0, 0), (1, 0), (0, 1), (1, -1), (1, 1), (2, 2), (1, 2)$.

Note for future reference that $C_D(A, B)$ always contains the pair $(1, 1)$ if $(A, B) \neq (0, 0)$.

The rest of this section shows that $C_D(A, B)$ is a differential addition chain that starts with $(0, 0), (1, 0), (0, 1), (1, -1)$ and that follows a uniform add-double-add pattern with all differences in $\{(0, 0), (1, 0), (0, 1), (1, 1), (1, -1)\}$. One can easily force the chain to contain any desired pair (m, n) of nonnegative integers by choosing, e.g., $(A, B) = (m, n)$ and $D = m \bmod 2$.

Minor variations are possible: omitting repeated elements, for example, and omitting unused elements near the end of the chain. Similar comments apply to the other chains in this paper.

The odd-odd pair in each line: first addition

Assume that $(A, B) \neq (0, 0)$. The pair $(A + (A + 1 \bmod 2), B + (B + 1 \bmod 2))$ in $C_D(A, B)$ is equal to $(2a + 1, 2b + 1)$ where $(a, b) = (\lfloor A/2 \rfloor, \lfloor B/2 \rfloor)$ as above.

If $(a, b) = (0, 0)$ then the pair is $(1, 1)$, which can be obtained by adding $(1, 0)$ to $(0, 1)$ with difference $(1, -1)$; so assume that $(a, b) \neq (0, 0)$.

The chain already includes $C_d(a, b)$, which contains three of the four pairs $(a, b), (a + 1, b), (a, b + 1), (a + 1, b + 1)$. Consequently, $(2a + 1, 2b + 1)$ can be obtained by adding $(a + 1, b)$ to $(a, b + 1)$ with difference $(1, -1)$, or by adding $(a + 1, b + 1)$ to (a, b) with difference $(1, 1)$; recall that $(1, 1)$ is also in $C_d(a, b)$.

The even-even pair in each line: doubling

The next pair $(A + (A \bmod 2), B + (B \bmod 2))$ in the chain $C_D(A, B)$ is equal to $(2a + 2(A \bmod 2), 2b + 2(B \bmod 2))$.

If $(a, b) = (0, 0)$ then the pair is $(2, 0)$ or $(0, 2)$ or $(2, 2)$, so it can be obtained by doubling $(1, 0)$ or $(0, 1)$ or $(1, 1)$, all of which appear earlier in the chain; so assume that $(a, b) \neq (0, 0)$.

The chain already contains, via $C_d(a, b)$, all pairs $(a+\{0, 1\}, b+\{0, 1\})$ except $(a + (a + d + 1 \bmod 2), b + (b + d \bmod 2))$. If $(a + (A \bmod 2), b + (B \bmod 2))$ equals the missing pair then $(a+A, b+B) \bmod 2 = (2a+d+1, 2b+d) \bmod 2 = (1 - d, d)$; but if $(a + A, b + B) \bmod 2 = (0, 1)$ then d is 0 by construction, and if $(a + A, b + B) \bmod 2 = (1, 0)$ then d is 1 by construction.

Thus $(a + (A \bmod 2), b + (B \bmod 2))$ is earlier in the chain, and doubling it produces the desired $(A + (A \bmod 2), B + (B \bmod 2))$.

The other pair in each line: second addition

If $D = 0$ then the pair $(A + (A + D \bmod 2), B + (B + D + 1 \bmod 2))$ is equal to $(2a + 2(A \bmod 2), 2b + 1)$. I claim that this pair can be obtained by adding $(a + (A \bmod 2), b + 1)$ and $(a + (A \bmod 2), b)$, with difference $(0, 1)$.

If $(a, b) = (0, 0)$ then $(a + (A \bmod 2), b + 1)$ is either $(0, 1)$ or $(1, 1)$, both of which are already in the chain; and $(a + (A \bmod 2), b)$ is either $(0, 0)$ or $(1, 0)$, both of which are already in the chain. So assume that $(a, b) \neq (0, 0)$.

