High-speed cryptography, part 2: more elliptic-curve formulas; field arithmetic

Daniel J. Bernstein
University of Illinois at Chicago &
Technische Universiteit Eindhoven

Speed-oriented Jacobian standards

2000 IEEE "Std 1363" uses Weierstrass curves in Jacobian coordinates to "provide the fastest arithmetic on elliptic curves." Also specifies a method of choosing curves $y^2 = x^3 - 3x + b$.

2000 NIST "FIPS 186–2" standardizes five such curves.

2005 NSA "Suite B" recommends two of the NIST curves as the only public-key cryptosystems for U.S. government use.

eed cryptography,

iptic-curve formulas;
hmetic

. Bernstein
ty of Illinois at Chicago &
che Universiteit Eindhoven

<u>Speed-oriented Jacobian standards</u>

2000 IEEE "Std 1363" uses Weierstrass curves in Jacobian coordinates to "provide the fastest arithmetic on elliptic curves." Also specifies a method of choosing curves $y^2 = x^3 - 3x + b$.

2000 NIST "FIPS 186–2" standardizes five such curves.

2005 NSA "Suite B" recommends two of the NIST curves as the only public-key cryptosystems for U.S. government use.

<u>Projectiv</u>

1986 Ch
Speed u
$(X/Z^2, Y$
$7\mathbf{M} + 3\mathbf{S}$
$12\mathbf{M} + 2$
$12\mathbf{M} + 2$

Option h
DBL do
But AD
some ap
batch si

graphy,

formulas;

n

is at Chicago &

siteit Eindhoven

Speed-oriented Jacobian standards

2000 IEEE "Std 1363" uses Weierstrass curves in Jacobian coordinates to "provide the fastest arithmetic on elliptic curves." Also specifies a method of choosing curves $y^2 = x^3 - 3x + b$.

2000 NIST "FIPS 186–2" standardizes five such curves.

2005 NSA "Suite B" recommends two of the NIST curves as the only public-key cryptosystems for U.S. government use.

Projective for Wei

1986 Chudnovsky–
Speed up ADD by
$(X/Z^2, Y/Z^3)$ to (
$7\mathbf{M} + 3\mathbf{S}$ for DBL
$12\mathbf{M} + 2\mathbf{S}$ for AD
$12\mathbf{M} + 2\mathbf{S}$ for reA

Option has been n
DBL dominates in
But ADD dominat
some applications:
batch signature ve

ago &
hoven

Speed-oriented Jacobian standards

2000 IEEE "Std 1363" uses Weierstrass curves in Jacobian coordinates to "provide the fastest arithmetic on elliptic curves." Also specifies a method of choosing curves $y^2 = x^3 - 3x + b$.

2000 NIST "FIPS 186–2" standardizes five such curves.

2005 NSA "Suite B" recommends two of the NIST curves as the only public-key cryptosystems for U.S. government use.

Projective for Weierstrass

1986 Chudnovsky–Chudnovs
Speed up ADD by switching
$(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z$
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly igno
DBL dominates in ECDH et
But ADD dominates in
some applications: e.g.,
batch signature verification.

Speed-oriented Jacobian standards

2000 IEEE "Std 1363" uses Weierstrass curves in Jacobian coordinates to "provide the fastest arithmetic on elliptic curves." Also specifies a method of choosing curves $y^2 = x^3 - 3x + b$.

2000 NIST "FIPS 186–2" standardizes five such curves.

2005 NSA "Suite B" recommends two of the NIST curves as the only public-key cryptosystems for U.S. government use.

Projective for Weierstrass

1986 Chudnovsky–Chudnovsky: Speed up ADD by switching from $(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z)$.
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$.
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly ignored: DBL dominates in ECDH etc. But ADD dominates in some applications: e.g., batch signature verification.

riented Jacobian standards

EE "Std 1363"
ierstrass curves
bian coordinates
vide the fastest
tic on elliptic curves."
ecifies a method of
g curves $y^2 = x^3 - 3x + b$.

ST "FIPS 186–2"
lizes five such curves.

SA "Suite B" recommends
he NIST curves as
public-key cryptosystems
government use.

Projective for Weierstrass

1986 Chudnovsky–Chudnovsky:
Speed up ADD by switching from
$(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z)$.
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$.
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly ignored:
DBL dominates in ECDH etc.
But ADD dominates in
some applications: e.g.,
batch signature verification.

Montgo

1987 Mo

Use by^2
Choose

$2(x_2, y_2$

$\Rightarrow x_4 =$

(x_3, y_3)
(x_3, y_3)

$\Rightarrow x_5 =$

cobian standards

363"

urves

nates

stest

tic curves."

ethod of

$^2 = x^3 - 3x + b$.

186–2"

uch curves.

B" recommends

urves as

y cryptosystems

nt use.

Projective for Weierstrass

1986 Chudnovsky–Chudnovsky:
Speed up ADD by switching from $(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z)$.
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$.
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly ignored:
DBL dominates in ECDH etc.
But ADD dominates in
some applications: e.g.,
batch signature verification.

Montgomery curve

1987 Montgomery

Use $by^2 = x^3 + a$

Choose small $(a +$

$2(x_2, y_2) = (x_4, y$

$\Rightarrow x_4 = \frac{(x_2^2 -}{4x_2(x_2^2 +}$

$(x_3, y_3) - (x_2, y_2)$

$(x_3, y_3) + (x_2, y_2)$

$\Rightarrow x_5 = \frac{(x_2 x_3 -}{x_1(x_2 -}$

ndards

”

$3x + b$.

s.

mends

stems

Projective for Weierstrass

1986 Chudnovsky–Chudnovsky:
Speed up ADD by switching from $(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z)$.
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$.
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly ignored: DBL dominates in ECDH etc. But ADD dominates in some applications: e.g., batch signature verification.

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$2(x_2, y_2) = (x_4, y_4)$
$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}$$

$(x_3, y_3) - (x_2, y_2) = (x_1, y_1$
$(x_3, y_3) + (x_2, y_2) = (x_5, y_5$
$$\Rightarrow x_5 = \frac{(x_2x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

Projective for Weierstrass

1986 Chudnovsky–Chudnovsky:
Speed up ADD by switching from $(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z)$.
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$.
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly ignored:
DBL dominates in ECDH etc.
But ADD dominates in some applications: e.g., batch signature verification.

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$$2(x_2, y_2) = (x_4, y_4)$$
$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}.$$

$$(x_3, y_3) - (x_2, y_2) = (x_1, y_1),$$
$$(x_3, y_3) + (x_2, y_2) = (x_5, y_5)$$
$$\Rightarrow x_5 = \frac{(x_2 x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

ve for Weierstrass

udnovsky–Chudnovsky:
p ADD by switching from
Y/Z^3) to $(X/Z, Y/Z)$.
S for DBL if $a = -3$.
2**S** for ADD.
2**S** for reADD.

has been mostly ignored:
minates in ECDH etc.
D dominates in
plications: e.g.,
gnature verification.

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$2(x_2, y_2) = (x_4, y_4)$
$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}.$$

$(x_3, y_3) - (x_2, y_2) = (x_1, y_1)$,
$(x_3, y_3) + (x_2, y_2) = (x_5, y_5)$
$$\Rightarrow x_5 = \frac{(x_2 x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

Represer
as $(X{:}Z$

$B = (X_2$
$C = (X_2$
$D = B -$
$Z_4 = D$
$2(X_2{:}Z_2$

$(X_3{:}Z_3)$
$E = (X_3$
$F = (X_3$
$X_5 = Z_1$
$Z_5 = X_1$
$(X_3{:}Z_3)$

erstrass

-Chudnovsky:

switching from

$(X/Z, Y/Z)$.

if $a = -3$.

D.

DD.

nostly ignored:

ECDH etc.

tes in

e.g.,

erification.

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$2(x_2, y_2) = (x_4, y_4)$

$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}.$$

$(x_3, y_3) - (x_2, y_2) = (x_1, y_1)$,
$(x_3, y_3) + (x_2, y_2) = (x_5, y_5)$

$$\Rightarrow x_5 = \frac{(x_2x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

Represent (x, y)
as $(X{:}Z)$ satisfyin

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 =$
$Z_4 = D \cdot (C + D($
$2(X_2{:}Z_2) = (X_4{:}Z$

$(X_3{:}Z_3) - (X_2{:}Z_2$
$E = (X_3 - Z_3) \cdot ($
$F = (X_3 + Z_3) \cdot ($
$X_5 = Z_1 \cdot (E + F$
$Z_5 = X_1 \cdot (E - F$
$(X_3{:}Z_3) + (X_2{:}Z_2$

sky:
from
).
3.

ored:
c.

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$2(x_2, y_2) = (x_4, y_4)$

$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}.$$

$(x_3, y_3) - (x_2, y_2) = (x_1, y_1)$,
$(x_3, y_3) + (x_2, y_2) = (x_5, y_5)$

$$\Rightarrow x_5 = \frac{(x_2 x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

Represent (x, y)
as $(X{:}Z)$ satisfying $x = X/Z$

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a+2)/4)$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z$
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z$

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$2(x_2, y_2) = (x_4, y_4)$

$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}.$$

$(x_3, y_3) - (x_2, y_2) = (x_1, y_1)$,
$(x_3, y_3) + (x_2, y_2) = (x_5, y_5)$

$$\Rightarrow x_5 = \frac{(x_2 x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

Represent (x, y)
as $(X{:}Z)$ satisfying $x = X/Z$.

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a+2)/4) \Rightarrow$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z_1)$,
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z_5)$.

mery curves

ontgomery:

$= x^3 + ax^2 + x.$

small $(a+2)/4$.

$) = (x_4, y_4)$

$$\frac{(x_2^2-1)^2}{4x_2(x_2^2+ax_2+1)}.$$

$- (x_2, y_2) = (x_1, y_1),$

$+ (x_2, y_2) = (x_5, y_5)$

$$\frac{(x_2x_3-1)^2}{x_1(x_2-x_3)^2}.$$

Represent (x, y)
as $(X{:}Z)$ satisfying $x = X/Z$.

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a+2)/4) \Rightarrow$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z_1)$,
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z_5)$.

This rep
does not
DADD,
Q, R, Q

e.g. $2P$,
e.g. $3P$,
e.g. $6P$,

$2\mathbf{M} + 2\mathbf{S}$
$4\mathbf{M} + 2\mathbf{S}$
Save $1\mathbf{M}$

Easily co
$\approx \lg n$ D
Almost a
Relativel

es

:

$x^2 + x$.

$2)/4$.

$_4)$

$- 1)^2$

$ax_2 + 1)$.

$= (x_1, y_1)$,

$= (x_5, y_5)$

$1)^2$

$x_3)^2$.

Represent (x, y) as $(X{:}Z)$ satisfying $x = X/Z$.

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a + 2)/4) \Rightarrow$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z_1)$,
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z_5)$.

This representatio
does not allow AD
DADD, "differenti
$Q, R, Q - R \mapsto Q$

e.g. $2P, P, P \mapsto 3$
e.g. $3P, 2P, P \mapsto$
e.g. $6P, 5P, P \mapsto$

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ fo
$4\mathbf{M} + 2\mathbf{S}$ for DAD
Save $1\mathbf{M}$ if $Z_1 =$

Easily compute $n($
$\approx \lg n$ DBL, $\approx \lg$
Almost as fast as
Relatively slow for

.

),

)

Represent (x, y)
as $(X{:}Z)$ satisfying $x = X/Z$.

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a + 2)/4) \Rightarrow$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z_1)$,
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z_5)$.

This representation
does not allow ADD but it a
DADD, “differential additio
$Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ u
$\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards n
Relatively slow for $mP + n$

Represent (x, y)
as $(X{:}Z)$ satisfying $x = X/Z$.

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a + 2)/4) \Rightarrow$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z_1)$,
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z_5)$.

This representation
does not allow ADD but it allows
DADD, “differential addition”:
$Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ using
$\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards nP.
Relatively slow for $mP + nQ$ etc.

nt (x, y)

) satisfying $x = X/Z$.

$_2 + Z_2)^2$,

$_2 - Z_2)^2$,

$- C$, $X_4 = B \cdot C$,

$\cdot (C + D(a + 2)/4) \Rightarrow$

$) = (X_4{:}Z_4)$.

$- (X_2{:}Z_2) = (X_1{:}Z_1)$,

$_3 - Z_3) \cdot (X_2 + Z_2)$,

$_3 + Z_3) \cdot (X_2 - Z_2)$,

$_1 \cdot (E + F)^2$,

$_1 \cdot (E - F)^2 \Rightarrow$

$+ (X_2{:}Z_2) = (X_5{:}Z_5)$.

This representation does not allow ADD but it allows DADD, "differential addition": $Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ using $\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards nP.
Relatively slow for $mP + nQ$ etc.

Doubling

2006 Do

Use y^2 =
Choose s

Use $(X$
to repres

$3\mathbf{M} + 4\mathbf{S}$
How? F
where φ

2007 Be
$2\mathbf{M} + 5\mathbf{S}$
on the s

g $x = X/Z$.

$= B \cdot C$,

$a + 2)/4) \Rightarrow$

$Z_4)$.

$) = (X_1{:}Z_1)$,

$X_2 + Z_2)$,

$X_2 - Z_2)$,

$)^2$,

$)^2 \Rightarrow$

$) = (X_5{:}Z_5)$.

This representation
does not allow ADD but it allows
DADD, "differential addition":
$Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ using
$\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards nP.
Relatively slow for $mP + nQ$ etc.

<u>Doubling-oriented</u>

2006 Doche–Icart–

Use $y^2 = x^3 + ax$
Choose small a.

Use $(X : Y : Z : Z$
to represent $(X/Z$

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ fo
How? Factor DBL
where φ is a 2-iso

2007 Bernstein–La
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ fo
on the same curve

Z.

$\Rightarrow$

$Z_1)$,

$Z_5)$.

This representation
does not allow ADD but it allows
DADD, “differential addition”:
$Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ using
$\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards nP.
Relatively slow for $mP + nQ$ etc.

<u>Doubling-oriented curves</u>

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.
Choose small a.

Use $(X : Y : Z : Z^2)$
to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.
How? Factor DBL as $\hat{\varphi}(\varphi)$
where φ is a 2-isogeny.

2007 Bernstein–Lange:
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL
on the same curves.

This representation
does not allow ADD but it allows
DADD, "differential addition":
$Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ using
$\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards nP.
Relatively slow for $mP + nQ$ etc.

<u>Doubling-oriented curves</u>

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.
Choose small a.

Use $(X : Y : Z : Z^2)$
to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.
How? Factor DBL as $\hat{\varphi}(\varphi)$
where φ is a 2-isogeny.

2007 Bernstein–Lange:
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL
on the same curves.

resentation

t allow ADD but it allows

"differential addition":

$- R \mapsto Q + R$.

$P, P \mapsto 3P$.

$2P, P \mapsto 5P$.

$5P, P \mapsto 11P$.

$\mathbf{S} + 1\mathbf{D}$ for DBL.

$\mathbf{S}$ for DADD.

$\mathbf{M}$ if $Z_1 = 1$.

ompute $n(X_1 : Z_1)$ using

DBL, $\approx \lg n$ DADD.

as fast as Edwards nP.

ly slow for $mP + nQ$ etc.

Doubling-oriented curves

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.

Choose small a.

Use $(X : Y : Z : Z^2)$

to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.

How? Factor DBL as $\hat{\varphi}(\varphi)$

where φ is a 2-isogeny.

2007 Bernstein–Lange:

$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL

on the same curves.

$12\mathbf{M} + 5$

Slower A

typically

of the ve

But isog

Example

fast DBI

genus-2

using sir

Tricky b

tripling-c

(see 200

double-b

n
D but it allows
al addition":
$+ R$.

P.
$5P$.
$11P$.

r DBL.
D.
1.

$(X_1 : Z_1)$ using
n DADD.
Edwards nP.
$mP + nQ$ etc.

Doubling-oriented curves

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.
Choose small a.

Use $(X : Y : Z : Z^2)$
to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.
How? Factor DBL as $\hat{\varphi}(\varphi)$
where φ is a 2-isogeny.

2007 Bernstein–Lange:
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL
on the same curves.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ f
Slower ADD than
typically outweighi
of the very fast D

But isogenies are
Example, 2005 Ga
fast DBL+DADD
genus-2 hyperellip
using similar facto

Tricky but potenti
tripling-oriented c
(see 2006 Doche–
double-base chain

allows
n":

using

aP.
Q etc.

Doubling-oriented curves

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.
Choose small a.

Use $(X : Y : Z : Z^2)$
to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.
How? Factor DBL as $\hat{\varphi}(\varphi)$
where φ is a 2-isogeny.

2007 Bernstein–Lange:
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL
on the same curves.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other syst
typically outweighing benefit
of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry:
fast DBL+DADD on Jacobi
genus-2 hyperelliptic curves,
using similar factorization.

Tricky but potentially helpfu
tripling-oriented curves
(see 2006 Doche–Icart–Kohe
double-base chains, . . .

Doubling-oriented curves

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.
Choose small a.

Use $(X : Y : Z : Z^2)$
to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.
How? Factor DBL as $\hat{\varphi}(\varphi)$
where φ is a 2-isogeny.

2007 Bernstein–Lange:
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL
on the same curves.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other systems,
typically outweighing benefit
of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry:
fast DBL+DADD on Jacobians of
genus-2 hyperelliptic curves,
using similar factorization.

Tricky but potentially helpful:
tripling-oriented curves
(see 2006 Doche–Icart–Kohel),
double-base chains, . . .

g-oriented curves

oche–Icart–Kohel:

$= x^3 + ax^2 + 16ax$.

small a.

: $Y : Z : Z^2$)

sent $(X/Z, Y/Z^2)$.

$\mathbf{S} + 2\mathbf{D}$ for DBL.

actor DBL as $\hat{\varphi}(\varphi)$

is a 2-isogeny.

rnstein–Lange:

$\mathbf{S} + 2\mathbf{D}$ for DBL

ame curves.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other systems,
typically outweighing benefit
of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry:
fast DBL+DADD on Jacobians of
genus-2 hyperelliptic curves,
using similar factorization.

Tricky but potentially helpful:
tripling-oriented curves
(see 2006 Doche–Icart–Kohel),
double-base chains, . . .

Hessian

Credited
by 1986

$(X : Y :$

on $x^3 +$

$12\mathbf{M}$ for

$X_3 = Y_1$

$Y_3 = X_1$

$Z_3 = Z_1$

$6\mathbf{M} + 3\mathbf{S}$

curves

-Kohel:

$^2 + 16ax$.

$Z^2)$

$, Y/Z^2)$.

r DBL.

L as $\hat{\varphi}(\varphi)$

geny.

ange:

r DBL

es.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other systems, typically outweighing benefit of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry: fast DBL+DADD on Jacobians of genus-2 hyperelliptic curves, using similar factorization.

Tricky but potentially helpful: tripling-oriented curves (see 2006 Doche–Icart–Kohel), double-base chains, . . .

Hessian curves

Credited to Sylves
by 1986 Chudnovs

$(X : Y : Z)$ repres
on $x^3 + y^3 + 1 =$

$12\mathbf{M}$ for ADD:

$X_3 = Y_1 X_2 \cdot Y_1 Z_2$

$Y_3 = X_1 Z_2 \cdot X_1 Y_2$

$Z_3 = Z_1 Y_2 \cdot Z_1 X_2$

$6\mathbf{M} + 3\mathbf{S}$ for DBL

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other systems, typically outweighing benefit of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry:
fast DBL+DADD on Jacobians of genus-2 hyperelliptic curves, using similar factorization.

Tricky but potentially helpful:
tripling-oriented curves
(see 2006 Doche–Icart–Kohel),
double-base chains, . . .

Hessian curves

Credited to Sylvester
by 1986 Chudn

$(X : Y : Z)$ represent $(X/Z,$
on $x^3 + y^3 + 1 = 3dxy$.

$12\mathbf{M}$ for ADD:
$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot$
$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot$
$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot$

$6\mathbf{M} + 3\mathbf{S}$ for DBL.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other systems, typically outweighing benefit of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry:
fast DBL+DADD on Jacobians of genus-2 hyperelliptic curves, using similar factorization.

Tricky but potentially helpful:
tripling-oriented curves
(see 2006 Doche–Icart–Kohel), double-base chains, ...

Hessian curves

Credited to Sylvester
by 1986 Chudnovsky–Chudnovsky:

$(X : Y : Z)$ represent $(X/Z, Y/Z)$
on $x^3 + y^3 + 1 = 3dxy$.

$12\mathbf{M}$ for ADD:

$$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot X_1Y_2,$$
$$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot Z_1X_2,$$
$$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot Y_1Z_2.$$

$6\mathbf{M} + 3\mathbf{S}$ for DBL.

5**S** + 1**D** for ADD.
ADD than other systems,
outweighing benefit
ery fast DBL.

genies are useful.
e, 2005 Gaudry:
L+DADD on Jacobians of
hyperelliptic curves,
milar factorization.

ut potentially helpful:
oriented curves
06 Doche–Icart–Kohel),
base chains, . . .

Hessian curves

Credited to Sylvester
by 1986 Chudnovsky–Chudnovsky:

$(X : Y : Z)$ represent $(X/Z, Y/Z)$
on $x^3 + y^3 + 1 = 3dxy$.

12**M** for ADD:

$$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot X_1Y_2,$$
$$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot Z_1X_2,$$
$$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot Y_1Z_2.$$

6**M** + 3**S** for DBL.

2001 Joy
$2(X_1 : Y$
$(Z_1 : X_1$
so can u

"Unified
helpful a
But need
2009 Be
Easily av

2008 His
$(X : Y :$
$: 2X$
6**M** + 6**S**
3**M** + 6**S**

or ADD.
other systems,
ing benefit
BL.

useful.
udry:
on Jacobians of
tic curves,
rization.

ally helpful:
urves
lcart–Kohel),
s, . . .

Hessian curves

Credited to Sylvester
by 1986 Chudnovsky–Chudnovsky:

$(X : Y : Z)$ represent $(X/Z, Y/Z)$
on $x^3 + y^3 + 1 = 3dxy$.

$12\mathbf{M}$ for ADD:

$$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot X_1Y_2,$$
$$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot Z_1X_2,$$
$$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot Y_1Z_2.$$

$6\mathbf{M} + 3\mathbf{S}$ for DBL.

2001 Joye–Quisqu
$2(X_1 : Y_1 : Z_1) =$
$(Z_1 : X_1 : Y_1) + ($Y
so can use ADD t

"Unified addition f
helpful against sid
But need to permu
2009 Bernstein–Ko
Easily avoid permu

2008 Hisil–Wong–
$(X : Y : Z : X^2 :$ Y
$: 2XY : 2XZ$
$6\mathbf{M} + 6\mathbf{S}$ for ADD
$3\mathbf{M} + 6\mathbf{S}$ for DBL

ems,

t

ans of

ul:

el),

<u>Hessian curves</u>

Credited to Sylvester
by 1986 Chudnovsky–Chudnovsky:

$(X : Y : Z)$ represent $(X/Z, Y/Z)$
on $x^3 + y^3 + 1 = 3dxy$.

12**M** for ADD:

$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot X_1Y_2,$
$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot Z_1X_2,$
$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot Y_1Z_2.$

6**M** + 3**S** for DBL.

2001 Joye–Quisquater:
$2(X_1 : Y_1 : Z_1) =$
$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X$
so can use ADD to double.

"Unified addition formulas,"
helpful against side channels
But need to permute inputs
2009 Bernstein–Kohel–Lang
Easily avoid permutation!

2008 Hisil–Wong–Carter–Daw
$(X : Y : Z : X^2 : Y^2 : Z^2$
$: 2XY : 2XZ : 2YZ).$
6**M** + 6**S** for ADD.
3**M** + 6**S** for DBL.

Hessian curves

Credited to Sylvester
by 1986 Chudnovsky–Chudnovsky:

$(X : Y : Z)$ represent $(X/Z, Y/Z)$
on $x^3 + y^3 + 1 = 3dxy$.

12**M** for ADD:

$$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot X_1Y_2,$$
$$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot Z_1X_2,$$
$$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot Y_1Z_2.$$

6**M** + 3**S** for DBL.

2001 Joye–Quisquater:
$2(X_1 : Y_1 : Z_1) =$
$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X_1)$
so can use ADD to double.

"Unified addition formulas,"
helpful against side channels.
But need to permute inputs.
2009 Bernstein–Kohel–Lange:
Easily avoid permutation!

2008 Hisil–Wong–Carter–Dawson:
$(X : Y : Z : X^2 : Y^2 : Z^2$
$: 2XY : 2XZ : 2YZ)$.
6**M** + 6**S** for ADD.
3**M** + 6**S** for DBL.

curves

to Sylvester

Chudnovsky–Chudnovsky:

Z) represent $(X/Z, Y/Z)$

$y^3 + 1 = 3dxy$.

ADD:

$X_2 \cdot Y_1 Z_2 - Z_1 Y_2 \cdot X_1 Y_2$,

$Z_2 \cdot X_1 Y_2 - Y_1 X_2 \cdot Z_1 X_2$,

$Y_2 \cdot Z_1 X_2 - X_1 Z_2 \cdot Y_1 Z_2$.

S for DBL.