The chain already contains, via $C_d(a, b)$, all pairs $(a+\{0, 1\}, b+\{0, 1\})$ except $(a+(a+d+1 \bmod 2), b+(b+d \bmod 2))$. Suppose that the missing pair is equal to $(a + (A \bmod 2), b + 1)$ or $(a + (A \bmod 2), b)$. Then $a + (a + d + 1 \bmod 2) = a+(A \bmod 2)$, so $a+A \bmod 2 = 2a+d+1 \bmod 2 = 1-d$. If $(a+A, b+B) \bmod 2 = (0, 1)$ then $d = 0$ by construction, contradiction. If $(a + A, b + B) \bmod 2 = (1, 0)$ then $d = 1$ by construction, contradiction. If $(a + A, b + B) \bmod 2 = (0, 0)$ then $d = D = 0$ by construction, contradiction. If $(a + A, b + B) \bmod 2 = (1, 1)$ then $d = 1 - D = 1$ by construction, contradiction.

Similarly, if $D = 1$, then the pair $(A+(A+D \bmod 2), B+(B+D+1 \bmod 2))$ in $C_D(A, B)$ is equal to $(2a+1, 2b+2(B \bmod 2))$, which can be obtained by adding $(a + 1, b + (B \bmod 2))$ and $(a, b + (B \bmod 2))$, with difference $(1, 0)$.

5 How do large differences help in dimension 1?

Let d, e be coprime integers with $0 \le d \le e$. This section reviews several ways to construct a *one-dimensional* differential addition chain that starts from $0, 1$ and that contains $e - d$ and d and e. Euclid's chain is the simplest but generally longest; Tsuruoka's chain is the most complicated but generally shortest.

Euclid's chain $E(d, e)$ is defined recursively as follows:

$$E(d,e) = \begin{cases} 0, e & \text{if } d = 0 \\ E(e-d, e) & \text{if } e/2 < d \\ E(d, e-d), e & \text{otherwise} \end{cases}$$

For example,

$$E(11, 97) = 0, 1, 2, 3, 5, 7, 9, 11, 20, 31, 42, 53, 64, 75, 86, 97.$$

One can easily prove by induction on e that $E(d, e)$ is a differential addition chain that starts from $0, 1$ and that contains $e - d$ and d and e. The point is that if $e > 1$ then e can be obtained by adding d and $e - d$, since the difference of d and $e - d$ is earlier in the chain.

A more sophisticated differential addition chain $S(d, e)$ is defined recursively as follows:

$$S(d,e) = \begin{cases} 0, e & \text{if } d = 0 \\ S(e-d, e) & \text{if } e/2 < d \\ S(d, e/2), e-d, e & \text{if } 0 < d < e/4 \text{ and } e \in 2\mathbf{Z} \\ S(d, e-d), e & \text{otherwise} \end{cases}$$

For example,

$$S(11, 97) = 0, 1, 2, 3, 4, 5, 9, 10, 11, 21, 32, 43, 75, 86, 97.$$

What's new in $S(d, e)$, compared to $E(d, e)$, is the $S(d, e/2), e - d, e$ case. In this case, e is obtained by doubling $e/2$; d appears in $S(d, e/2)$ by induction; and $e - d$ is obtained by adding $e/2 - d$ to $e/2$.

Bleichenbacher's differential addition chain $B(d, e)$, introduced in [4, Section 5.3] and republished without credit as the main result of [5], is defined recursively as follows:

$$B(d,e) = \begin{cases} 0, e & \text{if } d = 0 \\ B(e-d, e) & \text{if } e/2 < d \\ B(d, e/2), e-d, e & \text{if } 0 < d < e/5 \text{ and } e \in 2\mathbf{Z} \\ B(d, (e+d)/2), e-d, e & \text{if } 0 < d < e/5 \text{ and } e \notin 2\mathbf{Z} \text{ and } e+d \in 2\mathbf{Z} \\ B(d/2, e-d/2), d, e & \text{if } 0 < d < e/5 \text{ and } e \notin 2\mathbf{Z} \text{ and } d \in 2\mathbf{Z} \\ B(d, e-d), e & \text{otherwise} \end{cases}$$

For example,

$$B(11,97) = 0,1,2,3,5,6,10,11,21,32,43,54,86,97.$$

Beware that there are two typographical errors in [4, Section 5.3]: "$x-y, y/2, z$" should be "$x-y/2, y/2, z$" and "$x/2, x-y, z$" should be "$x/2, y-x/2, z$."

Tsuruoka's differential addition chain $T(d,e)$, introduced in [12], is defined recursively as follows:

$$T(d,e) = \begin{cases} 0, e & \text{if } d = 0 \\ T(e-d, e) & \text{if } e < 2d \\ T(d, e/2), e-d, e & \text{if } 2d \le e \le 2.09d \text{ and } e \in 2\mathbf{Z} \\ T(d, e/2), e-d, e & \text{if } 3.92d \le e \text{ and } e \in 2\mathbf{Z} \\ T(d, (e+d)/3), \\ \quad (2e-d)/3, e-d, e & \text{if not and } 5.7d \le e \text{ and } e+d \in 3\mathbf{Z} \\ T(d, (e-d)/3), (e+2d)/3, \\ \quad (2e-2d)/3, e-d, e & \text{if not and } 4.9d \le e \text{ and } e-d \in 3\mathbf{Z} \\ T(d, (e+d)/2), e-d, e & \text{if not and } 4.9d \le e \text{ and } d+e \in 2\mathbf{Z} \\ T(d, e/3), \\ \quad d+e/3, 2e/3, e-d, e & \text{if not and } 6.8d \le e \text{ and } e \in 3\mathbf{Z} \\ T(d/2, e-d/2), d, e & \text{if not and } 9d \le e \text{ and } d \in 6\mathbf{Z} \\ T(d, e-d), e & \text{otherwise} \end{cases}$$

For example,

$$T(11,97) = 0,1,2,3,4,7,11,14,25,36,61,86,97.$$

All of these chains were designed with the goal of minimizing length, i.e., the number of additions. Slightly different constructions should do better in other cost measures, such as the number of field multiplications in the elliptic-curve context.

CFRC, PRAC, etc.

With dual algorithms one can construct a two-dimensional differential addition-subtraction chain for (d,e). For example, given $P, Q, P-Q$, one can first compute $P+Q$, and then compute $dP+eQ$ as $d(P+Q)+(e-d)Q$.

Montgomery's "CFRC" chain in [10, Section 5] is a simple construction of a two-dimensional chain for (d,e), comparable to Euclid's chain. Montgomery's "PRAC" chain in [10, Section 7] is a more complicated construction, comparable to (and predating) Tsuruoka's chain.

Beware that duality does not preserve costs for differential chains—see, e.g., [11, Example 3.28] and [11, Section 3.4, final paragraph]—so optimizing a one-dimensional chain for d and e is not the same problem as optimizing a two-dimensional chain for the pair (d,e).

Allowing d to vary

The standard way to construct a one-dimensional differential addition chain for e is to choose some d coprime to e and use one of the above algorithms to construct a chain containing $e-d, d, e$. Here are three refinements:

- Choose d to be very close to $2e/(1+\sqrt{5})$. This guarantees that the top half of the bits of e will be handled with about 1.44 additions per bit; see, e.g., [11, Proposition 3.34]. For example, with $e = 100$, choosing $d = 61$ produces a chain ending $17, 22, 39, 61, 100$.
- Try many d's and take the shortest chain for e. One can, for example, take a range of d's around $2e/(1+\sqrt{5})$, or around e/α for various constants α having continued fractions consisting of almost entirely 1's and a few 2's.
- If e has a known factor g, construct a chain for e by constructing a chain for e/g, multiplying it by g, and merging the result with a chain for g. This generally produces shorter chains (for a given amount of d-searching time) than handling e directly.