2001 Joye–Quisquater:

$2(X_1 : Y_1 : Z_1) =$

$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X_1)$

so can use ADD to double.

"Unified addition formulas,"

helpful against side channels.

But need to permute inputs.

2009 Bernstein–Kohel–Lange:

Easily avoid permutation!

2008 Hisil–Wong–Carter–Dawson:

$(X : Y : Z : X^2 : Y^2 : Z^2$

$: 2XY : 2XZ : 2YZ)$.

$6\mathbf{M} + 6\mathbf{S}$ for ADD.

$3\mathbf{M} + 6\mathbf{S}$ for DBL.

$x^3 - y^3$

ter

sky–Chudnovsky:

ent $(X/Z, Y/Z)$

$3dxy$.

$- Z_1Y_2 \cdot X_1Y_2$,

$- Y_1X_2 \cdot Z_1X_2$,

$- X_1Z_2 \cdot Y_1Z_2$.

.

2001 Joye–Quisquater:
$2(X_1 : Y_1 : Z_1) =$
$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X_1)$
so can use ADD to double.

"Unified addition formulas,"
helpful against side channels.
But need to permute inputs.
2009 Bernstein–Kohel–Lange:
Easily avoid permutation!

2008 Hisil–Wong–Carter–Dawson:
$(X : Y : Z : X^2 : Y^2 : Z^2$
$: 2XY : 2XZ : 2YZ)$.
$6\mathbf{M} + 6\mathbf{S}$ for ADD.
$3\mathbf{M} + 6\mathbf{S}$ for DBL.

$x^3 - y^3 + 1 = 0.3$

ovsky:

$Y/Z)$

$X_1 Y_2$,

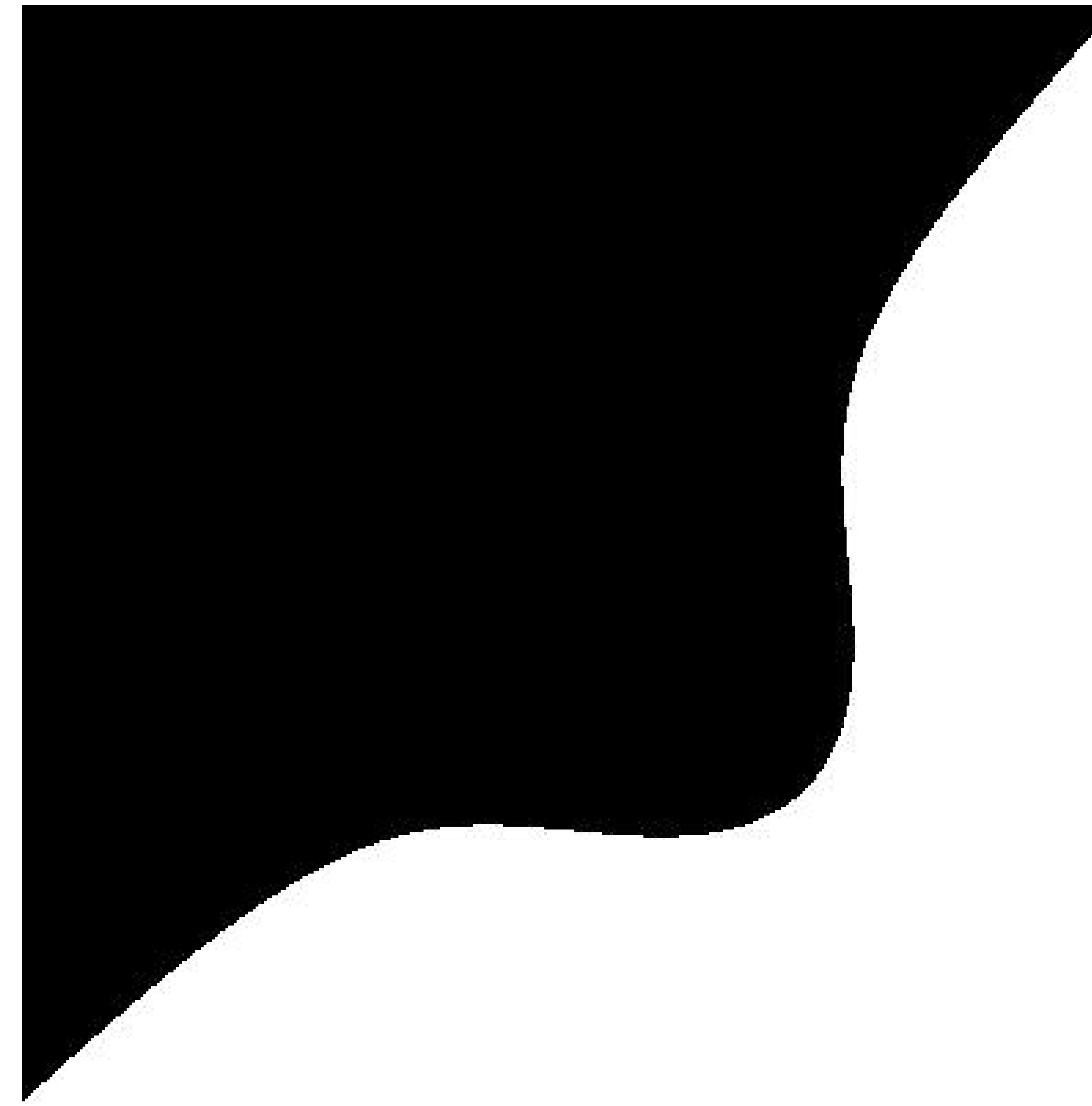
$Z_1 X_2$,

$Y_1 Z_2$.

2001 Joye–Quisquater:
$2(X_1 : Y_1 : Z_1) =$
$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X_1)$
so can use ADD to double.

"Unified addition formulas,"
helpful against side channels.
But need to permute inputs.
2009 Bernstein–Kohel–Lange:
Easily avoid permutation!

2008 Hisil–Wong–Carter–Dawson:
$(X : Y : Z : X^2 : Y^2 : Z^2$
$: 2XY : 2XZ : 2YZ)$.
$6\mathbf{M} + 6\mathbf{S}$ for ADD.
$3\mathbf{M} + 6\mathbf{S}$ for DBL.



$x^3 - y^3 + 1 = 0.3xy$

2001 Joye–Quisquater:

$2(X_1 : Y_1 : Z_1) =$

$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X_1)$

so can use ADD to double.

“Unified addition formulas,”

helpful against side channels.

But need to permute inputs.

2009 Bernstein–Kohel–Lange:

Easily avoid permutation!

2008 Hisil–Wong–Carter–Dawson:

$(X : Y : Z : X^2 : Y^2 : Z^2$

$\quad : 2XY : 2XZ : 2YZ)$.

$6\mathbf{M} + 6\mathbf{S}$ for ADD.

$3\mathbf{M} + 6\mathbf{S}$ for DBL.

$x^3 - y^3 + 1 = 0.3xy$

ye–Quisquater:

$_1 : Z_1) =$

$_1 : Y_1) + (Y_1 : Z_1 : X_1)$

se ADD to double.

addition formulas,"

gainst side channels.

d to permute inputs.

rnstein–Kohel–Lange:

void permutation!

sil–Wong–Carter–Dawson:

$Z : X^2 : Y^2 : Z^2$

$XY : 2XZ : 2YZ)$.

S for ADD.

S for DBL.

$x^3 - y^3 + 1 = 0.3xy$

ater:

$Y_1 : Z_1 : X_1)$

o double.

formulas,”

e channels.

ute inputs.

ohel–Lange:

utation!

Carter–Dawson:

$Y^2 : Z^2$

$: 2YZ)$.

).

.

$x^3 - y^3 + 1 = 0.3xy$

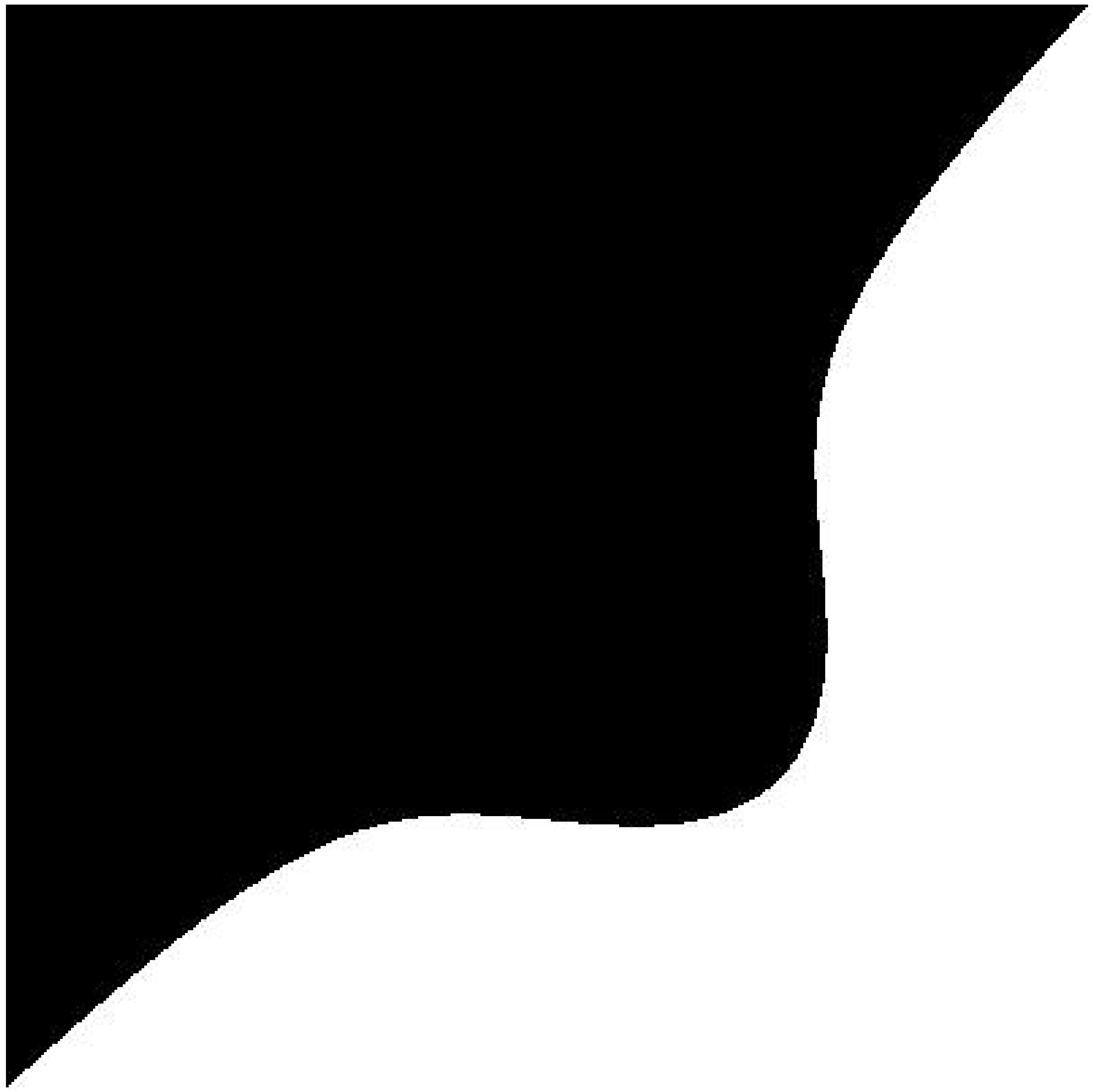
$_1)$

s.

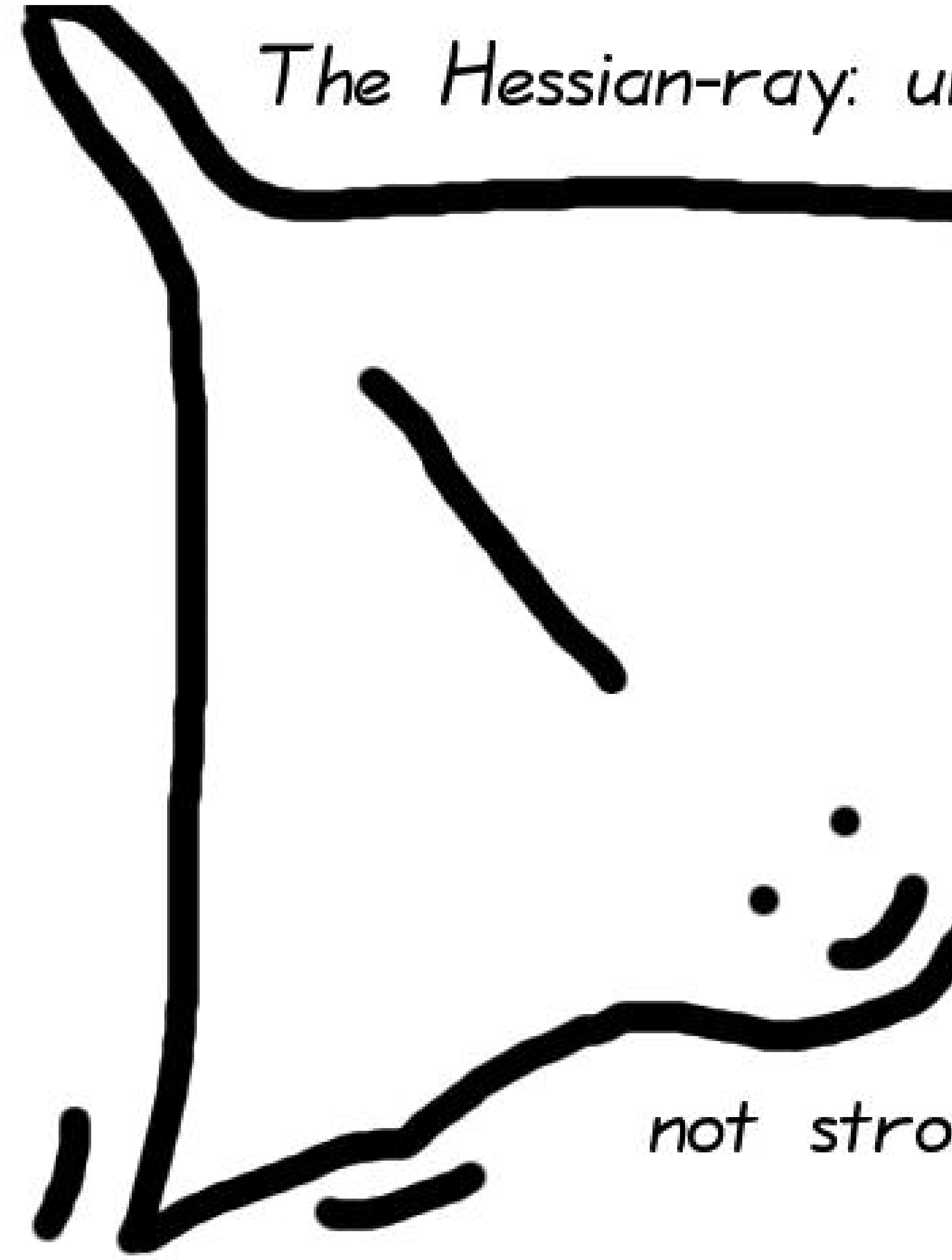
.

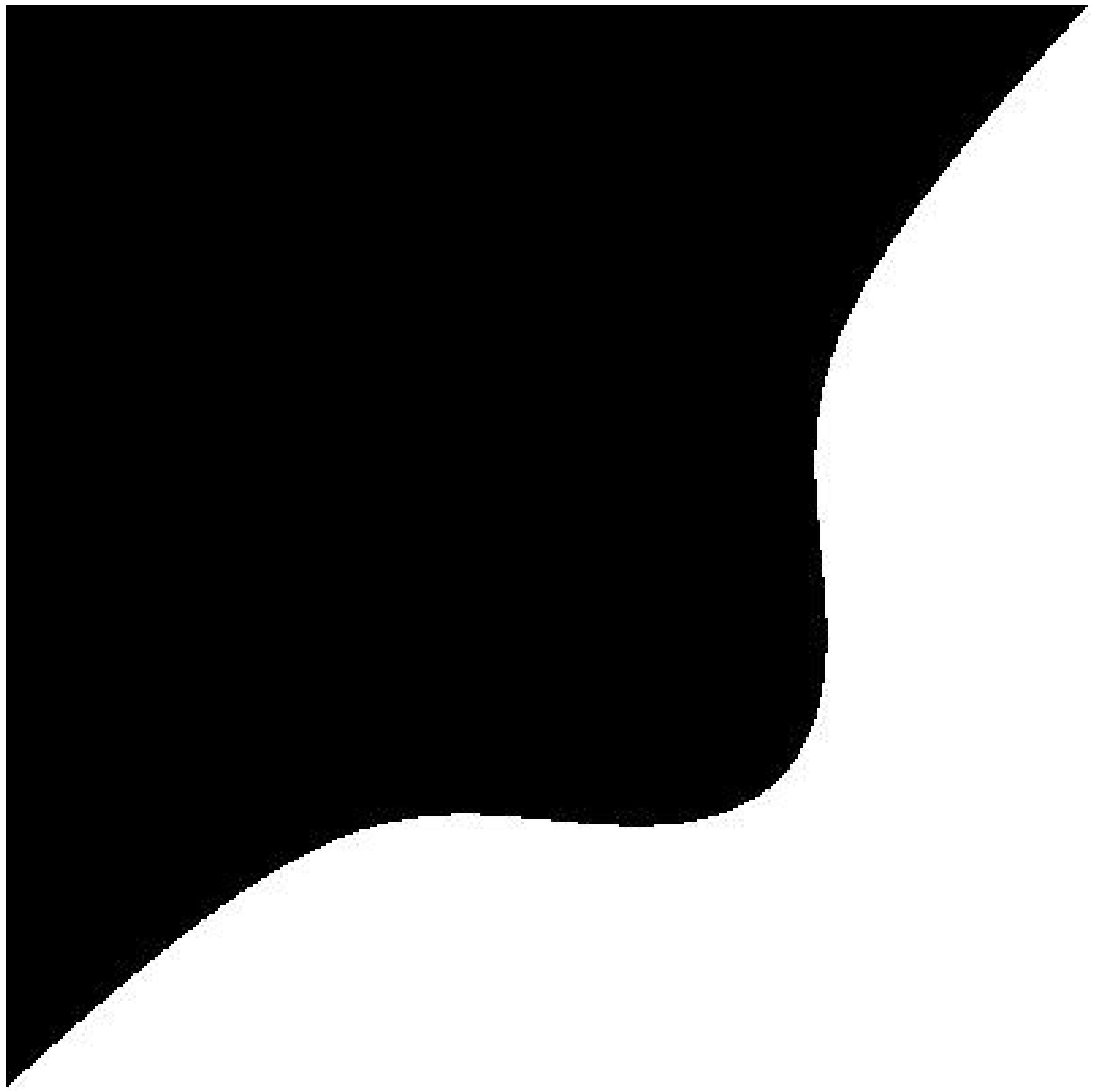
e:

wson:

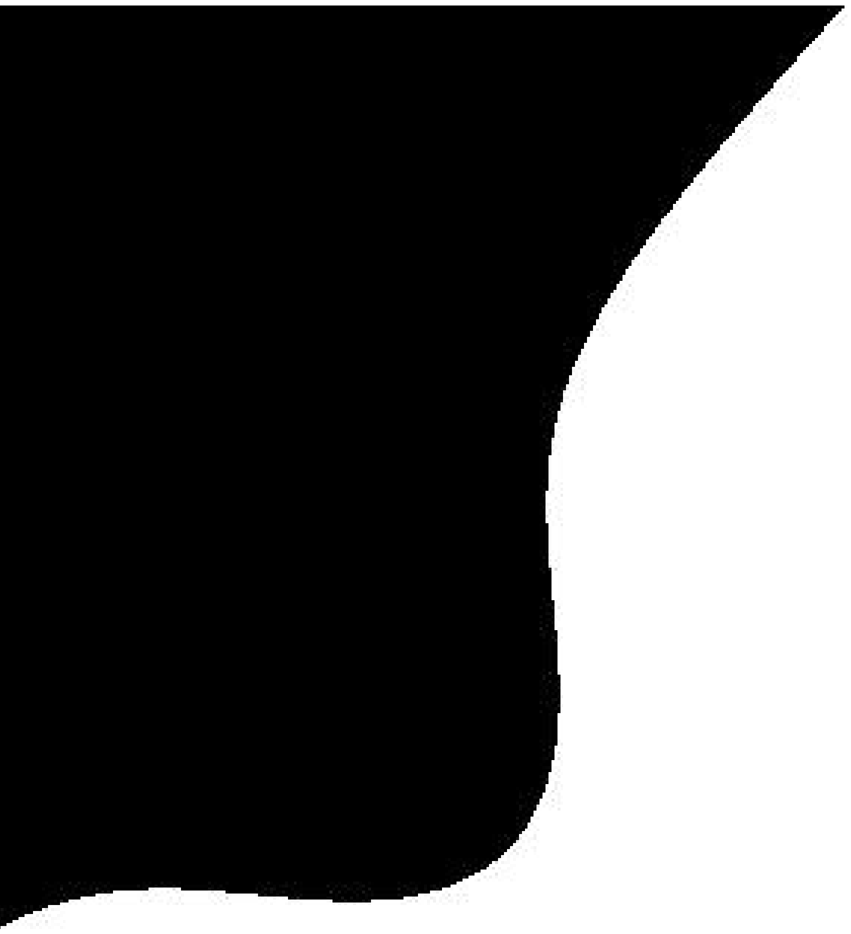


$x^3 - y^3 + 1 = 0.3xy$





$$x^3 - y^3 + 1 = 0.3xy$$



$+ 1 = 0.3xy$

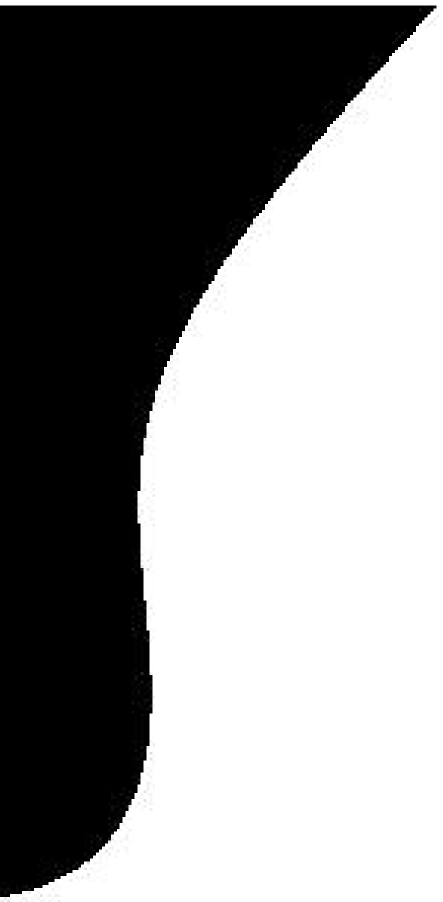
Jacobi in

1986 Ch

$(S : C :$
$(S/Z, C$
$s^2 + c^2 =$

$14\mathbf{M} + 2$
"Tremen
of being

$5\mathbf{M} + 3\mathbf{S}$
"Perhap
efficient
which do
coefficie



xy

Jacobi intersection

1986 Chudnovsky-

$(S : C : D : Z)$ rep
$(S/Z, C/Z, D/Z)$
$s^2 + c^2 = 1$, $as^2 +$

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ f
"Tremendous adva
of being strongly u

$5\mathbf{M} + 3\mathbf{S}$ for DBL
"Perhaps (?) . . . t
efficient duplicatio
which do not depe
coefficients of an e

Jacobi intersections

1986 Chudnovsky–Chudnovs

$(S : C : D : Z)$ represent
$(S/Z, C/Z, D/Z)$ on
$s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
"Tremendous advantage"
of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
"Perhaps (?) . . . the most
efficient duplication formulas
which do not depend on the
coefficients of an elliptic cur

Jacobi intersections

1986 Chudnovsky–Chudnovsky:

$(S : C : D : Z)$ represent $(S/Z, C/Z, D/Z)$ on $s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
"Tremendous advantage" of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
"Perhaps (?) . . . the most efficient duplication formulas which do not depend on the coefficients of an elliptic curve."

Jacobi intersections

1986 Chudnovsky–Chudnovsky:

$(S : C : D : Z)$ represent
$(S/Z, C/Z, D/Z)$ on
$s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
"Tremendous advantage"
of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
"Perhaps (?) . . . the most
efficient duplication formulas
which do not depend on the
coefficients of an elliptic curve."

2001 Lia

$13\mathbf{M} + 2$

$4\mathbf{M} + 3\mathbf{S}$

2007 Be

$3\mathbf{M} + 4\mathbf{S}$

2008 His

$13\mathbf{M} + 1$

$2\mathbf{M} + 5\mathbf{S}$

Also $(S$

$11\mathbf{M} + 1$

$2\mathbf{M} + 5\mathbf{S}$

Jacobi intersections

1986 Chudnovsky–Chudnovsky:

$(S : C : D : Z)$ represent $(S/Z, C/Z, D/Z)$ on $s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
“Tremendous advantage” of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
“Perhaps (?) . . . the most efficient duplication formulas which do not depend on the coefficients of an elliptic curve.”