All of these improvements were suggested by Montgomery in [10, Section 7], in the context of Montgomery's PRAC chain.

Here is a simple experiment to illustrate the importance of trying many d's. Consider each prime number e below 10^6. For each e, try several successive d's coprime to e, starting just above $2e/(1+\sqrt{5})$; find the shortest $E(d, e)$, the shortest $S(d, e)$, the shortest $B(d, e)$, and the shortest $T(d, e)$. Average the number of additions in these chains as e varies. The following table shows the resulting averages:

number of d's tried	1	2	4	8	16	32	64	128
$E((\text{best } d), e)$	47.550	34.405	31.286	29.912	29.364	28.876	28.579	28.428
$S((\text{best } d), e)$	34.125	29.630	28.758	28.371	28.194	28.048	27.950	27.899
$B((\text{best } d), e)$	30.794	29.606	28.818	28.415	28.241	28.093	27.993	27.936
$T((\text{best } d), e)$	29.159	28.723	28.431	28.220	28.105	27.996	27.919	27.875

Beware that [12, Section 4.2] uses only two d's, and [11, Algorithm 3.33] uses only one d. This is the main reason for the discrepancies between 1.64 additions per bit in [11], 1.61 additions per bit in [12, Table 7], and 1.56 additions per bit in Section 2 of this paper. The benefit of trying many chains, and keeping the shortest, was pointed out by Montgomery but does not seem to have been adequately emphasized in the literature. Note that, as shown by the crossover between the S and B rows in the above table, optimizing chains for many d's is not the same as optimizing them for a single d.

Another way to generate good chains for e is to modify the above chains by occasionally skipping (or including) a division step. One can, for example, count the number of divisions used in building a chain, and skip a division when the counter is 3 or 8 or 14. By using many sets in place of $\{3, 8, 14\}$ one should obtain many reasonably good chains. I don't know whether this is more effective than trying additional d's; I will leave this exploration to future research. The same idea can also be used in the more complicated context of Section 6.

6 How does the new extended-gcd chain work?

This section presents a new method to construct a two-dimensional differential addition-subtraction chain for the pair (e, e'), given nonnegative integers e and e'.

If g is a nontrivial common factor of e, e' then one can build a two-dimensional chain for (e, e') out of a one-dimensional chain for g and a two-dimensional chain for $(e/g, e'/g)$. The resulting chains are usually shorter than chains for coprime pairs (e, e'), since one-dimensional chains are relatively cheap. Assume from now on that e, e' are coprime.

Choose a small nonzero integer Δ. Find integers d, d' with $0 < d < e$, $0 < d' < e'$, and $d'e - de' = \Delta$. The standard way to do this is to perform an extended-gcd computation to find integers a, b such that $ae + be' = 1$, and then compute $(d, d') = (-b\Delta \bmod e, a\Delta \bmod e')$.

Recall the strategies of Section 5 for constructing a differential addition chain that contains the three integers $e - d$, d, and e. Use the same strategies to construct a differential addition chain that contains the three pairs $(e-d, e'-d')$, (d, d'), and (e, e').

Example: One strategy in Section 5, used when e is even and e is much larger than d, is to recursively construct a chain that contains $e/2 - d$, d, and $e/2$, and then use two additions to obtain $e-d$ and e. Here is a two-dimensional version of the same strategy: when (e, e') is even, and e is much larger than d, recursively construct a chain that contains $(e/2 - d, e'/2 - d')$, (d, d'), and $(e/2, e'/2)$, and then use two additions to obtain $(e - d, e' - d')$ and (e, e').