2001 Liardet–Sma

$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ fo

$4\mathbf{M} + 3\mathbf{S}$ for DBL

2007 Bernstein–La

$3\mathbf{M} + 4\mathbf{S}$ for DBL

2008 Hisil–Wong–

$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ fo

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for

Also $(S : C : D : Z$

$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ fo

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for

Jacobi intersections

1986 Chudnovsky–Chudnovsky:

$(S : C : D : Z)$ represent $(S/Z, C/Z, D/Z)$ on $s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
"Tremendous advantage" of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
"Perhaps (?) . . . the most efficient duplication formulas which do not depend on the coefficients of an elliptic curve."

2001 Liardet–Smart:
$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
$4\mathbf{M} + 3\mathbf{S}$ for DBL.

2007 Bernstein–Lange:
$3\mathbf{M} + 4\mathbf{S}$ for DBL.

2008 Hisil–Wong–Carter–Da
$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.
$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.
Also $(S : C : D : Z : SC : D$
$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.
$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Jacobi intersections

1986 Chudnovsky–Chudnovsky:

$(S : C : D : Z)$ represent
$(S/Z, C/Z, D/Z)$ on
$s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
"Tremendous advantage"
of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
"Perhaps (?) . . . the most
efficient duplication formulas
which do not depend on the
coefficients of an elliptic curve."

2001 Liardet–Smart:
$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
$4\mathbf{M} + 3\mathbf{S}$ for DBL.

2007 Bernstein–Lange:
$3\mathbf{M} + 4\mathbf{S}$ for DBL.

2008 Hisil–Wong–Carter–Dawson:
$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.
$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.
Also $(S : C : D : Z : SC : DZ)$:
$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.
$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

ntersections

udnovsky–Chudnovsky:

$D : Z)$ represent

$/Z, D/Z)$ on

$= 1$, $as^2 + d^2 = 1$.

$2\mathbf{S} + 1\mathbf{D}$ for ADD.

ndous advantage”

strongly unified.

S for DBL.

s (?) . . . the most

duplication formulas

o not depend on the

nts of an elliptic curve.”

2001 Liardet–Smart:

$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.

$4\mathbf{M} + 3\mathbf{S}$ for DBL.

2007 Bernstein–Lange:

$3\mathbf{M} + 4\mathbf{S}$ for DBL.

2008 Hisil–Wong–Carter–Dawson:

$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Also $(S : C : D : Z : SC : DZ)$:

$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Jacobi q

$(X:Y:Z)$

on $y^2 =$

1986 Ch

$3\mathbf{M} + 6$

Slow AD

2002 Bil

New ch

$10\mathbf{M} +$

strongly

2007 Be

$1\mathbf{M} + 9$

ns

-Chudnovsky:

present

on

$- d^2 = 1.$

or ADD.

antage"

unified.

.

the most

on formulas

end on the

elliptic curve."

2001 Liardet–Smart:

$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.

$4\mathbf{M} + 3\mathbf{S}$ for DBL.

2007 Bernstein–Lange:

$3\mathbf{M} + 4\mathbf{S}$ for DBL.

2008 Hisil–Wong–Carter–Dawson:

$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Also $(S : C : D : Z : SC : DZ)$:

$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Jacobi quartics

$(X{:}Y{:}Z)$ represent

on $y^2 = x^4 + 2ax$

1986 Chudnovsky–

$3\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ fo

Slow ADD.

2002 Billet–Joye:

New choice of neu

$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ f

strongly unified.

2007 Bernstein–La

$1\mathbf{M} + 9\mathbf{S} + 1\mathbf{D}$ fo sky:

s

e

rve."

2001 Liardet–Smart:
13**M** + 2**S** + 1**D** for ADD.
4**M** + 3**S** for DBL.

2007 Bernstein–Lange:
3**M** + 4**S** for DBL.

2008 Hisil–Wong–Carter–Dawson:
13**M** + 1**S** + 2**D** for ADD.
2**M** + 5**S** + 1**D** for DBL.
Also $(S : C : D : Z : SC : DZ)$:
11**M** + 1**S** + 2**D** for ADD.
2**M** + 5**S** + 1**D** for DBL.

Jacobi quartics

$(X{:}Y{:}Z)$ represent $(X/Z, Y/$
on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovs
3**M** + 6**S** + 2**D** for DBL.
Slow ADD.

2002 Billet–Joye:
New choice of neutral eleme
10**M** + 3**S** + 1**D** for ADD,
strongly unified.

2007 Bernstein–Lange:
1**M** + 9**S** + 1**D** for DBL.

2001 Liardet–Smart:

$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.

$4\mathbf{M} + 3\mathbf{S}$ for DBL.

2007 Bernstein–Lange:

$3\mathbf{M} + 4\mathbf{S}$ for DBL.

2008 Hisil–Wong–Carter–Dawson:

$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Also $(S : C : D : Z : SC : DZ)$:

$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

<u>Jacobi quartics</u>

$(X{:}Y{:}Z)$ represent $(X/Z, Y/Z^2)$ on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovsky:

$3\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.

Slow ADD.

2002 Billet–Joye:

New choice of neutral element.

$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ for ADD, strongly unified.

2007 Bernstein–Lange:

$1\mathbf{M} + 9\mathbf{S} + 1\mathbf{D}$ for DBL.

ardet–Smart:

$2\mathbf{S} + 1\mathbf{D}$ for ADD.

$\mathbf{S}$ for DBL.

rnstein–Lange:

$\mathbf{S}$ for DBL.

sil–Wong–Carter–Dawson:

$1\mathbf{S} + 2\mathbf{D}$ for ADD.

$\mathbf{S} + 1\mathbf{D}$ for DBL.

$: C : D : Z : SC : DZ)$:

$1\mathbf{S} + 2\mathbf{D}$ for ADD.

$\mathbf{S} + 1\mathbf{D}$ for DBL.

Jacobi quartics

$(X{:}Y{:}Z)$ represent $(X/Z, Y/Z^2)$ on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovsky:
$3\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.
Slow ADD.

2002 Billet–Joye:
New choice of neutral element.
$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ for ADD,
strongly unified.

2007 Bernstein–Lange:
$1\mathbf{M} + 9\mathbf{S} + 1\mathbf{D}$ for DBL.

2007 His

$2\mathbf{M} + 6\mathbf{S}$

2007 Fe

$2\mathbf{M} + 6\mathbf{S}$

$1\mathbf{M} + 7\mathbf{S}$

on curve

More sp

2007 His

2008 His

use $(X :$

or $(X : Y$

Can com

Competi

rt:
or ADD.
.
ange:
.
Carter–Dawson:
or ADD.
r DBL.
$Z : SC : DZ$):
or ADD.
r DBL.

Jacobi quartics

$(X{:}Y{:}Z)$ represent $(X/Z, Y/Z^2)$ on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovsky:
$3\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.
Slow ADD.

2002 Billet–Joye:
New choice of neutral element.
$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ for ADD,
strongly unified.

2007 Bernstein–Lange:
$1\mathbf{M} + 9\mathbf{S} + 1\mathbf{D}$ for DBL.

2007 Hisil–Carter–
$2\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ fo

2007 Feng–Wu:
$2\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ fo
$1\mathbf{M} + 7\mathbf{S} + 3\mathbf{D}$ fo
on curves chosen w

More speedups: 20
2007 Hisil–Carter–
2008 Hisil–Wong–C
use $(X : Y : Z : X$
or $(X : Y : Z : X^2$
Can combine with
Competitive with

wson:

Z):

Jacobi quartics

$(X{:}Y{:}Z)$ represent $(X/Z, Y/Z^2)$
on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovsky:
3**M** + 6**S** + 2**D** for DBL.
Slow ADD.

2002 Billet–Joye:
New choice of neutral element.
10**M** + 3**S** + 1**D** for ADD,
strongly unified.

2007 Bernstein–Lange:
1**M** + 9**S** + 1**D** for DBL.

2007 Hisil–Carter–Dawson:
2**M** + 6**S** + 2**D** for DBL.

2007 Feng–Wu:
2**M** + 6**S** + 1**D** for DBL.
1**M** + 7**S** + 3**D** for DBL
on curves chosen with $a^2 +$

More speedups: 2007 Duque
2007 Hisil–Carter–Dawson,
2008 Hisil–Wong–Carter–Da
use $(X : Y : Z : X^2 : Z^2)$
or $(X : Y : Z : X^2 : Z^2 : 2X$
Can combine with Feng–Wu
Competitive with Edwards!

Jacobi quartics

$(X{:}Y{:}Z)$ represent $(X/Z, Y/Z^2)$ on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovsky:
$3\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.
Slow ADD.

2002 Billet–Joye:
New choice of neutral element.
$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ for ADD,
strongly unified.

2007 Bernstein–Lange:
$1\mathbf{M} + 9\mathbf{S} + 1\mathbf{D}$ for DBL.

2007 Hisil–Carter–Dawson:
$2\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.

2007 Feng–Wu:
$2\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ for DBL.
$1\mathbf{M} + 7\mathbf{S} + 3\mathbf{D}$ for DBL
on curves chosen with $a^2 + c^2 = 1$.

More speedups: 2007 Duquesne,
2007 Hisil–Carter–Dawson,
2008 Hisil–Wong–Carter–Dawson:
use $(X : Y : Z : X^2 : Z^2)$
or $(X : Y : Z : X^2 : Z^2 : 2XZ)$.
Can combine with Feng–Wu.
Competitive with Edwards!

uartics

) represent $(X/Z, Y/Z^2)$

$x^4 + 2ax^2 + 1$.

udnovsky–Chudnovsky:

S + 2**D** for DBL.

DD.

let–Joye:

oice of neutral element.

3**S** + 1**D** for ADD,

unified.

rnstein–Lange:

S + 1**D** for DBL.

2007 Hisil–Carter–Dawson:

2**M** + 6**S** + 2**D** for DBL.

2007 Feng–Wu:

2**M** + 6**S** + 1**D** for DBL.

1**M** + 7**S** + 3**D** for DBL

on curves chosen with $a^2 + c^2 = 1$.

More speedups: 2007 Duquesne,

2007 Hisil–Carter–Dawson,

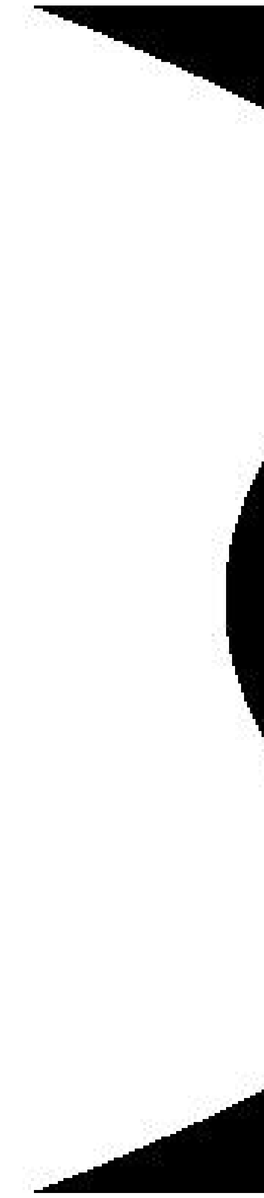
2008 Hisil–Wong–Carter–Dawson:

use $(X : Y : Z : X^2 : Z^2)$

or $(X : Y : Z : X^2 : Z^2 : 2XZ)$.

Can combine with Feng–Wu.

Competitive with Edwards!



$x^2 = y^4$

t $(X/Z, Y/Z^2)$

$^2 + 1$.

–Chudnovsky:

r DBL.

utral element.

or ADD,

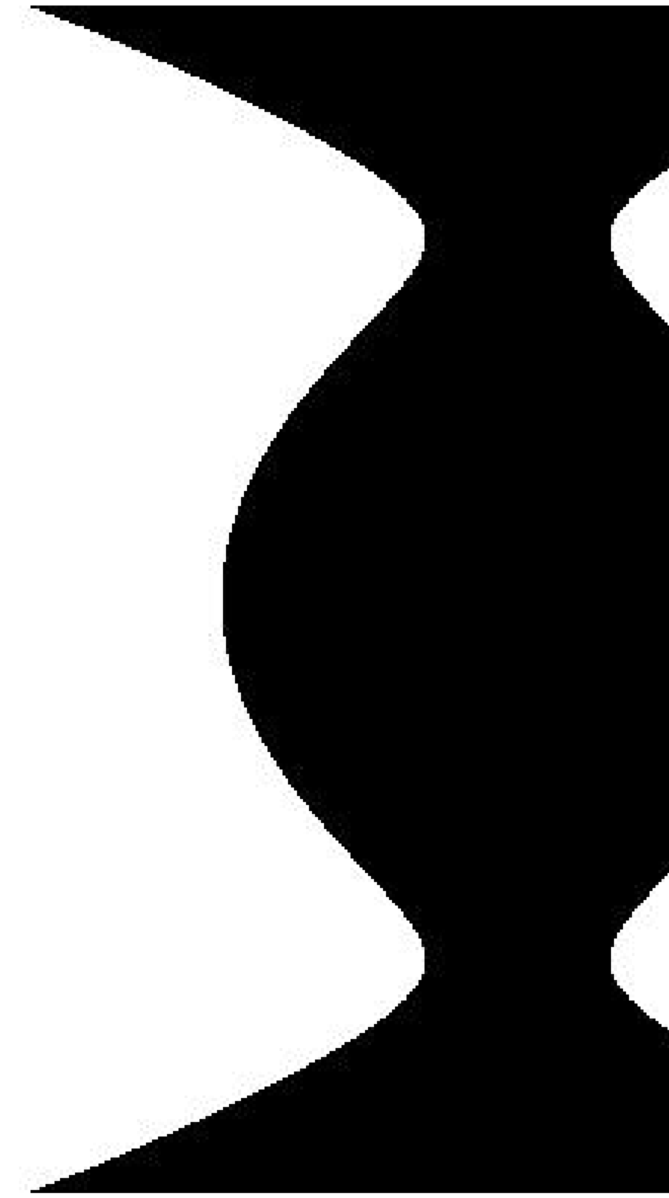
ange:

r DBL.

2007 Hisil–Carter–Dawson:
$2\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.

2007 Feng–Wu:
$2\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ for DBL.
$1\mathbf{M} + 7\mathbf{S} + 3\mathbf{D}$ for DBL
on curves chosen with $a^2 + c^2 = 1$.

More speedups: 2007 Duquesne,
2007 Hisil–Carter–Dawson,
2008 Hisil–Wong–Carter–Dawson:
use $(X : Y : Z : X^2 : Z^2)$
or $(X : Y : Z : X^2 : Z^2 : 2XZ)$.
Can combine with Feng–Wu.
Competitive with Edwards!



$x^2 = y^4 - 1.9y^2 +$

$/Z^2)$

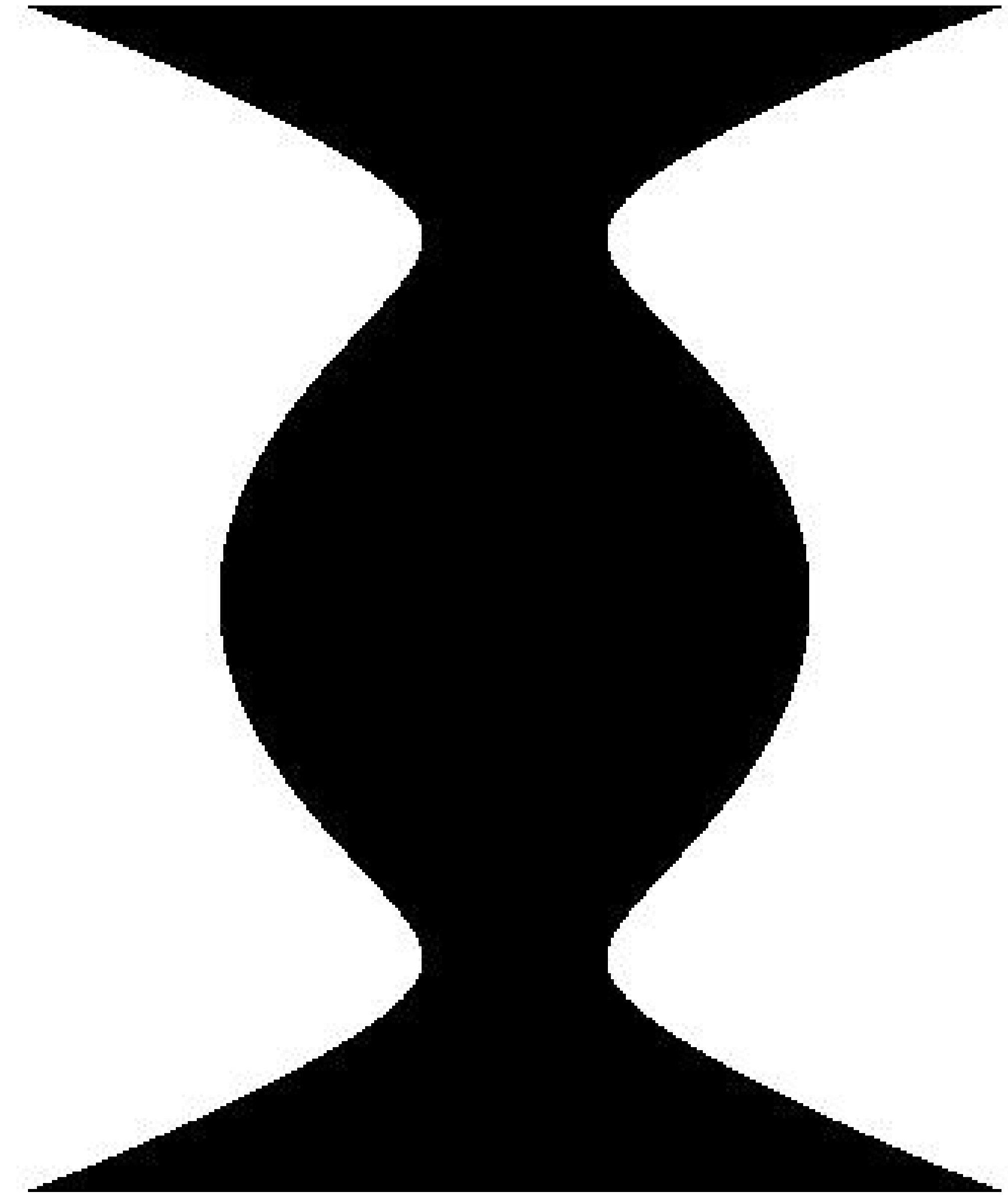
sky:

ent.

2007 Hisil–Carter–Dawson:
$2\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.

2007 Feng–Wu:
$2\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ for DBL.
$1\mathbf{M} + 7\mathbf{S} + 3\mathbf{D}$ for DBL
on curves chosen with $a^2 + c^2 = 1$.

More speedups: 2007 Duquesne, 2007 Hisil–Carter–Dawson, 2008 Hisil–Wong–Carter–Dawson:
use $(X : Y : Z : X^2 : Z^2)$
or $(X : Y : Z : X^2 : Z^2 : 2XZ)$.
Can combine with Feng–Wu.
Competitive with Edwards!



$x^2 = y^4 - 1.9y^2 + 1$

2007 Hisil–Carter–Dawson:

$2\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.

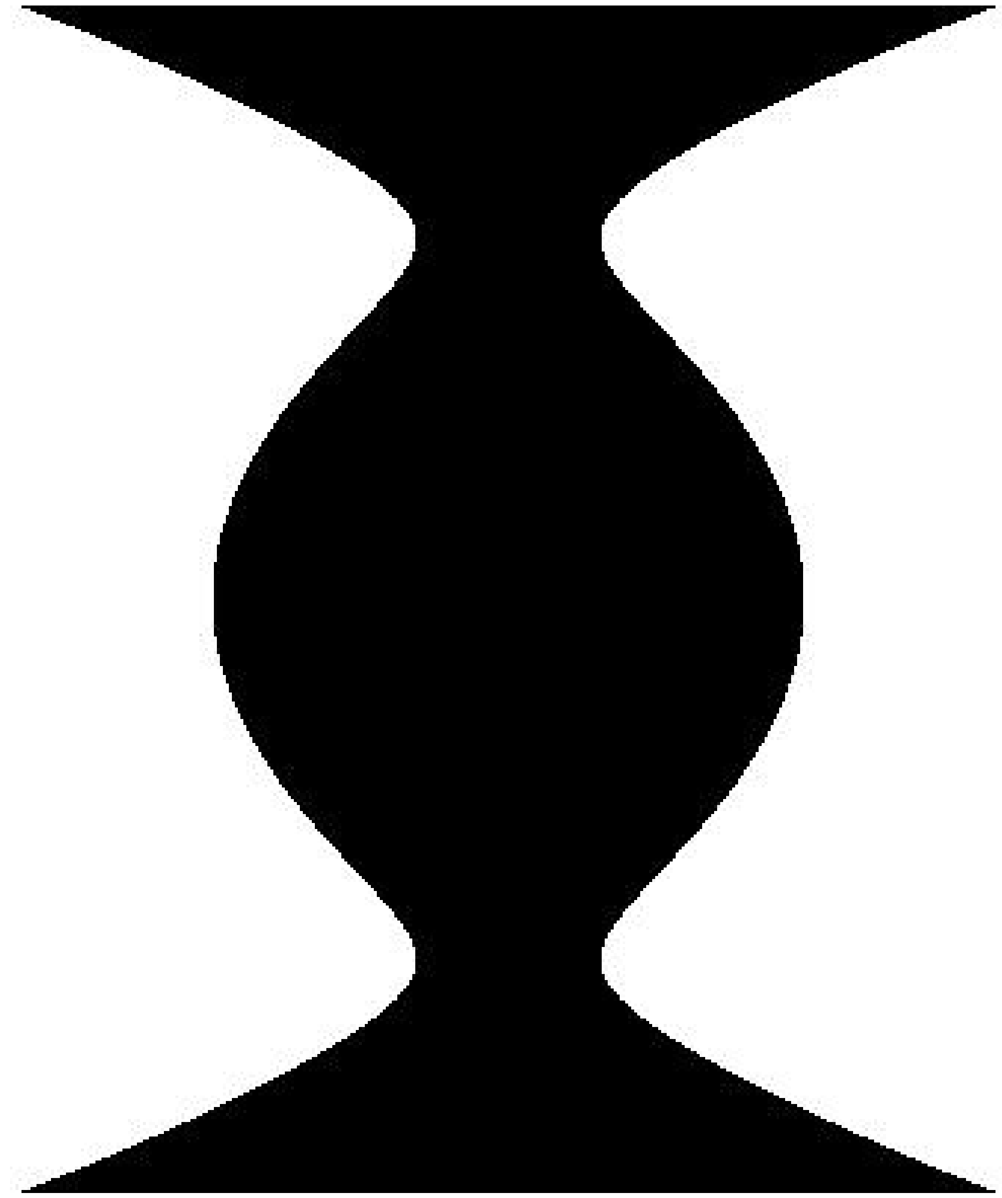
2007 Feng–Wu:

$2\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ for DBL.

$1\mathbf{M} + 7\mathbf{S} + 3\mathbf{D}$ for DBL

on curves chosen with $a^2+c^2 = 1$.

More speedups: 2007 Duquesne, 2007 Hisil–Carter–Dawson, 2008 Hisil–Wong–Carter–Dawson: use $(X : Y : Z : X^2 : Z^2)$ or $(X : Y : Z : X^2 : Z^2 : 2XZ)$. Can combine with Feng–Wu. Competitive with Edwards!



$x^2 = y^4 - 1.9y^2 + 1$

sil–Carter–Dawson:

S $+ 2$**D** for DBL.

ng–Wu:

S $+ 1$**D** for DBL.

S $+ 3$**D** for DBL

es chosen with $a^2 + c^2 = 1$.

eedups: 2007 Duquesne,

sil–Carter–Dawson,

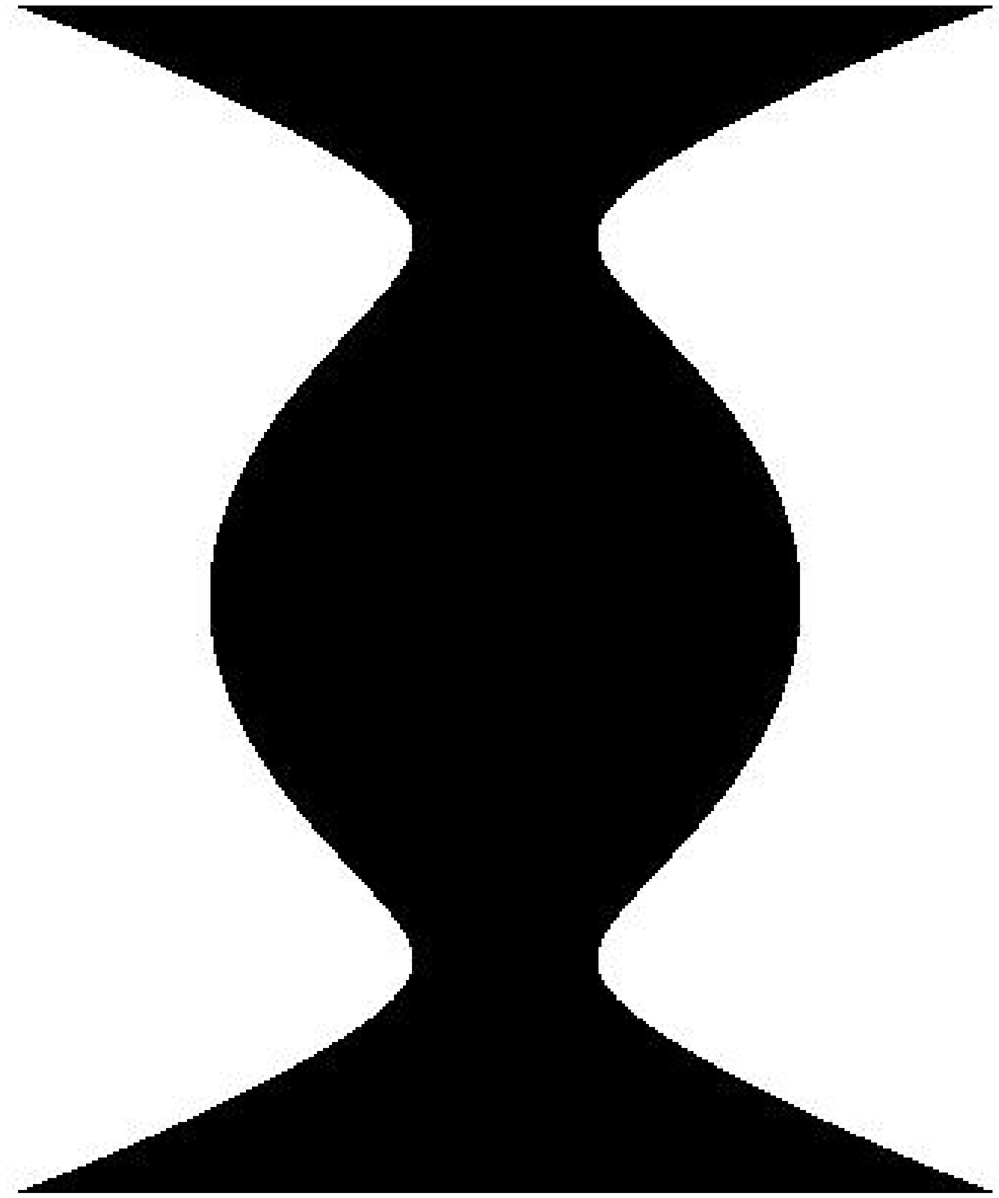
sil–Wong–Carter–Dawson:

$Y : Z : X^2 : Z^2)$

$Y : Z : X^2 : Z^2 : 2XZ)$.

nbine with Feng–Wu.

itive with Edwards!



$x^2 = y^4 - 1.9y^2 + 1$

The Jac

extended

XXYZZR

giant sq

-Dawson:

r DBL.

r DBL.

r DBL

with $a^2+c^2=1$.

007 Duquesne,

-Dawson,

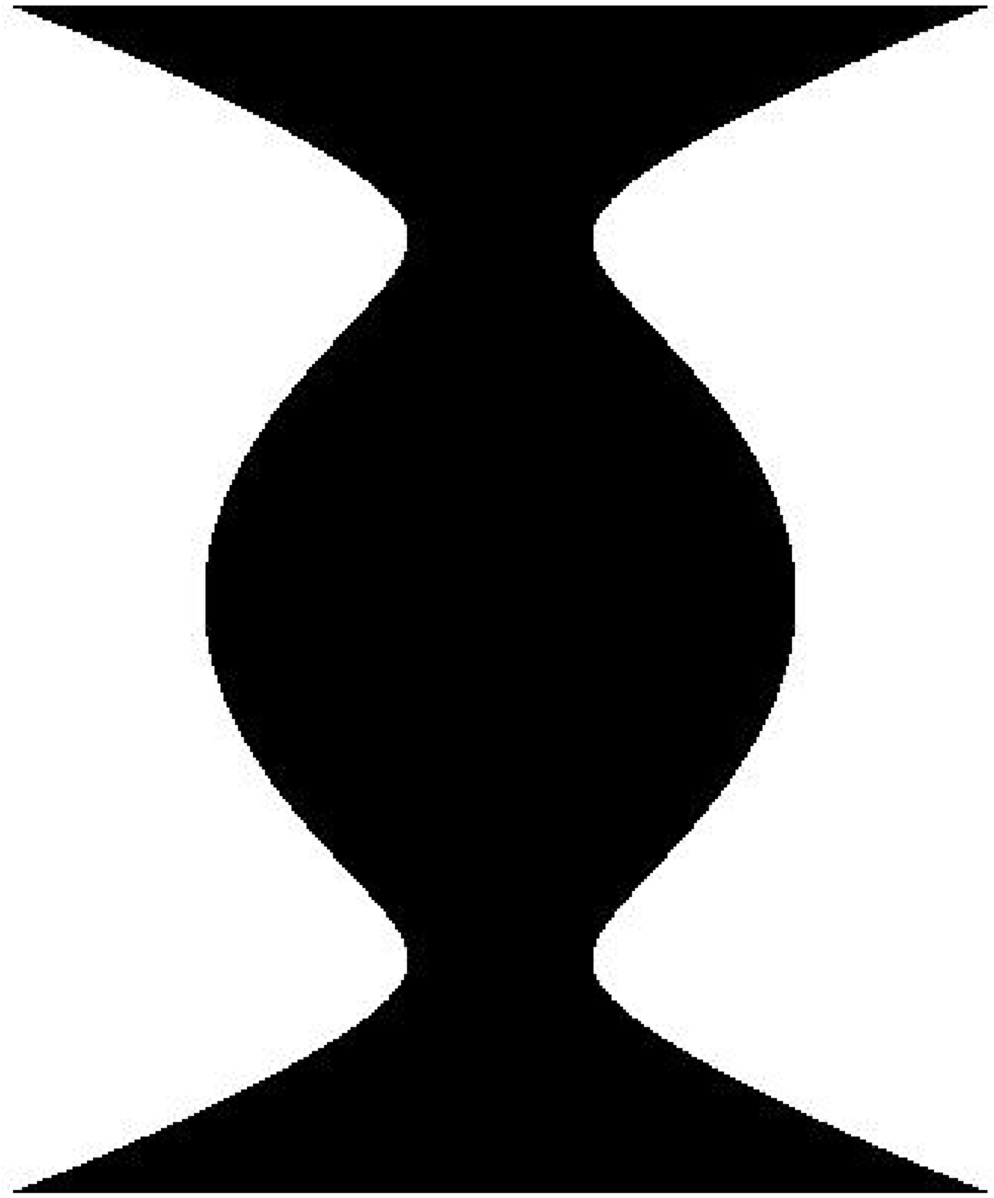
Carter–Dawson:

$^2 : Z^2)$

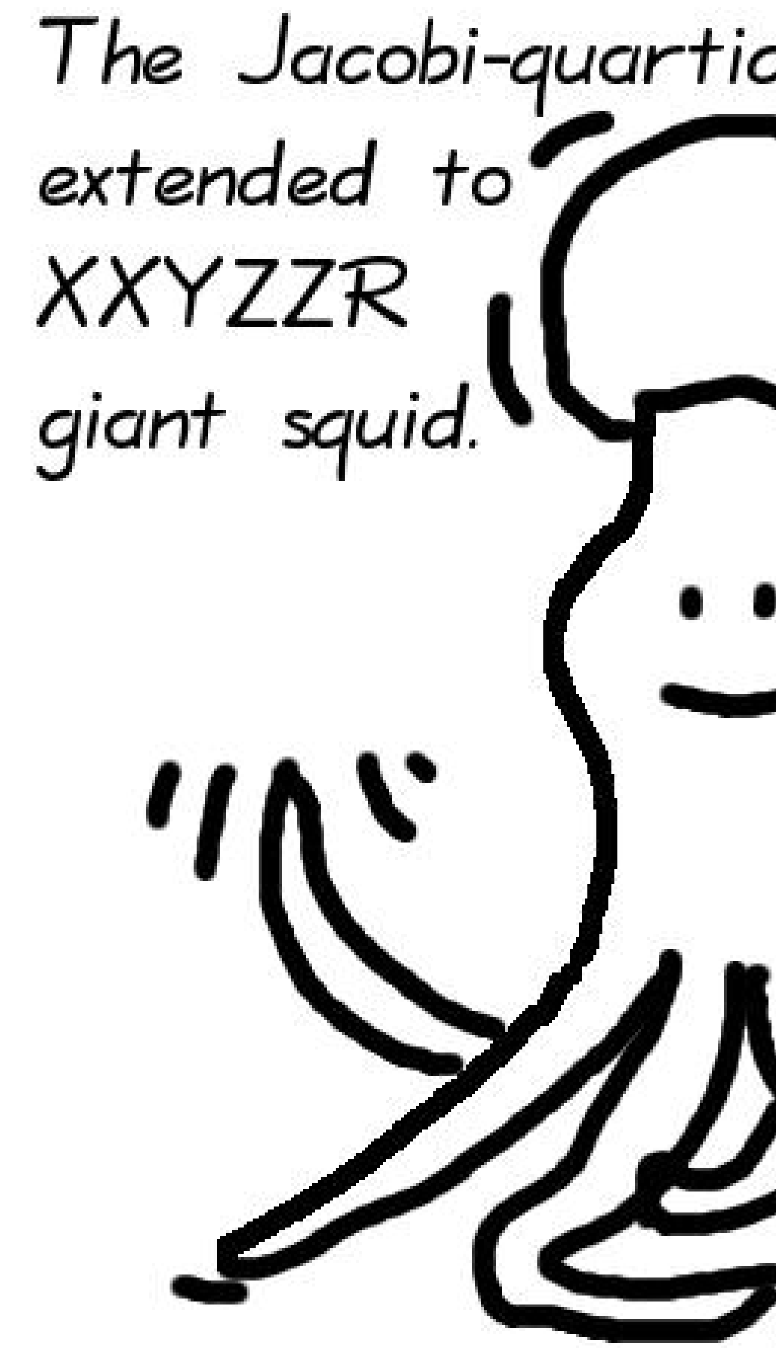
$: Z^2 : 2XZ)$.

Feng–Wu.

Edwards!



$x^2 = y^4 - 1.9y^2 + 1$



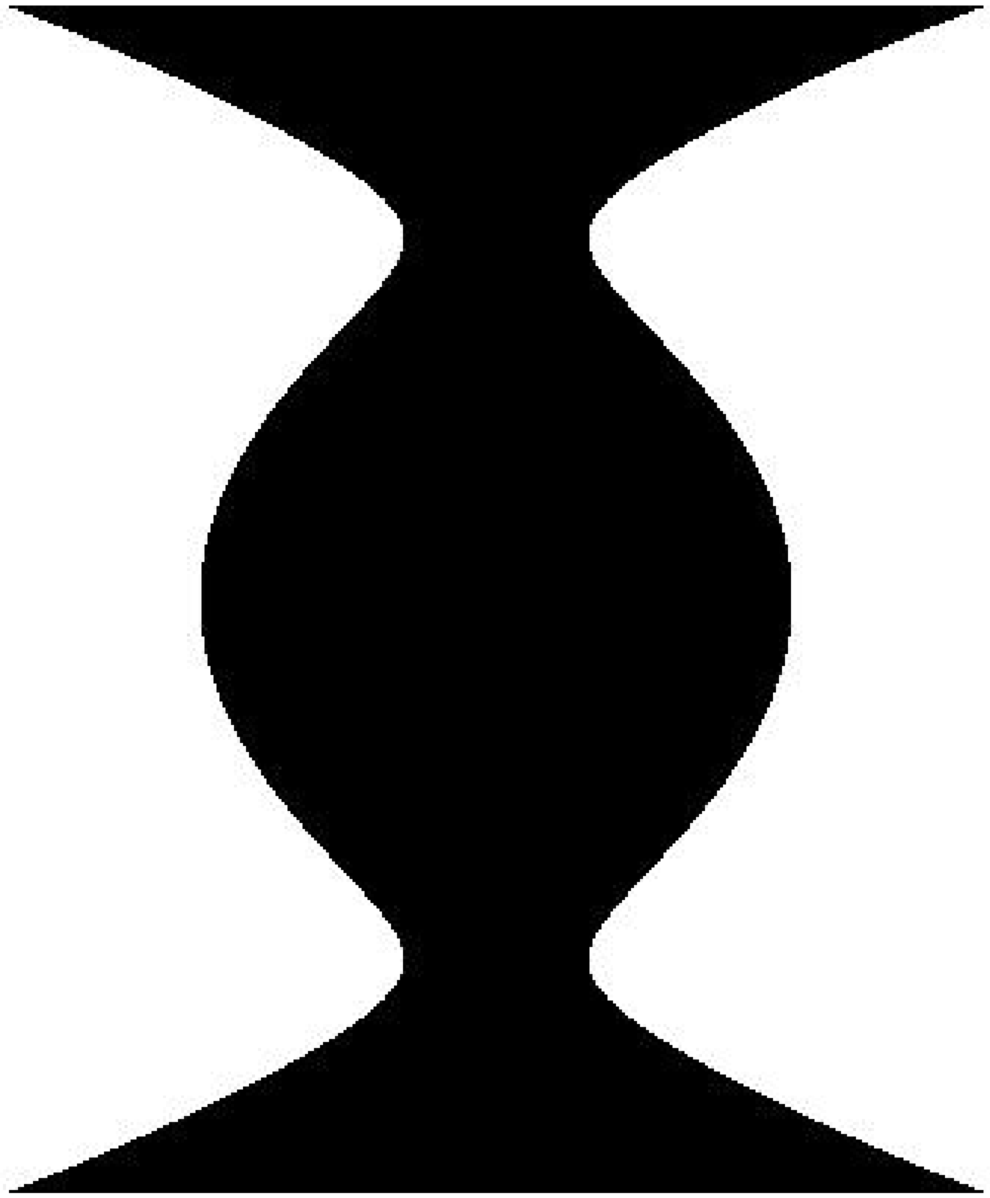
$c^2 = 1$.

esne,

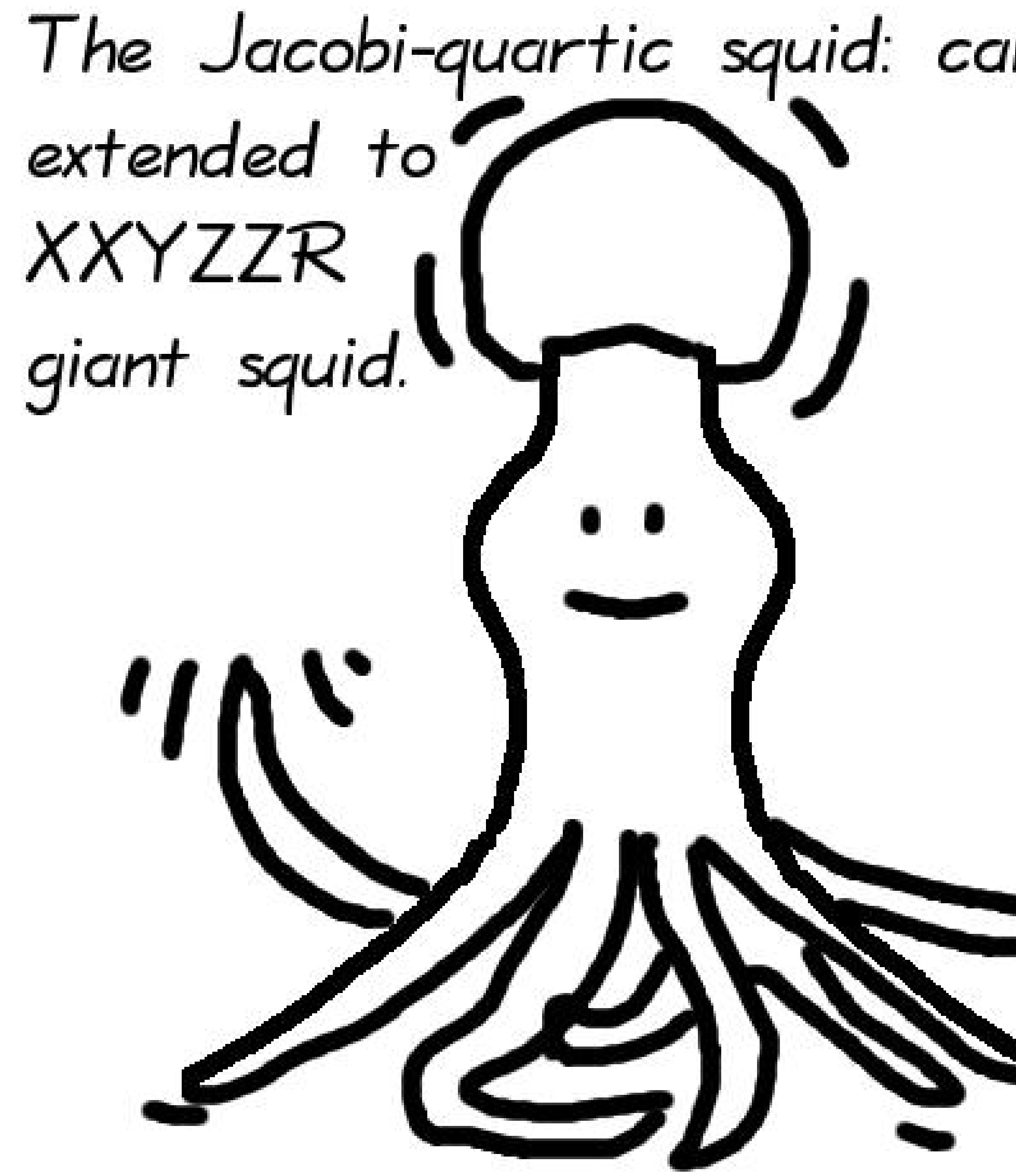
wson:

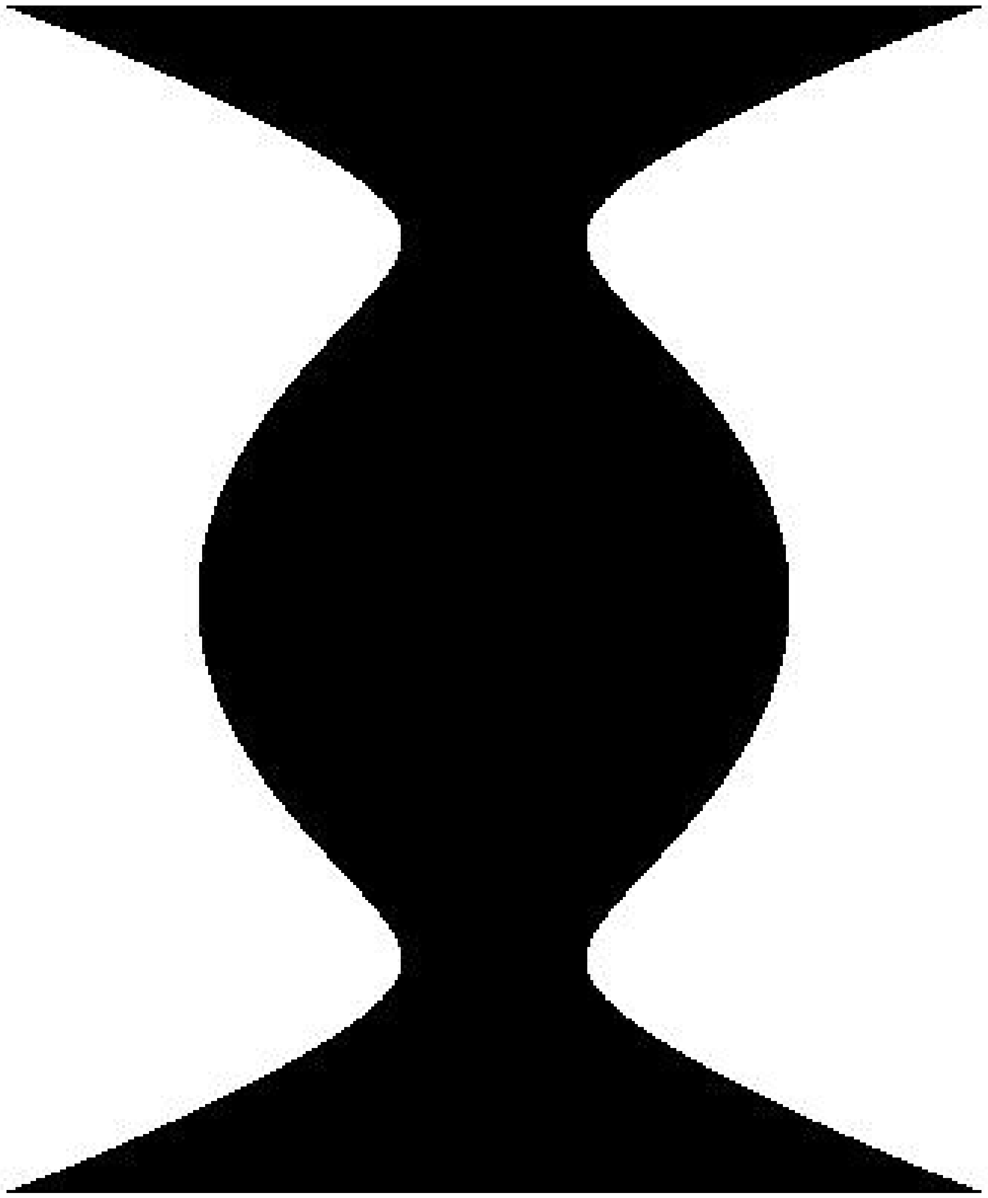
Z).

n.