If $\Delta = \pm 1$ then the simplest strategy—repeated subtraction, as in Euclid's chain—will eventually produce $(0, 1)$ and $(1, 0)$. For example, take $(e, e') = (314, 271)$ and $(d, d') = (73, 63)$. Subtracting $(73, 63)$ a few times from $(314, 271)$ produces $(22, 19)$; subtracting $(22, 19)$ a few times from $(73, 63)$ produces $(7, 6)$; subtracting $(7, 6)$ a few times from $(22, 19)$ produces $(1, 1)$; subtracting $(1, 1)$ several times from $(7, 6)$ produces $(1, 0)$; subtracting $(1, 0)$ from $(1, 1)$ produces $(0, 1)$. The corresponding differential addition chain is

$$\begin{gathered}(0,0),(1,0),(0,1),(1,-1),(1,1),(2,1),(3,2),(4,3),\\(5,4),(6,5),(7,6),(8,7),(15,13),(22,19),(29,25),\\(51,44),(73,63),(95,82),(168,145),(241,208),(314,271).\end{gathered}$$

One cannot expect to see $(0, 1)$ and $(1, 0)$ if $|\Delta| \geq 2$. Repeated subtraction reduces the basis $(d, d'), (e, e')$ of a determinant-Δ lattice, producing two small basis vectors; for example, given $(e, e') = (314, 271)$ and $(d, d') = (146, 126)$, repeated subtraction produces $(0, 1), (2, 0)$. The more sophisticated strategies displayed in Section 5 often remove factors of 2 and 3 but still can't be expected to produce $(0, 1)$ and $(1, 0)$.

So I use a backup plan along with the strategies of Section 5: to construct a differential addition-subtraction chain with $(e-d, e'-d')$, (d, d'), and (e, e') when d, d', e, e' are small, I use Montgomery's PRAC to generate separate chains for $(e-d, e'-d')$, (d, d'), and (e, e'), and then I merge those chains. This backup plan is obviously rather crude, and I have in mind several improvements to explore, but the backup plan accounts for only a small fraction of the exponent bits.

The experiments described in Section 2 used the following choices for Δ: 1 or 5 or 7, times any power of 3 keeping the product below $e^{1/4}$, times any power of 2 keeping the product below $e^{1/2}$. I used the most straightforward two-dimensional adaptation of Tsuruoka's strategies, with the same d/e cutoffs as Tsuruoka, together with the backup plan described above. I also tried simply applying PRAC to (e, e'), although this was suboptimal for more than 90% of the pairs (e, e'). I kept the lowest-cost chain that I found for each (e, e'). I saved time by aborting chains early if they seemed unproductive compared to other known chains.

Example: Consider $(e, e') = (1863038891, 3401169406)$. I tried $\Delta = 2^5 3^3$, producing $(d, d') = (1201249765, 2193005186)$. I then constructed smaller pairs, Tsuruoka-style, ending up with a 55-addition chain for (e, e'):

$$\begin{gathered}(1863038891, 3401169406)\\(1201249765, 2193005186)\\(661789126, 1208164220)\\(539460639, 984840966)\\(330894563, 604082110)\\(208566076, 380758856)\\ \vdots\\(6, 11)\\(5, 9)\\(4, 7)\\(3, 5)\\(2, 3)\\(1, 2)\\(1, 1)\end{gathered}$$

For comparison, the PRAC chain for (e, e') uses 62 additions.

Another example: Consider $(e, e') = (647016469, 352055910)$. I tried $\Delta = 2^4 3^2 5$, producing $(d, d') = (379968888, 206749440)$. I worked backwards to build a chain containing $(e - d, e' - d')$, (d, d'), and (e, e'), starting from $(1, 0)$, $(8, 5)$, and $(9, 5)$. I hooked this chain to PRAC chains for $(8, 5)$ and $(9, 5)$, and ended up with a 52-addition chain for (e, e'). For comparison, the PRAC chain for (e, e') uses 61 additions.

The flexibility of trying many Δ's, and choosing the lowest-cost chain that results, is an important feature of the extended-gcd chain. For example, in my 256-bit experiments, changing "1 or 5 or 7" to "1" increased the average number of additions from 1.76988 per bit to 1.78148 per bit, although it had the benefit of reducing the chain-search time.