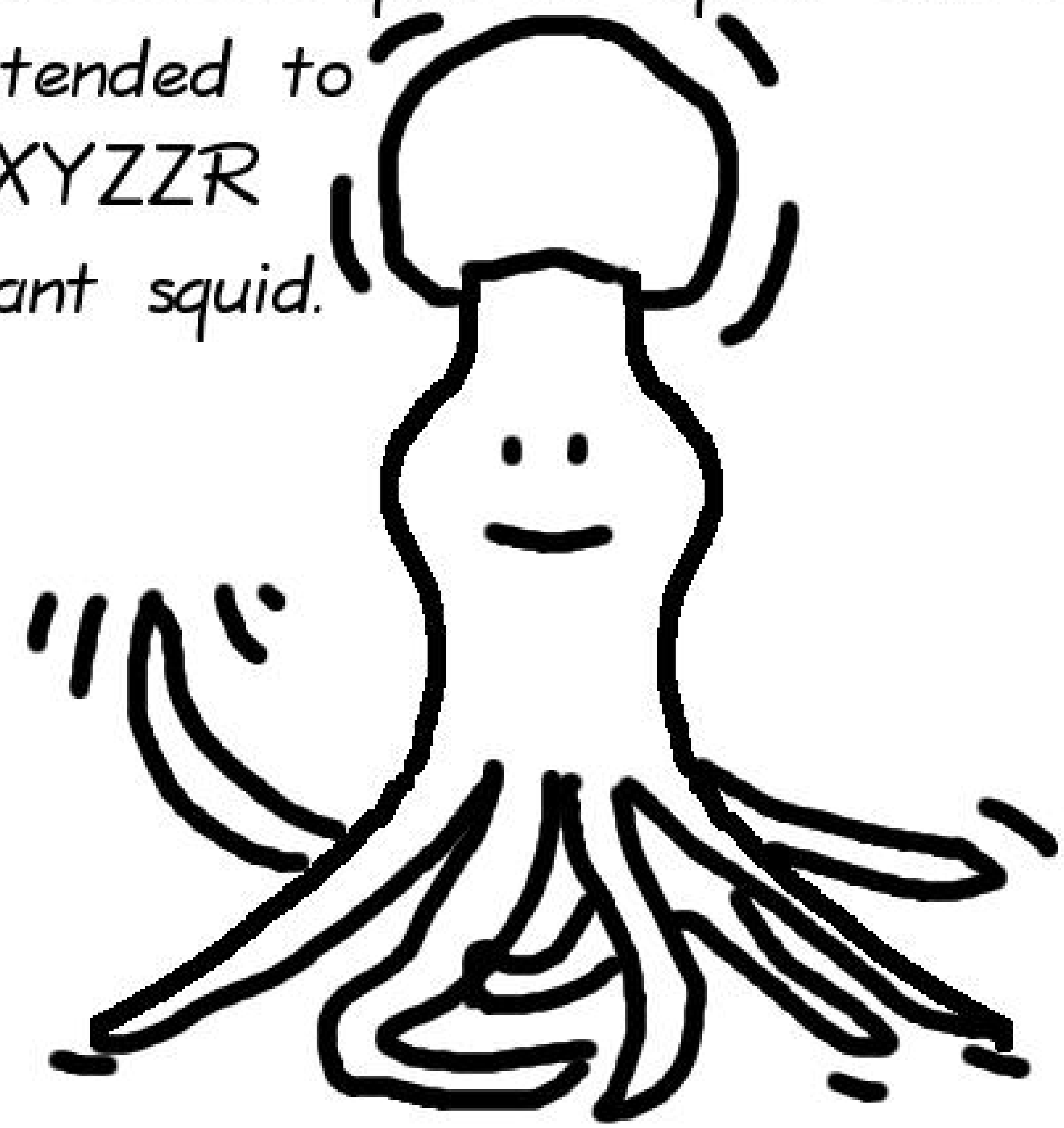
$x^2 = y^4 - 1.9y^2 + 1$

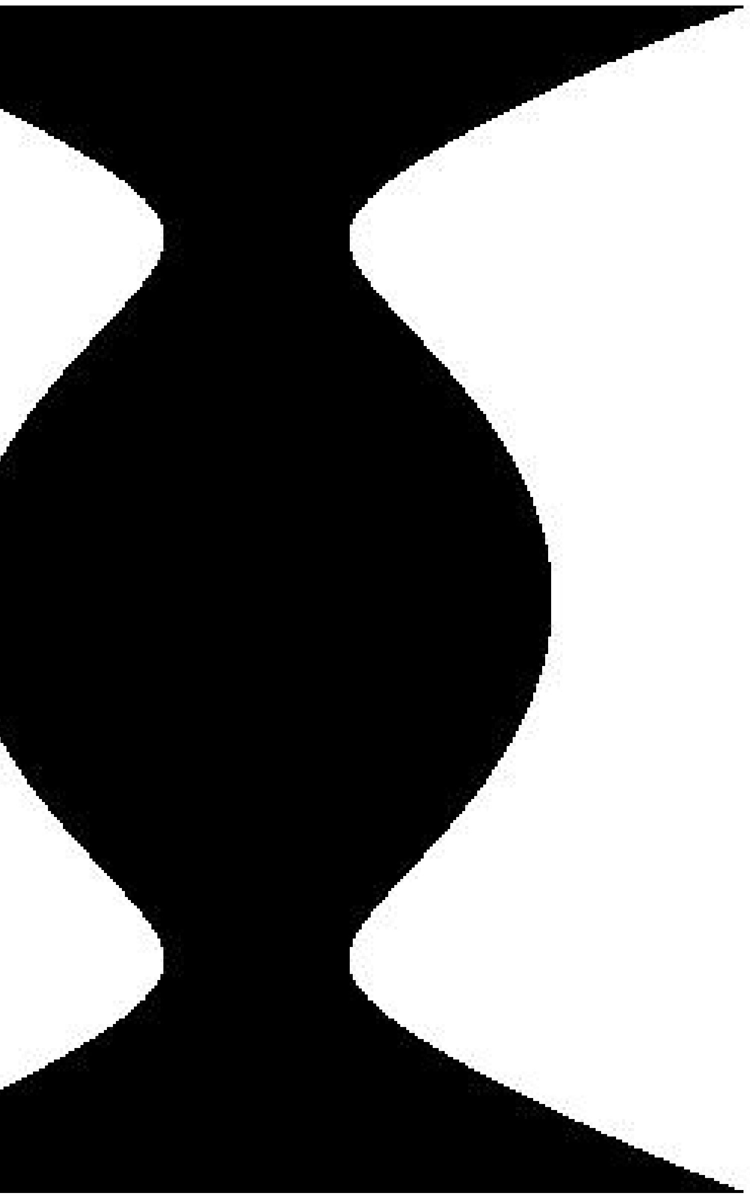




$$x^2 = y^4 - 1.9y^2 + 1$$

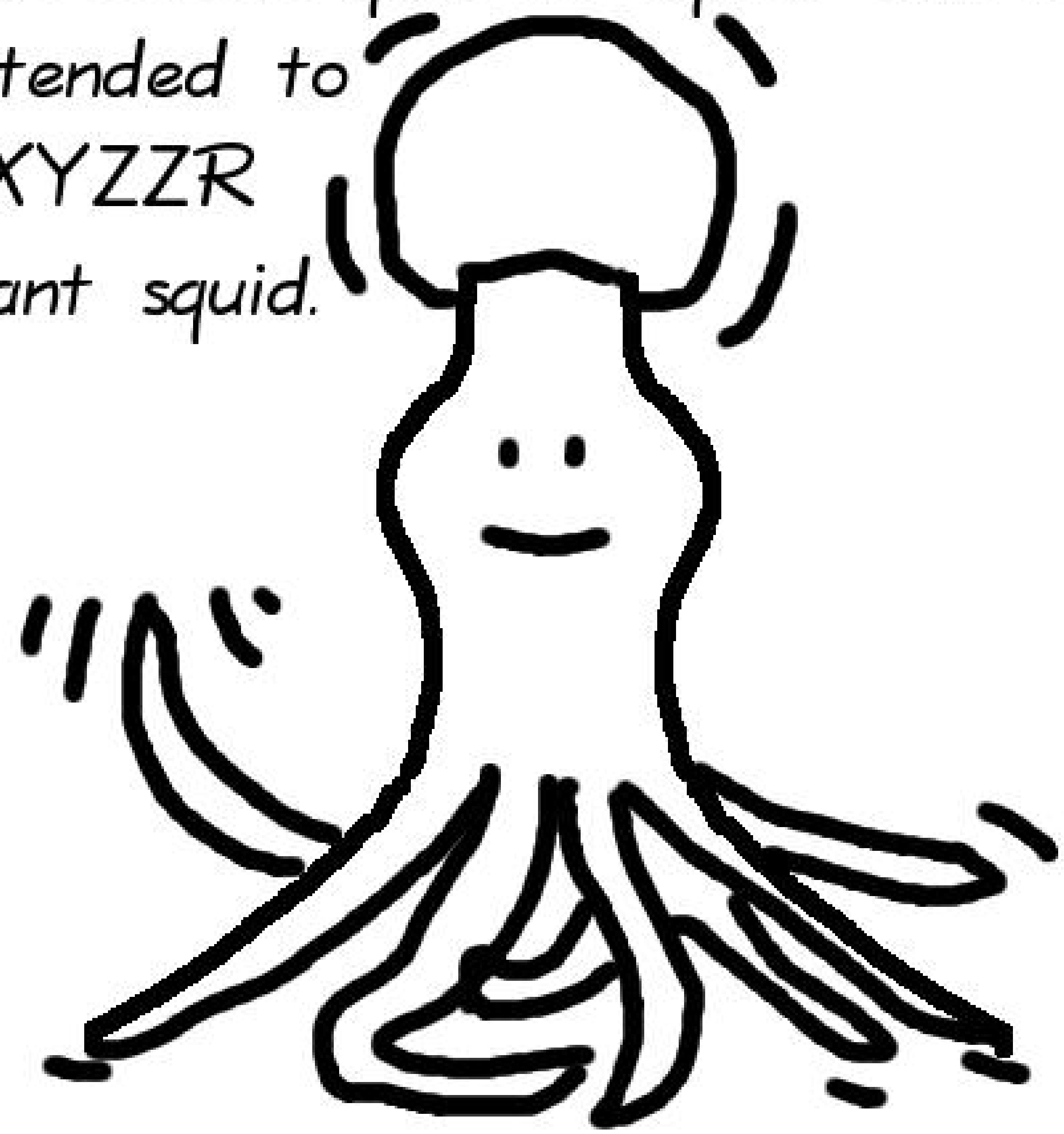
The Jacobi-quartic squid: can be extended to XXYZZR giant squid.

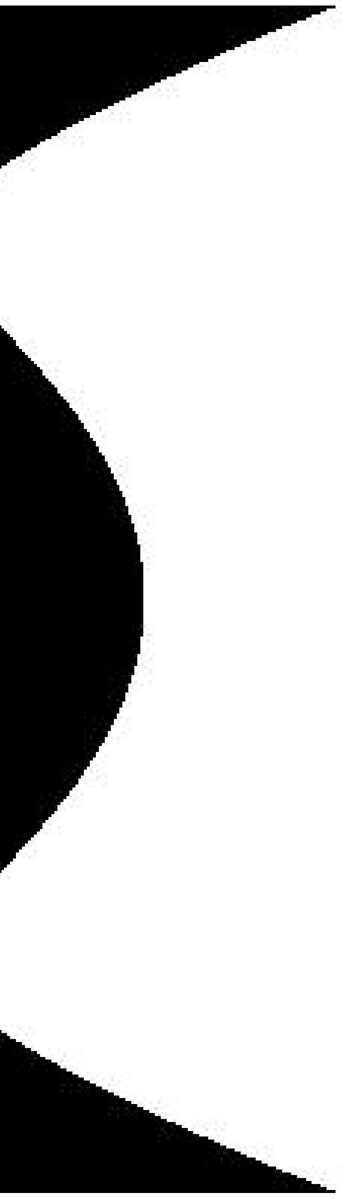




$- 1.9y^2 + 1$

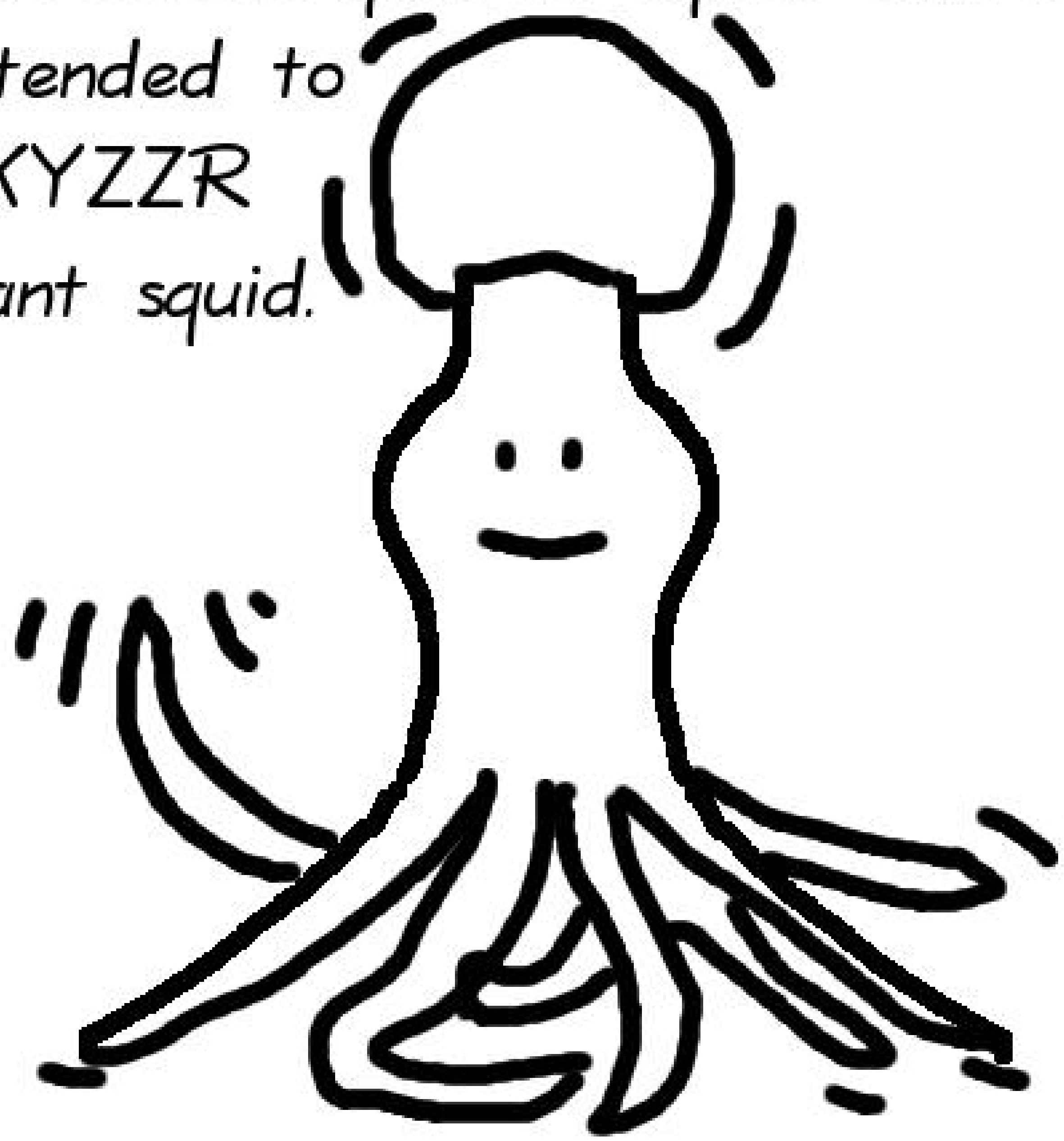
The Jacobi-quartic squid: can be extended to XXYZZR giant squid.



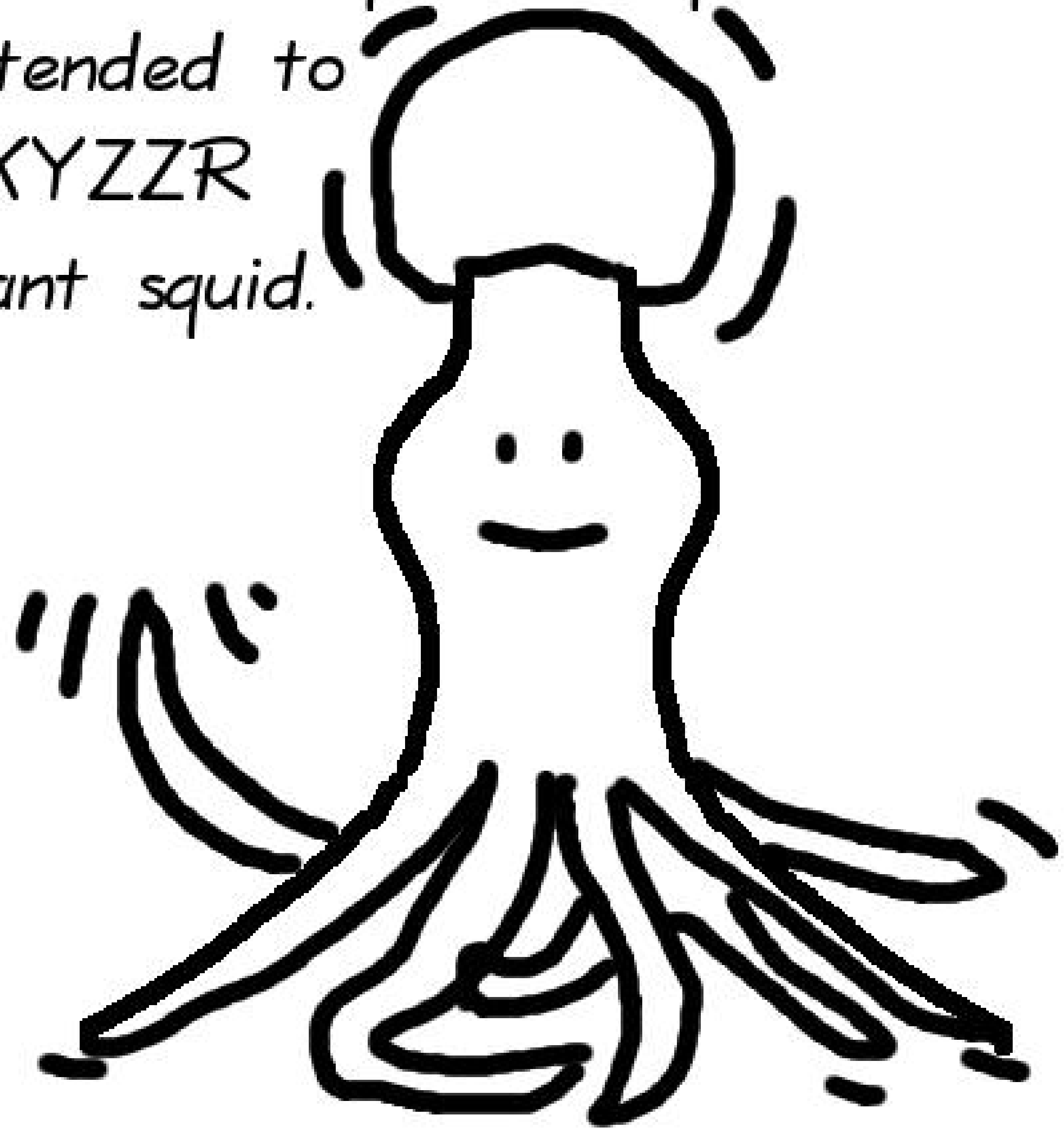


– 1

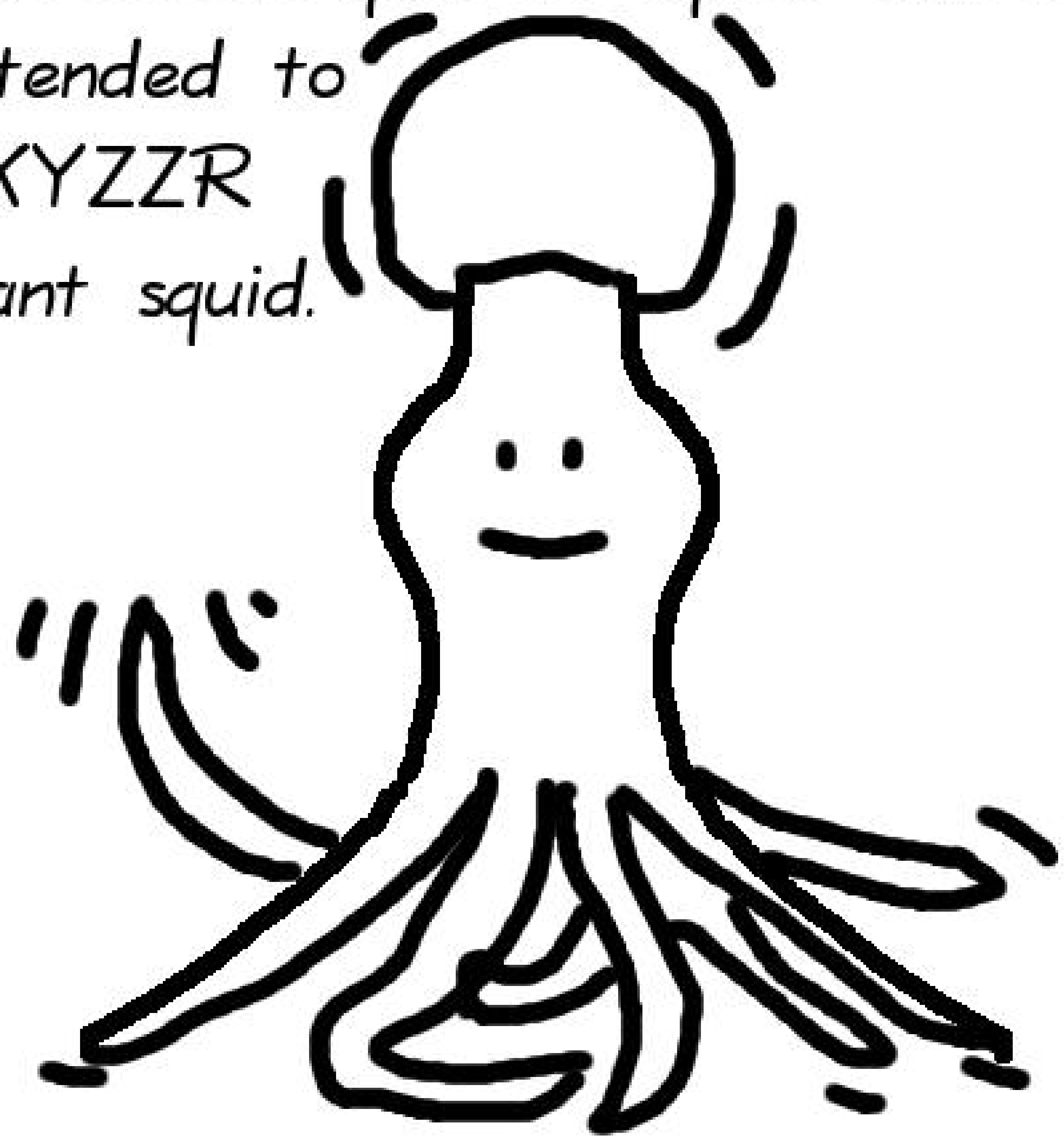
The Jacobi-quartic squid: can be extended to XXYZZR giant squid.



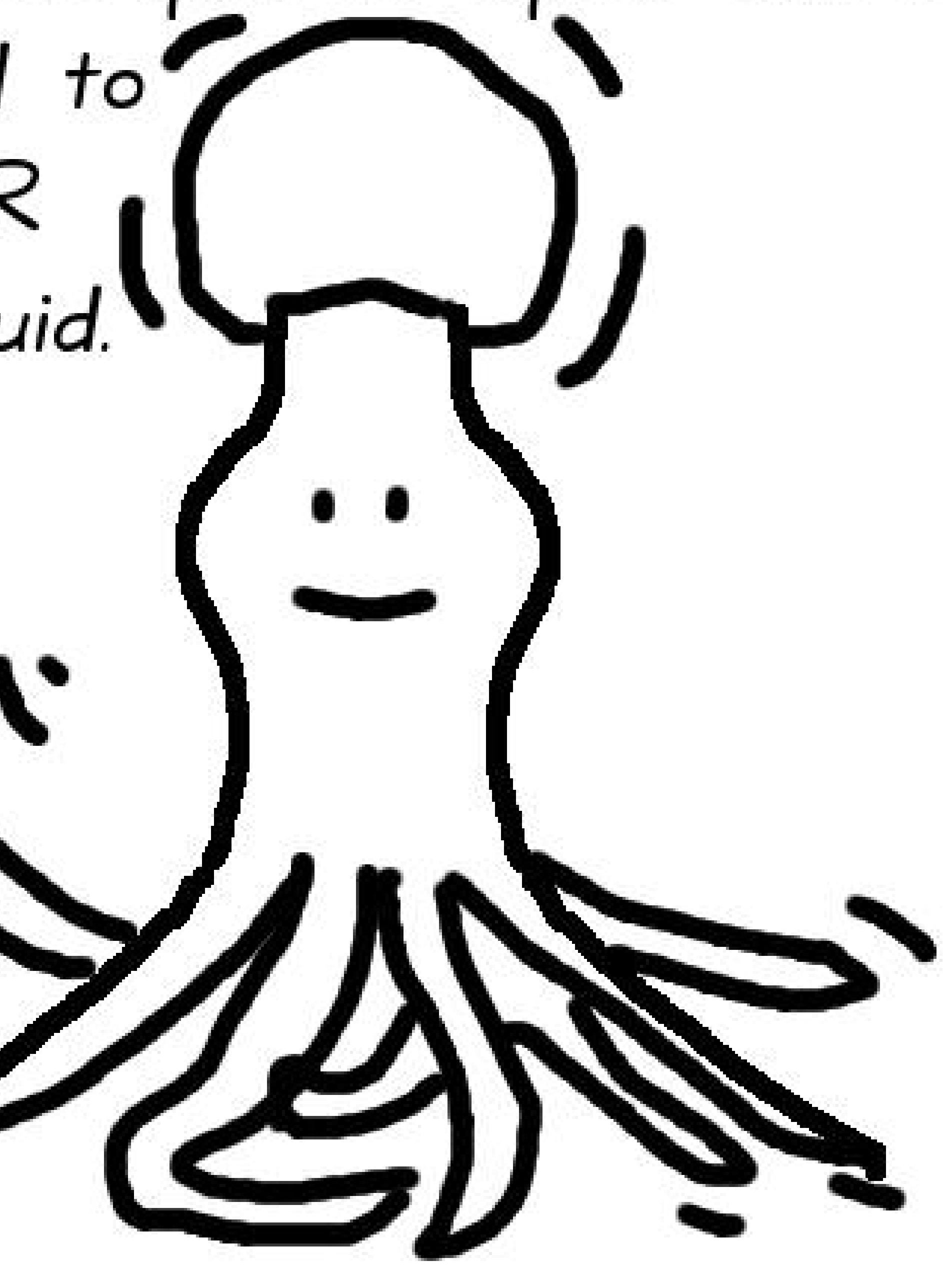
The Jacobi-quartic squid: can be extended to XXYZZR giant squid.



The Jacobi-quartic squid: can be extended to XXYZZR giant squid.



obi-quartic squid: can be
l to
R
uid.



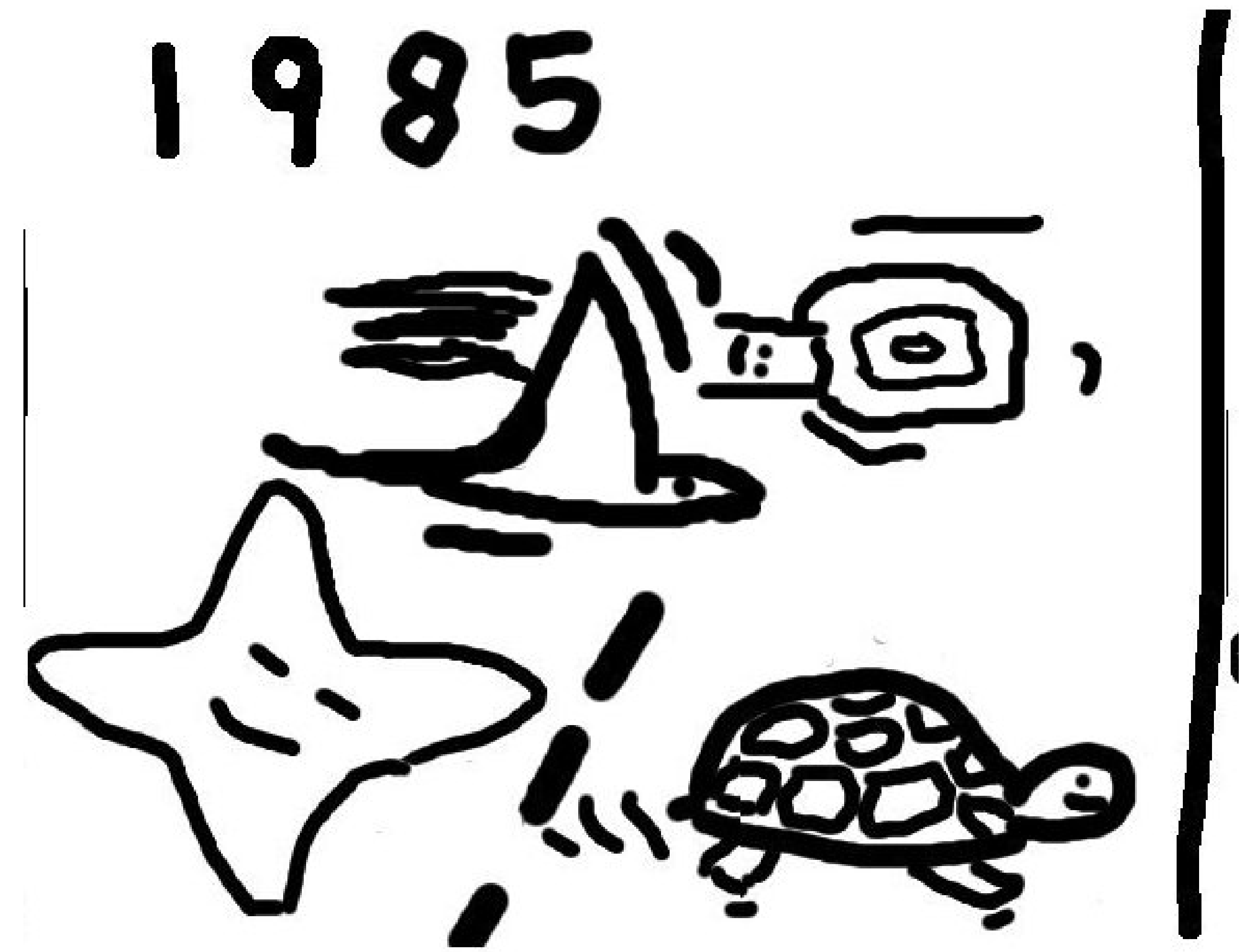
19

squid: can be

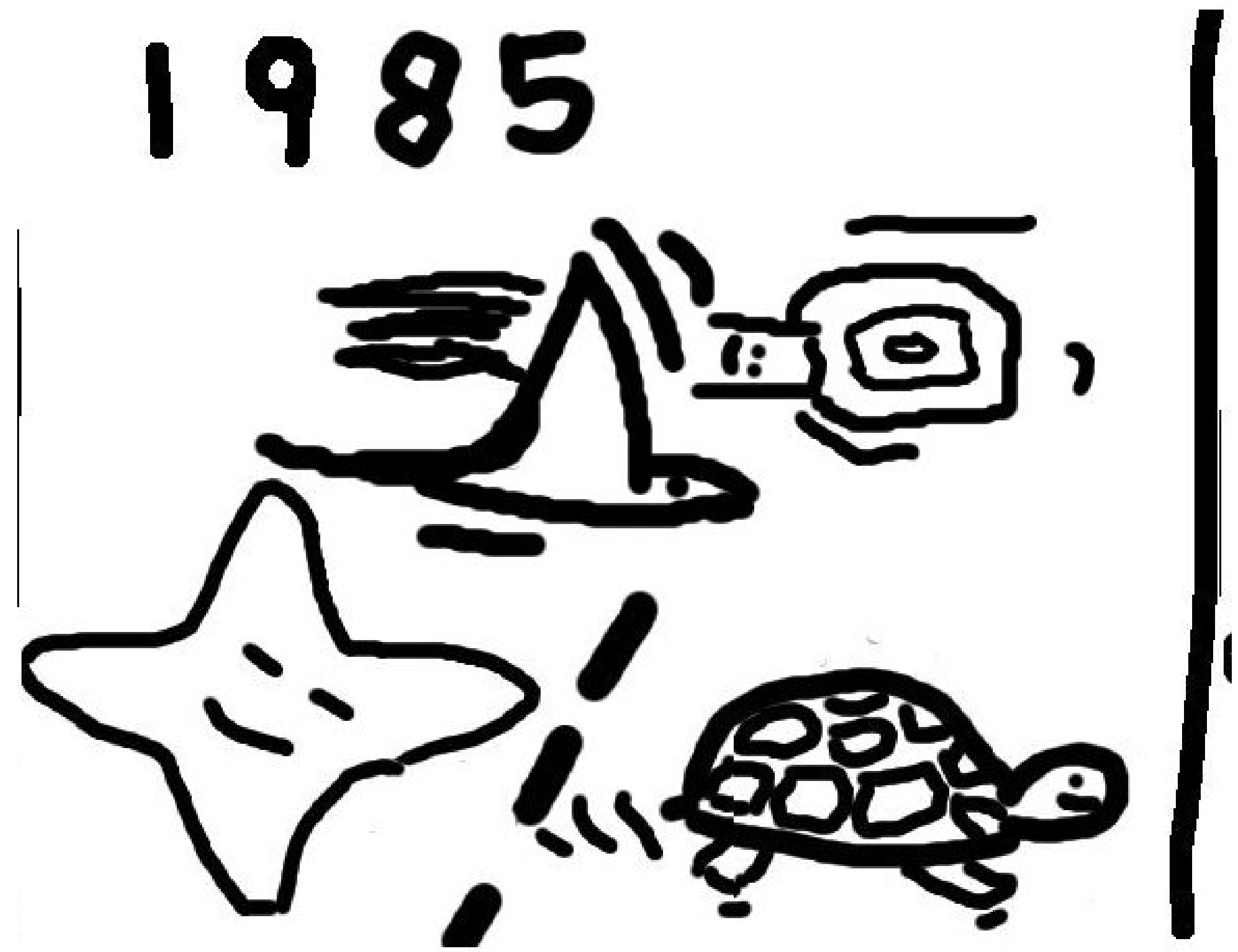
1985

n be

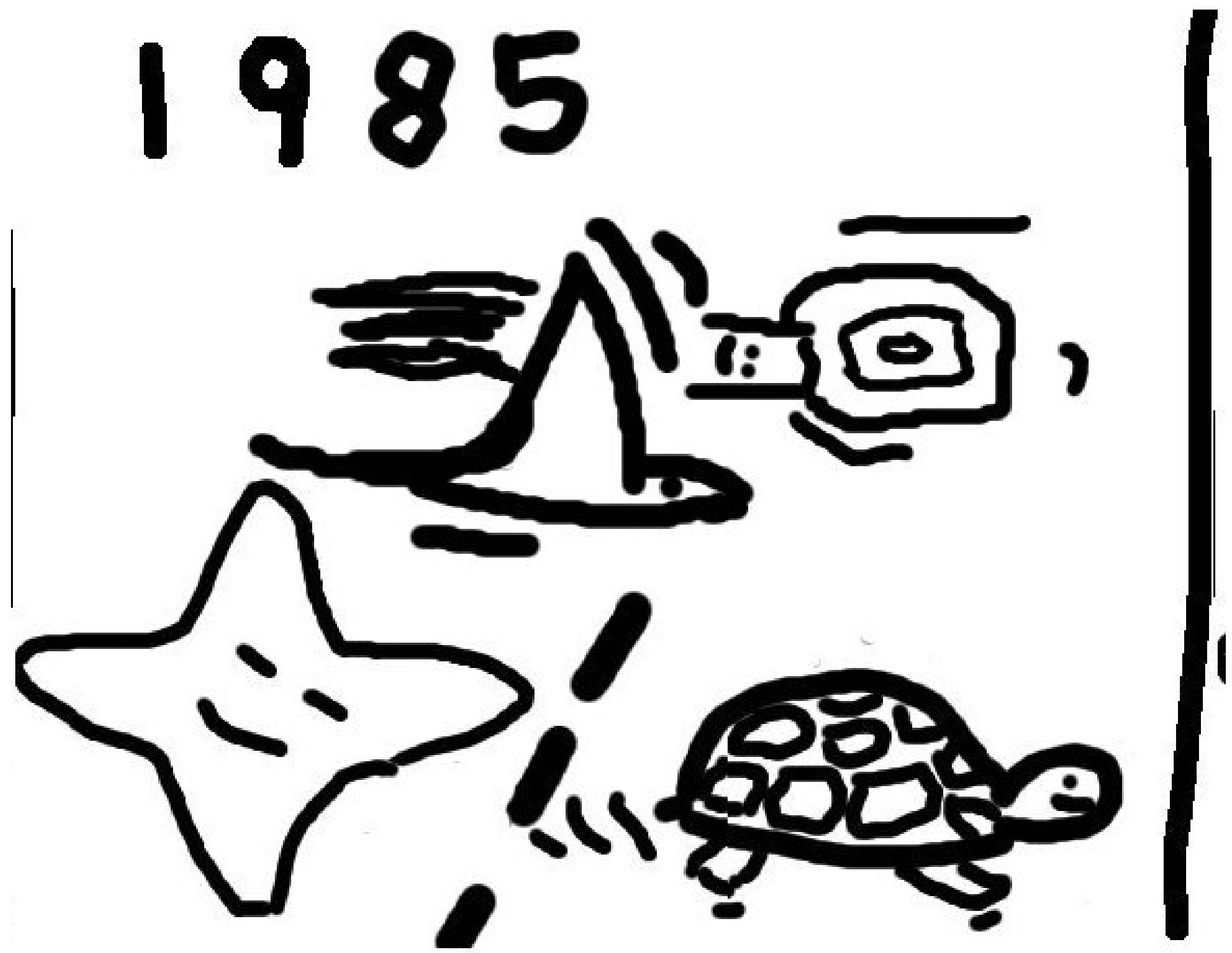

START

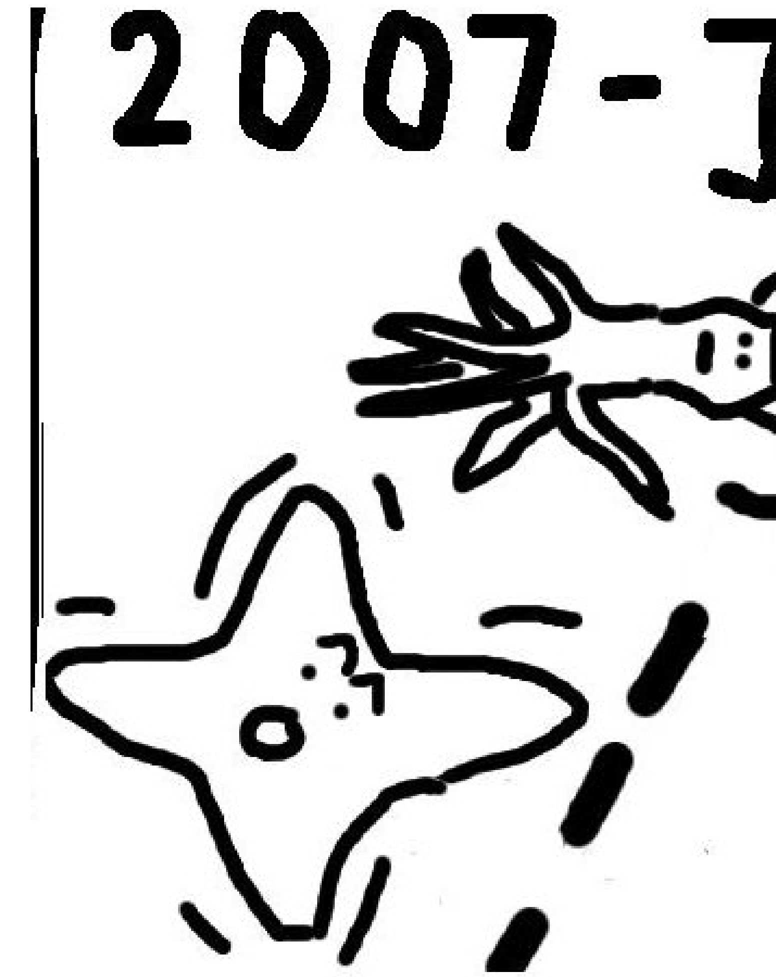

1985

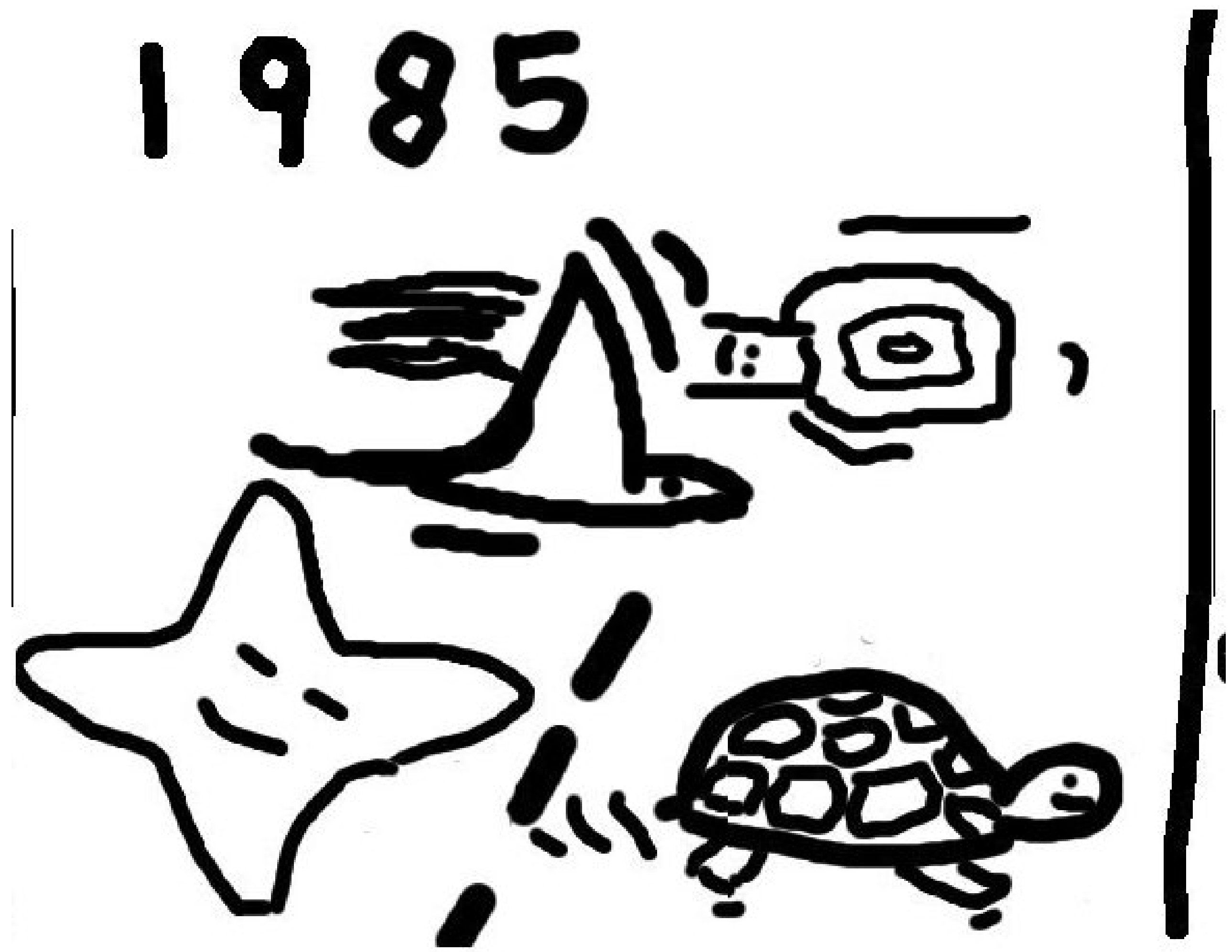

START

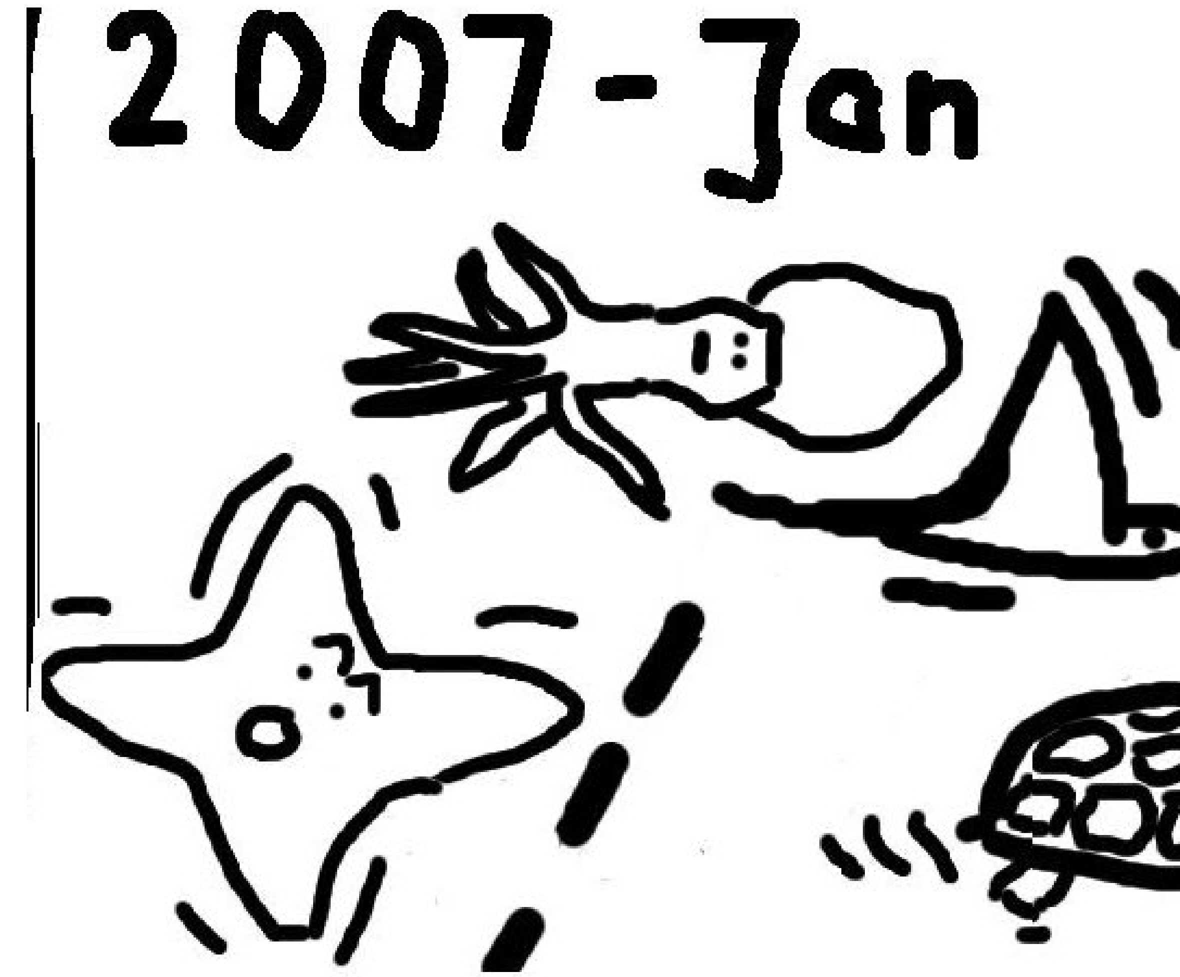

1985

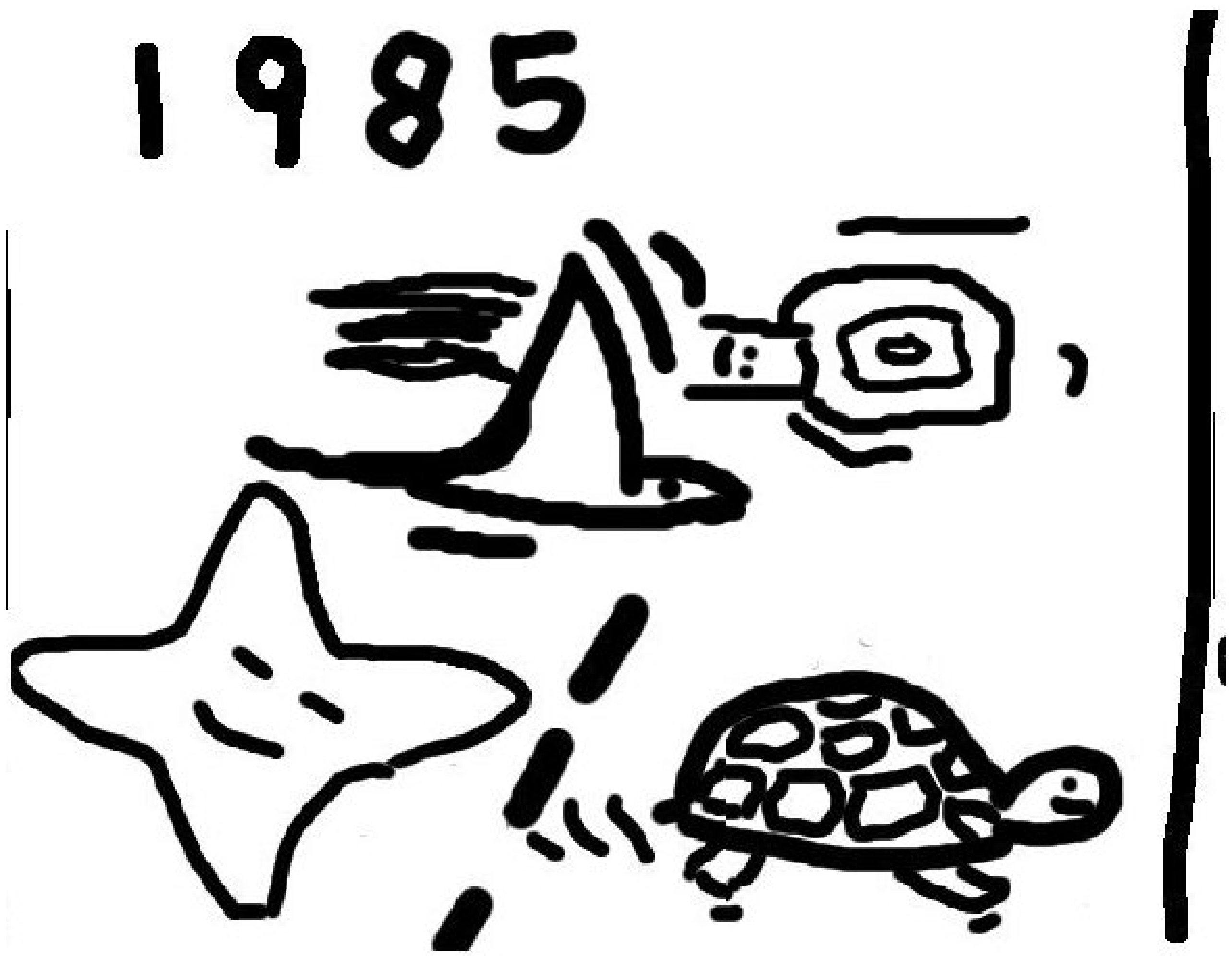

20

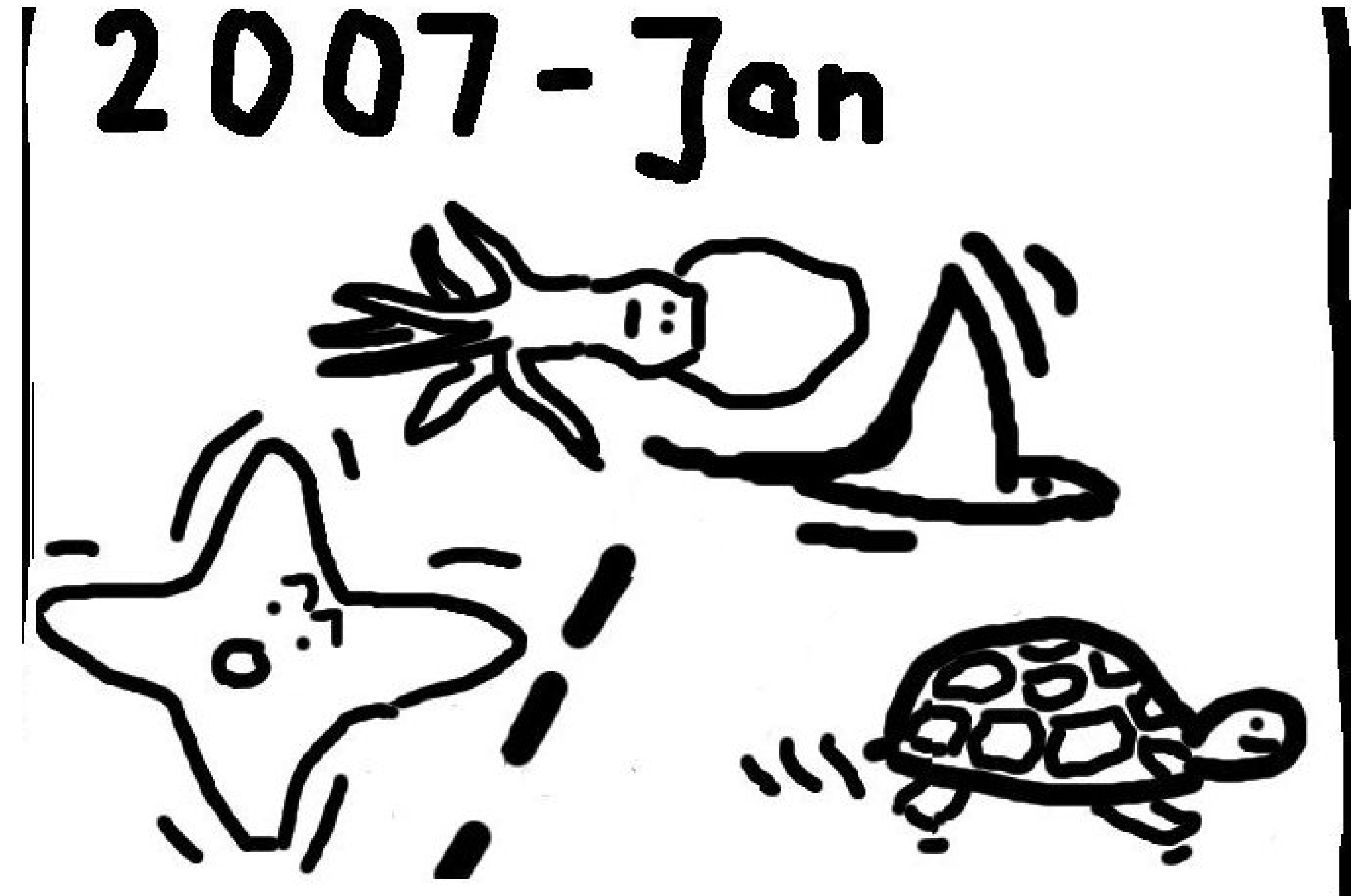

1985

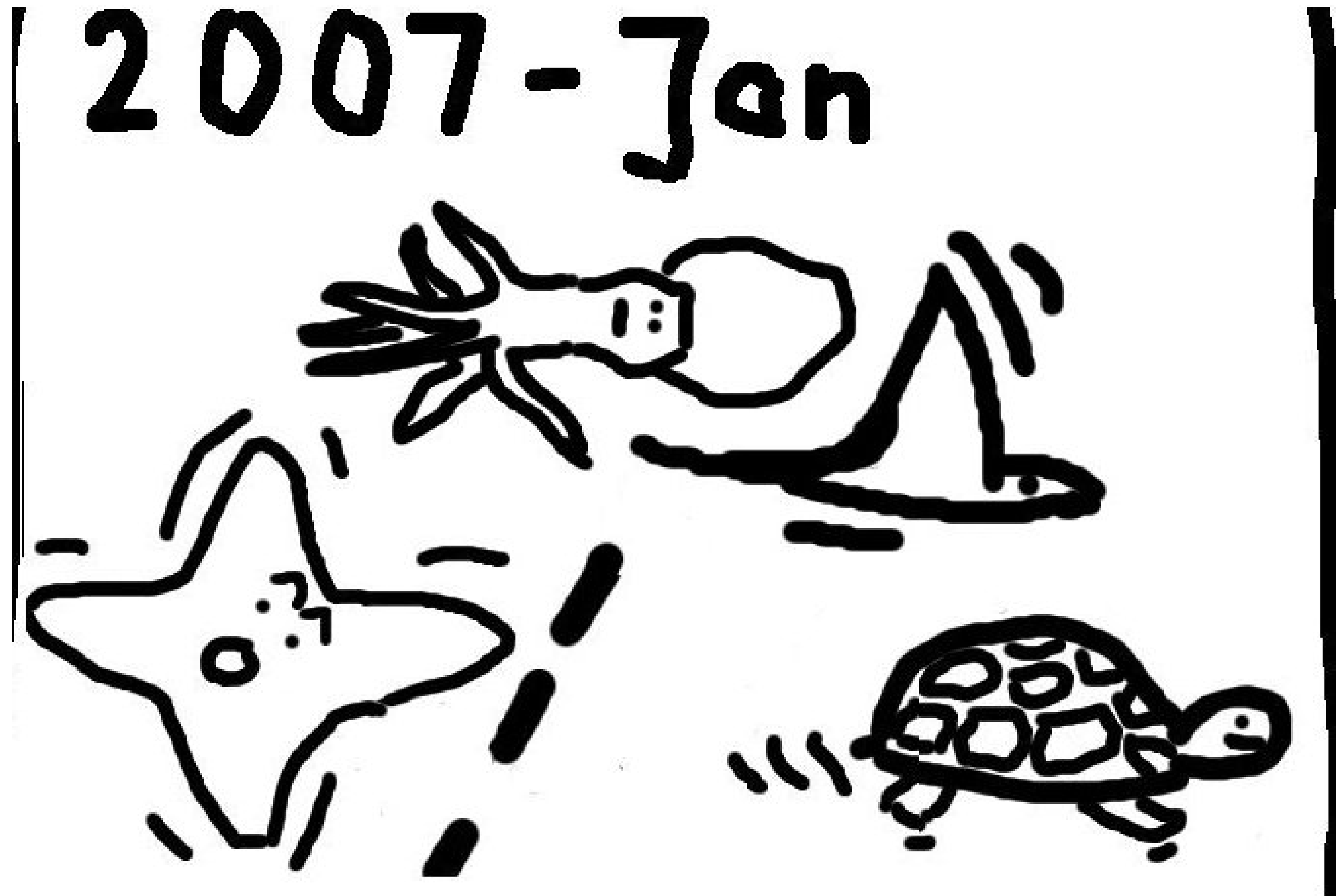
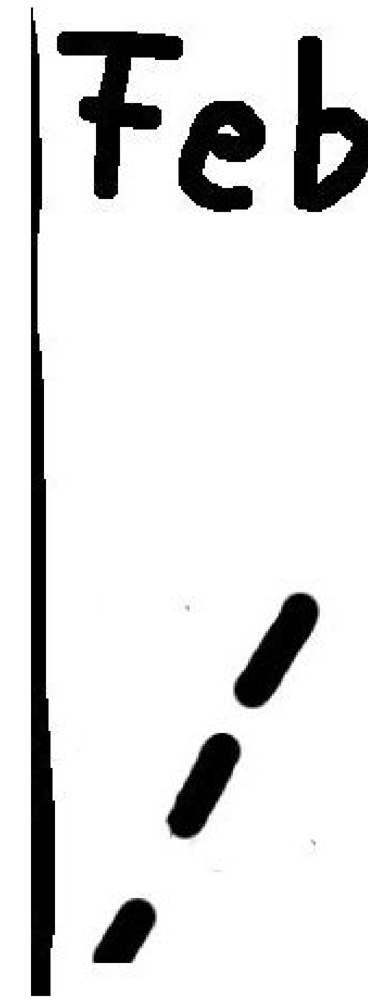