References

1. Toru Akishita, *Fast simultaneous scalar multiplication on elliptic curve with Montgomery form*, in [13] (2001), 255–268. Citations in this paper: §3, §4.

2. Adrian Antipa, Daniel Brown, Robert Gallant, Rob Lambert, René Struik, Scott Vanstone, *Accelerated verification of ECDSA signatures* (2005). URL: `http://www.cacr.math.uwaterloo.ca/techreports/2005/tech_reports2005.html`. Citations in this paper: §2.
3. Daniel J. Bernstein, *Curve25519: new Diffie-Hellman speed records* (2006). URL: `http://cr.yp.to/papers.html#curve25519`. ID `4230efdfa673480fc079449d90f322c0`. Citations in this paper: §2.
4. Daniel Bleichenbacher, *Efficiency and security of cryptosystems based on number theory*, Ph.D. thesis, ETH Zürich, 1996. URL: `http://www.bell-labs.com/user/bleichen/diss/thesis.html`. Citations in this paper: §5, §5.
5. S. Y. Chiou, C. S. Laih, *An efficient algorithm for computing the Luc chain*, IEE Proceedings on Computers and Digital Techniques **147** (2000), 263–265. Citations in this paper: §5.
6. Marc Joye, Sung-Ming Yen, *The Montgomery powering ladder*, in [7] (2003), 291–302. URL: `http://www.gemplus.com/smart/rd/publications/pdf/JY03mont.pdf`. Citations in this paper: §3.
7. Burton S. Kaliski Jr., Çetin Kaya Koç, Christof Paar (editors), *Cryptographic hardware and embedded systems—CHES 2002, 4th international workshop, Redwood Shores, CA, USA, August 13–15, 2002, revised papers*, Lecture Notes in Computer Science, 2523, Springer-Verlag, 2003. ISBN 3–540–00409–2. See [6].
8. Martin Kutz, *Lower bounds for Lucas chains*, SIAM Journal on Computing **31** (2002), 1896–1908. ISSN 0097–5397. URL: `http://www.mpi-sb.mpg.de/~mkutz/publications.html`. Citations in this paper: §1, §1, §3.
9. Peter L. Montgomery, *Speeding the Pollard and elliptic curve methods of factorization*, Mathematics of Computation **48** (1987), 243–264. ISSN 0025–5718. MR 88e:11130. URL: `http://links.jstor.org/sici?sici=0025-5718(198701)48:177<243:STPAEC>2.0.CO;2-3`. Citations in this paper: §2.
10. Peter L. Montgomery, *Evaluating recurrences of form $X_{m+n} = f(X_m, X_n, X_{m-n})$ via Lucas chains* (1992). URL: `ftp://ftp.cwi.nl/pub/pmontgom/Lucas.ps.gz`. Citations in this paper: §1, §1, §3, §3, §5, §5, §5.
11. Martijn Stam, *Speeding up subgroup cryptosystems*, Ph.D. thesis, Technische Universiteit Eindhoven, 2003. URL: `http://www.cs.bris.ac.uk/Publications/pub_by_author.jsp?id=137272`. Citations in this paper: §1, §1, §1, §1, §2, §3, §3, §4, §5, §5, §5, §5, §5.
12. Yukio Tsuruoka, *Computing short Lucas chains for elliptic curve cryptosystems*, IEICE Transactions on Fundamentals **E84-A(5)** (2001), 1227–1233. Citations in this paper: §1, §1, §5, §5, §5.
13. Serge Vaudenay, Amr M. Youssef (editors), *Selected areas in cryptography: 8th annual international workshop, SAC 2001, Toronto, Ontario, Canada, August 16–17, 2001, revised papers*, Lecture Notes in Computer Science, 2259, Springer, 2001. ISBN 3–540–43066–0. MR 2004k:94066. See [1].