2007-


1985


2007-Jan

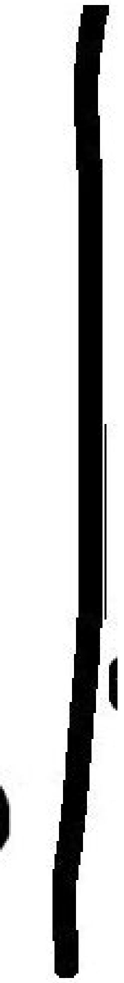

1985

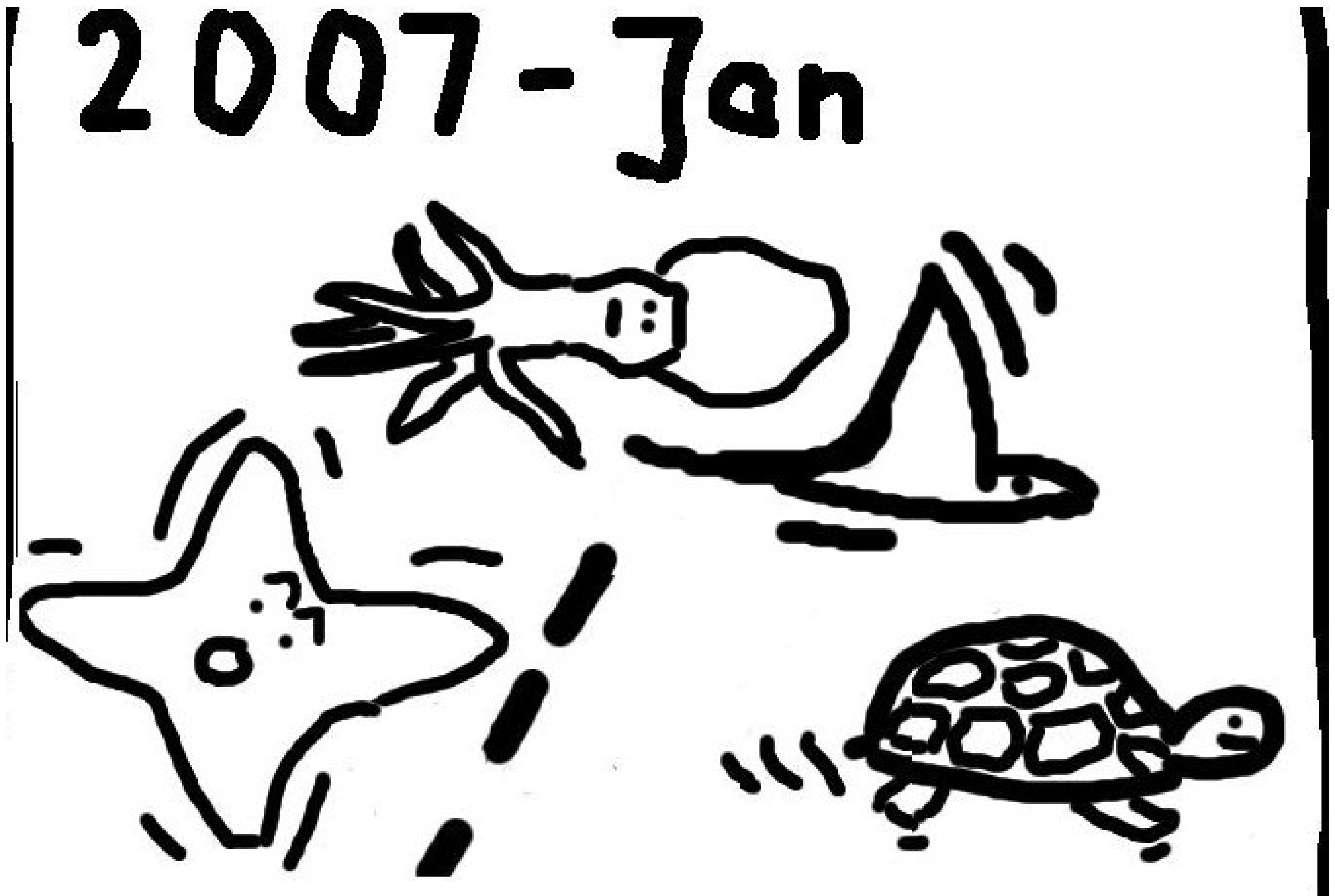

2007-Jan

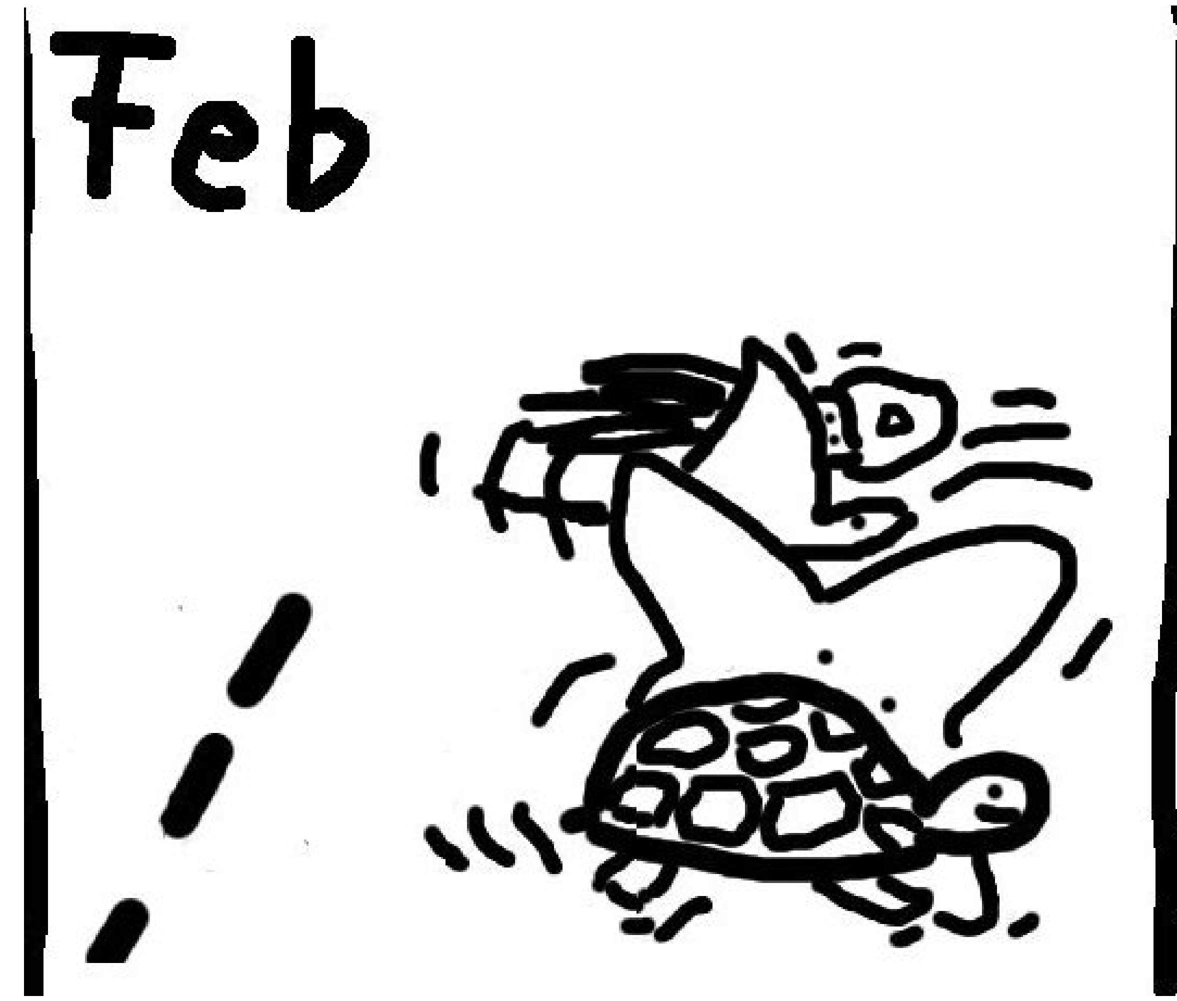

85

2007-Jan

Feb

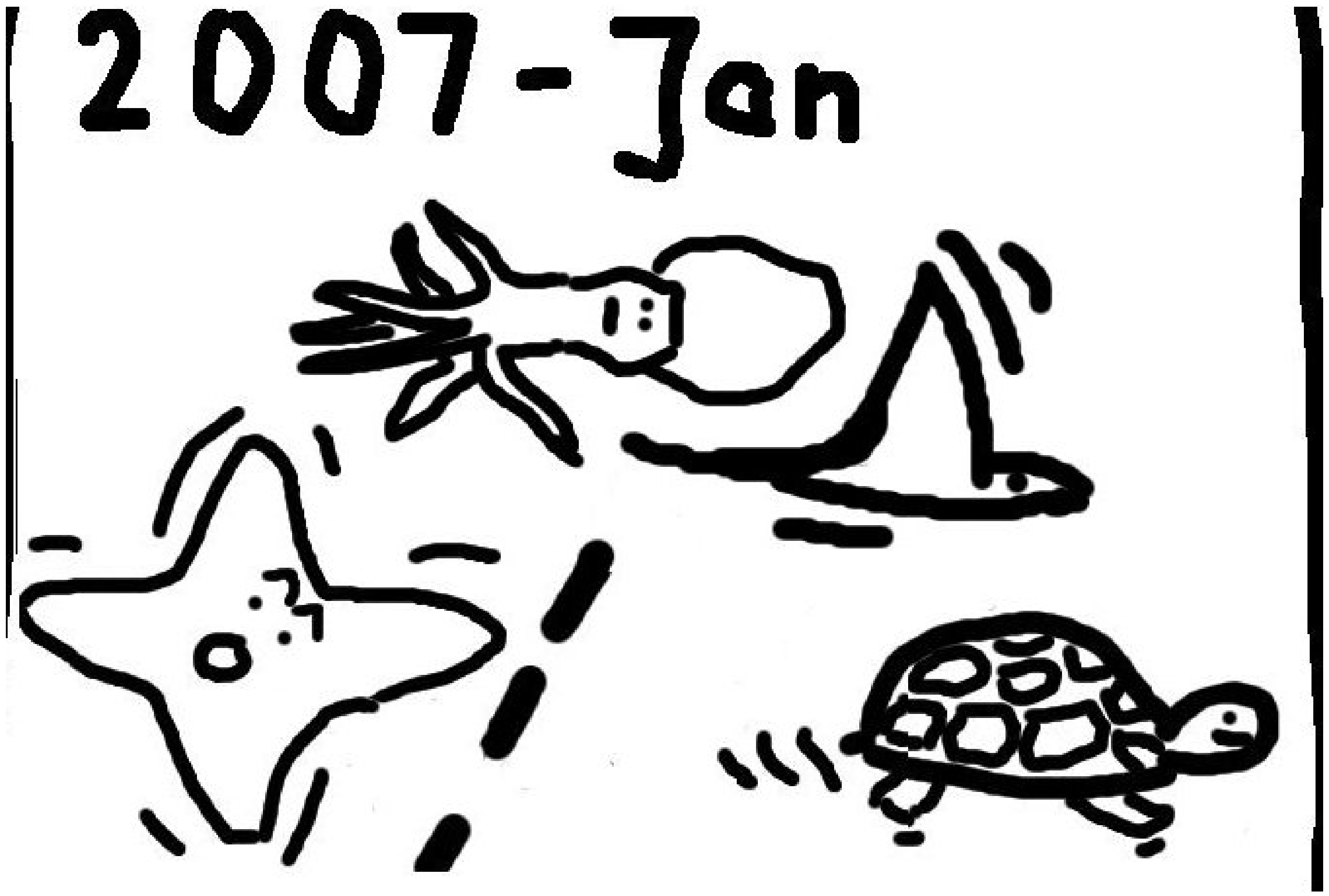

2007-Jan

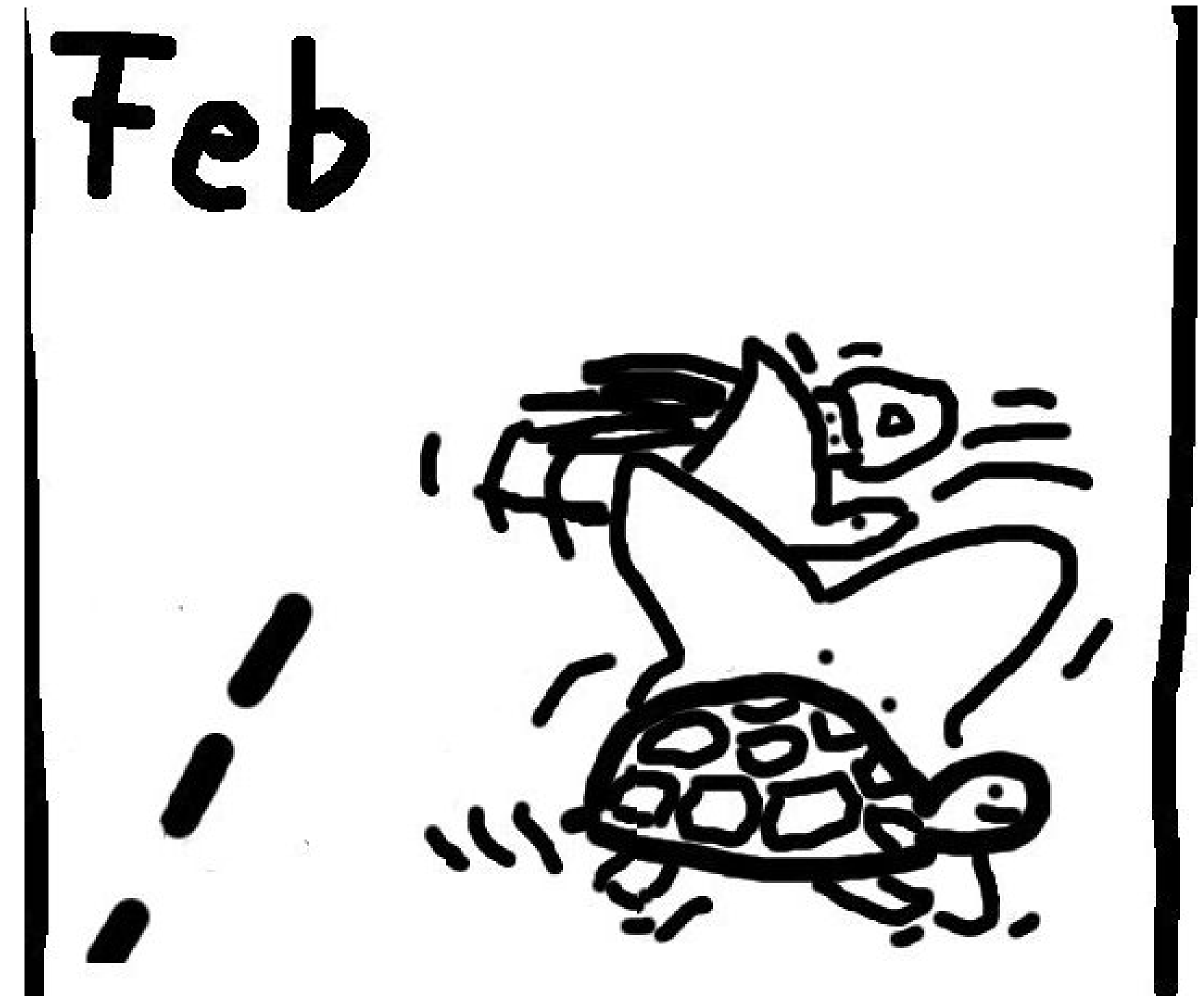

Feb

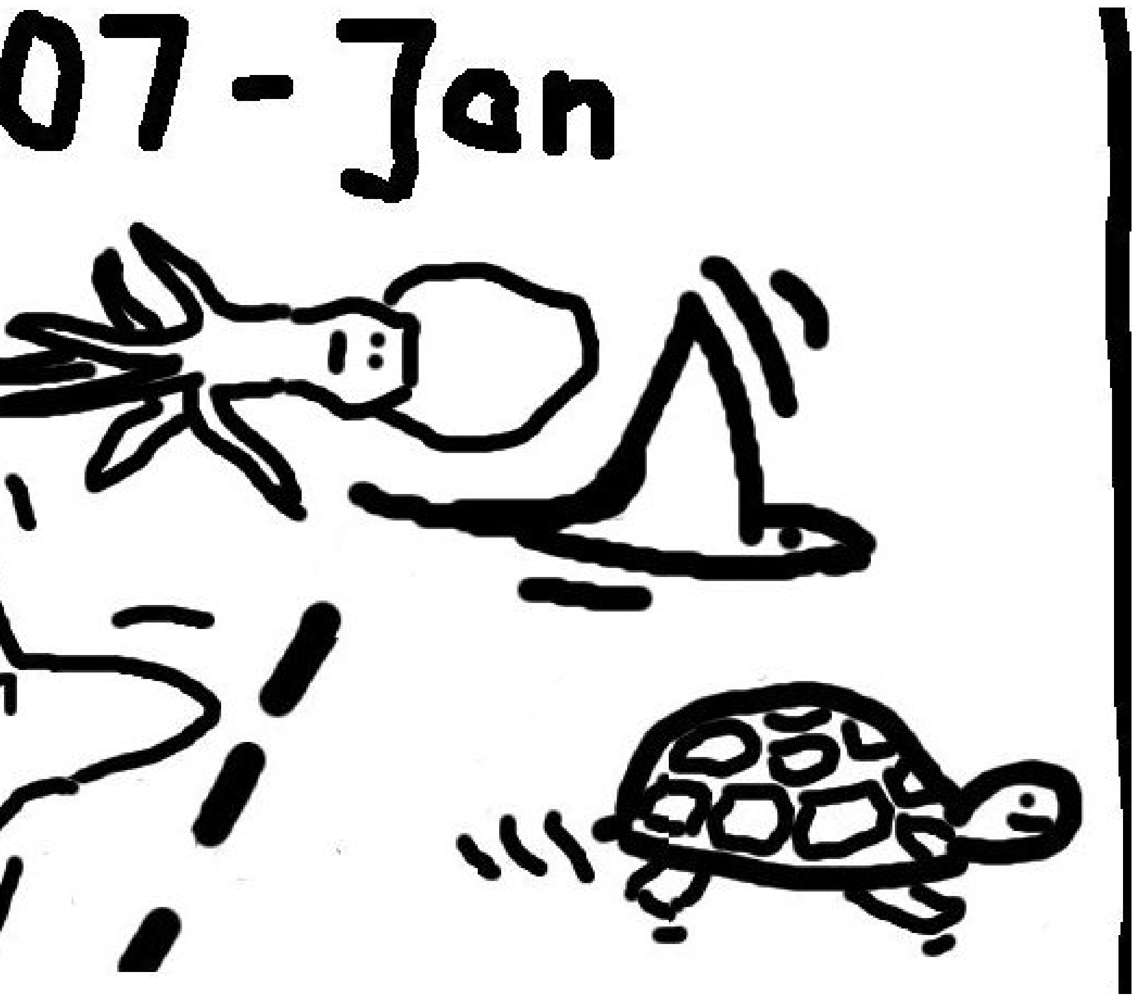


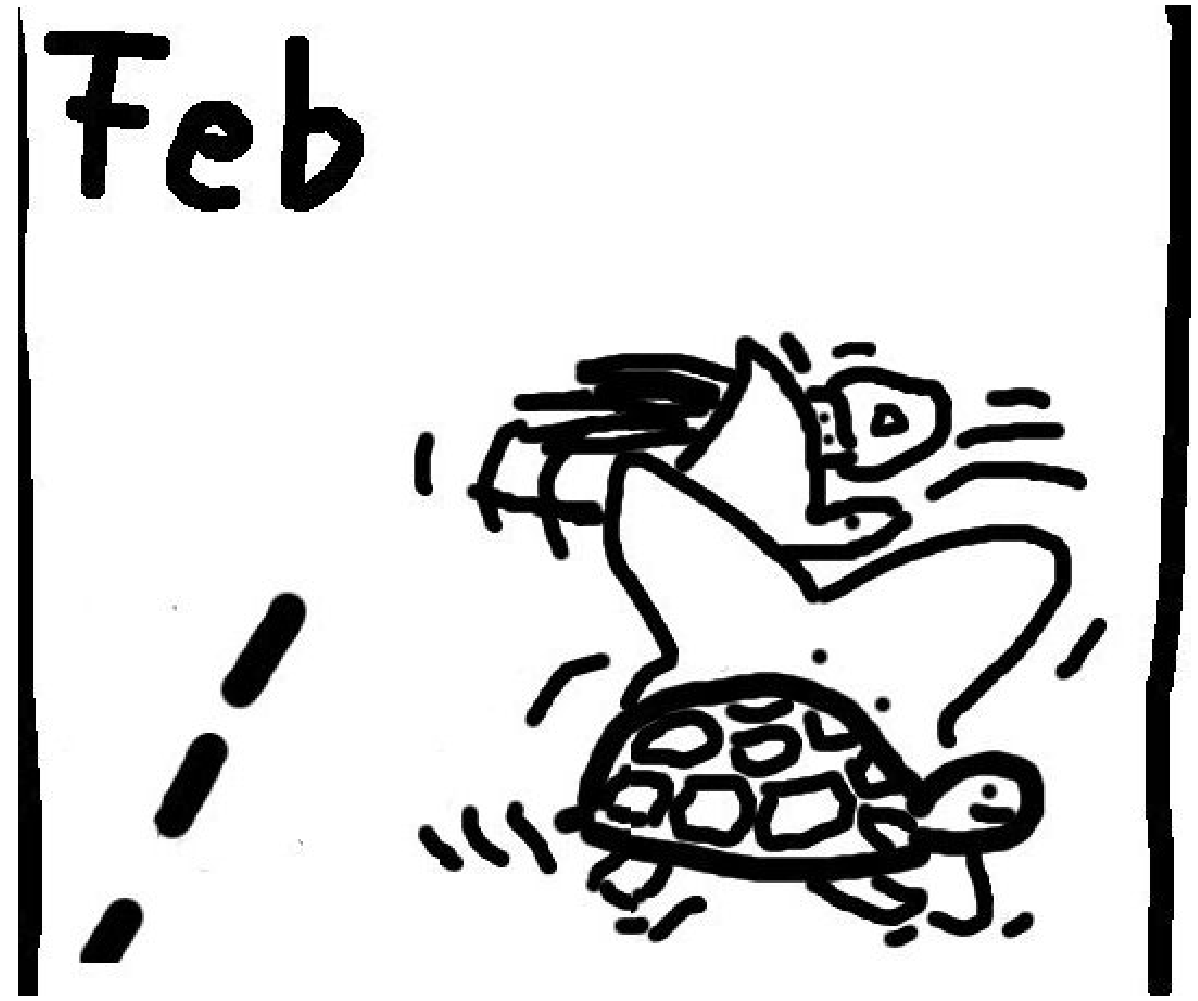

2007-Jan


Feb

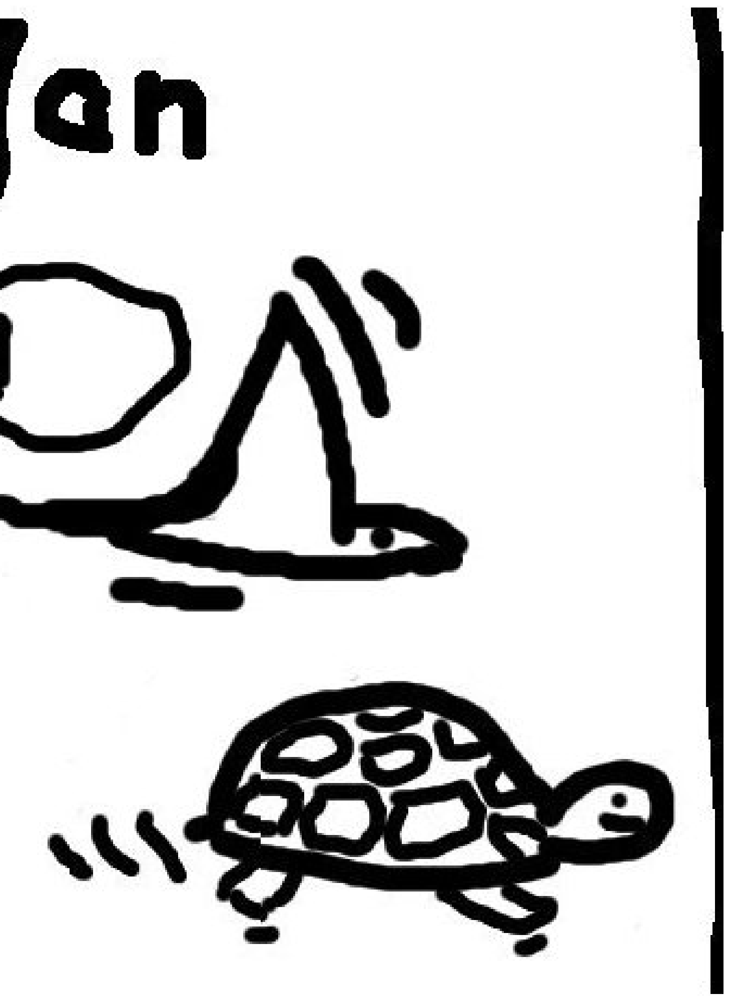

2007-Jan

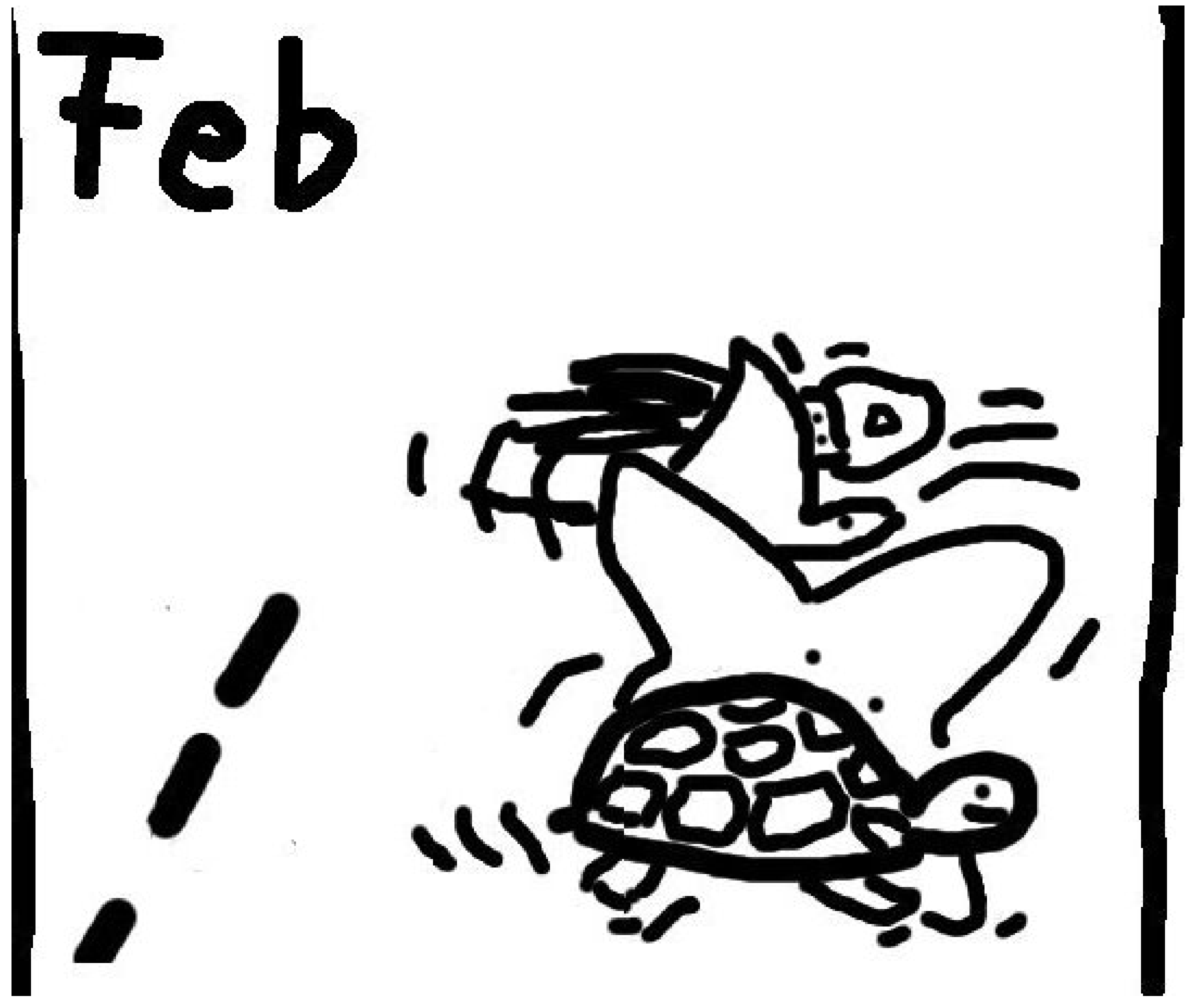

Feb


07 - Jan

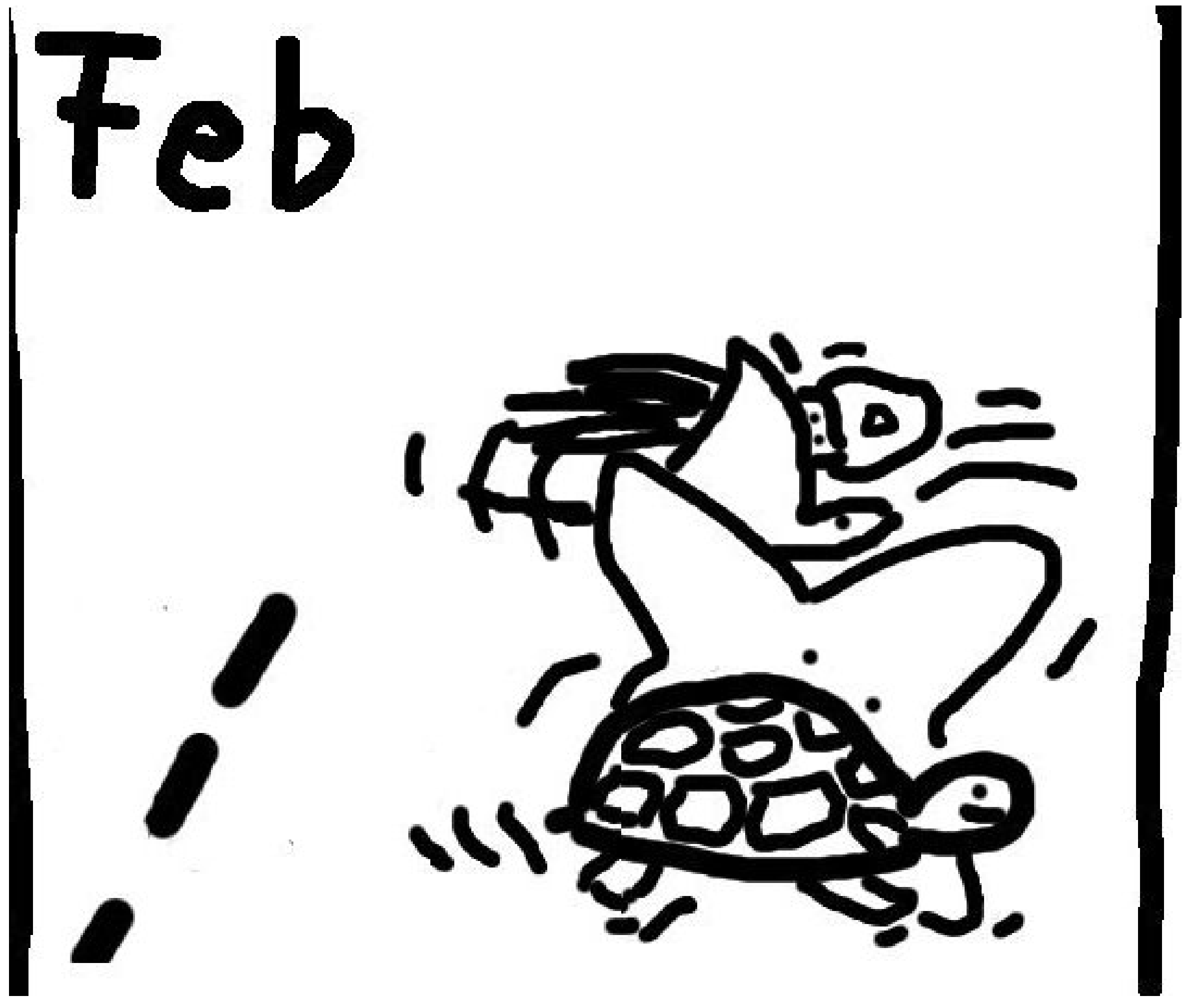

Feb

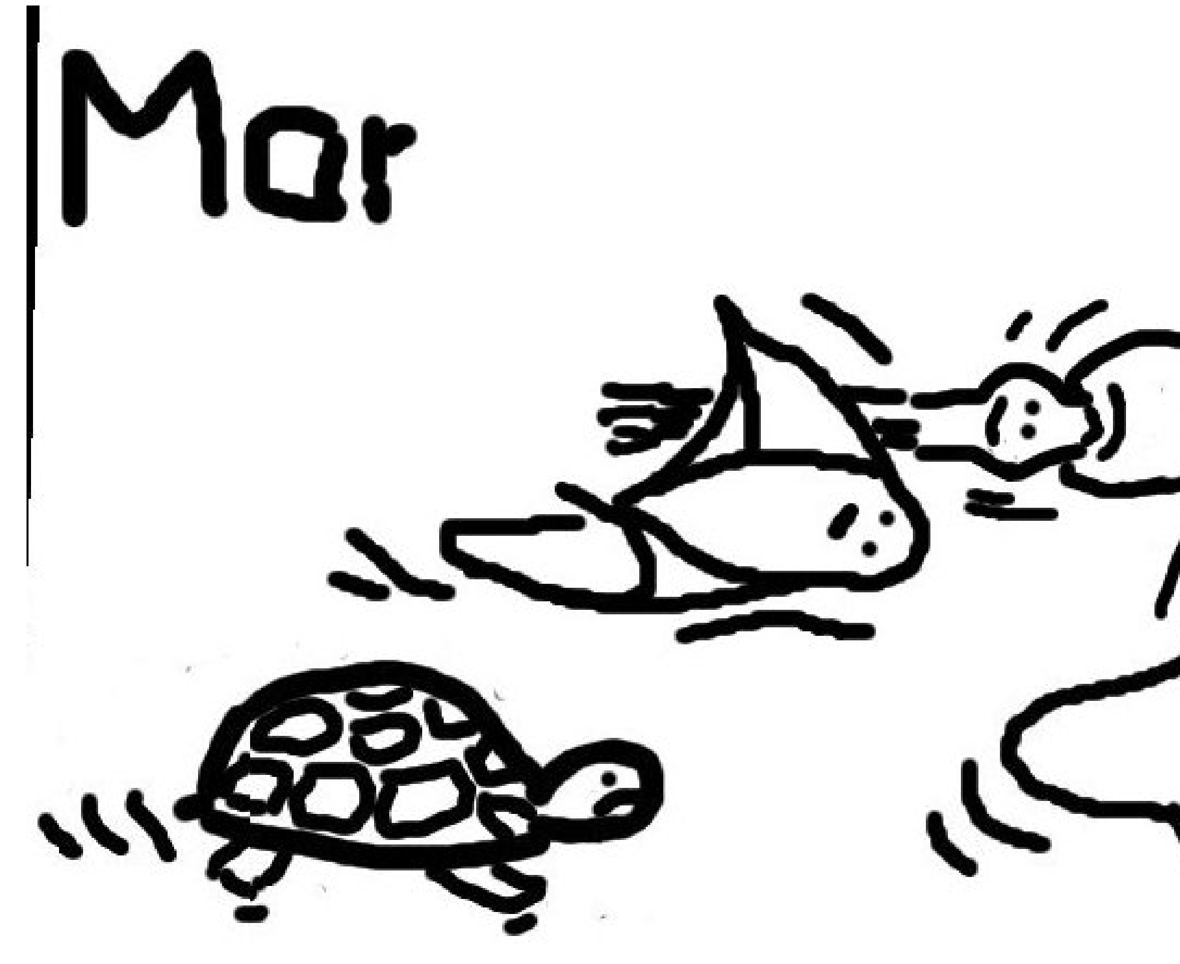

Ma

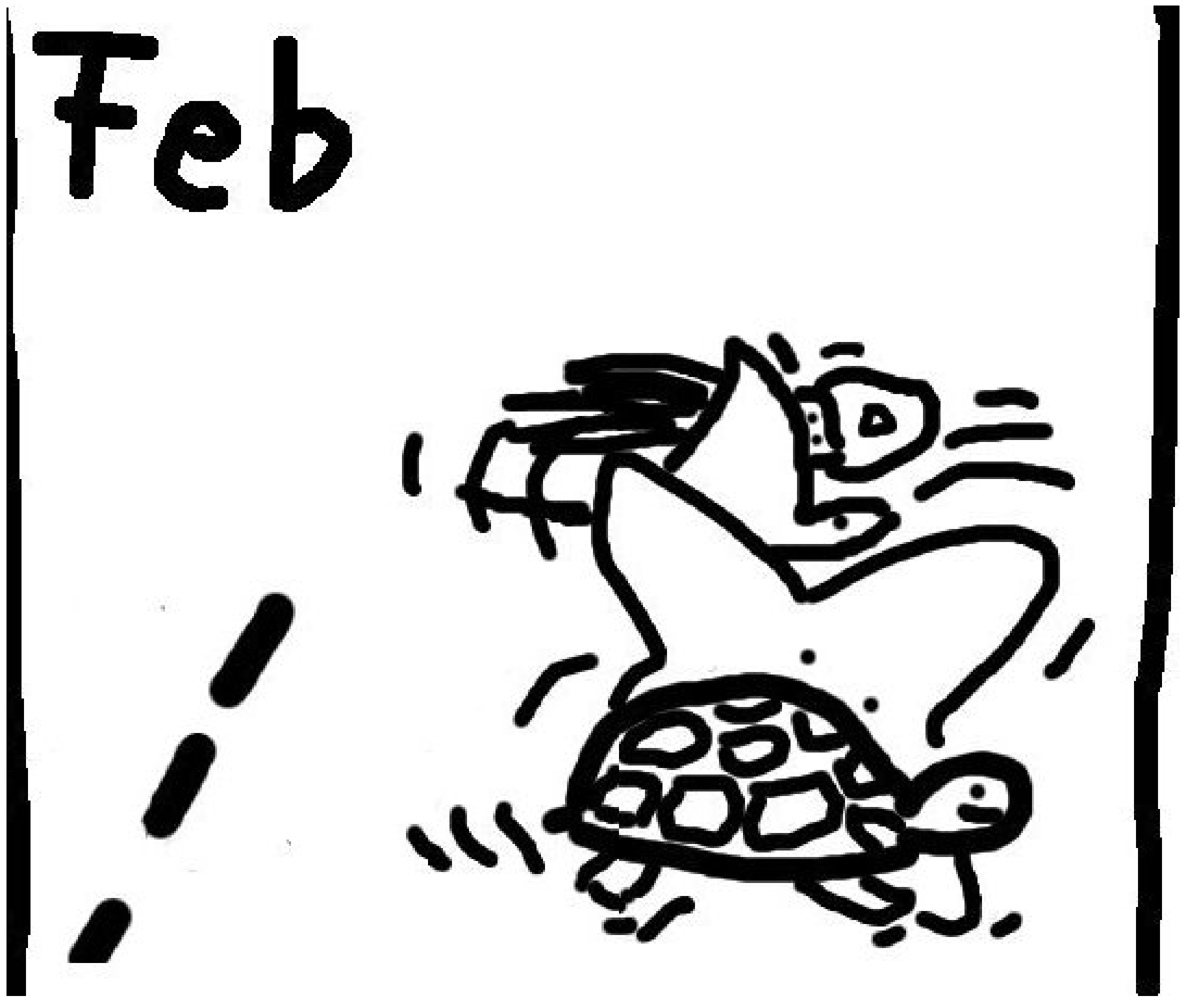

Jan

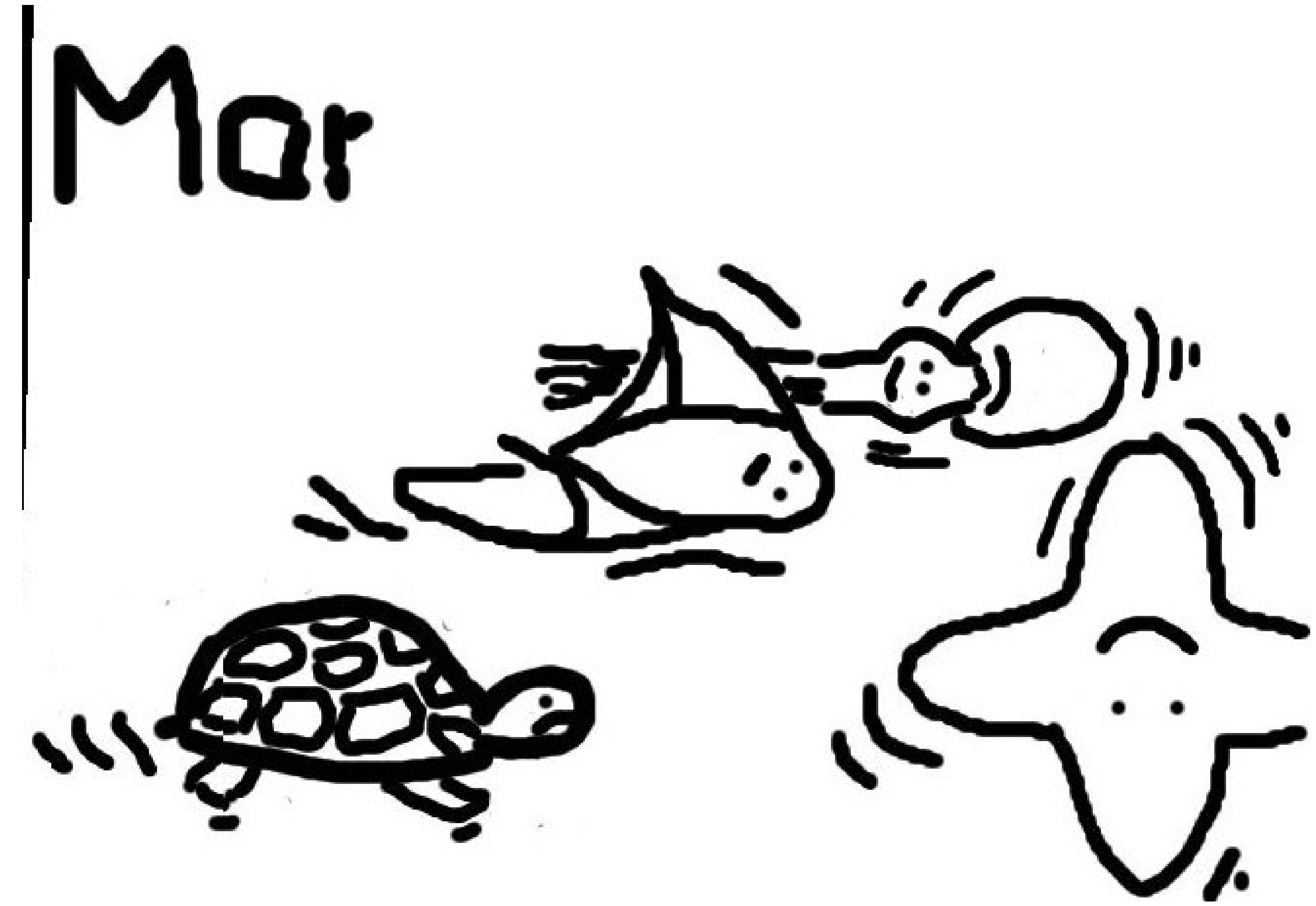

Feb


Mar

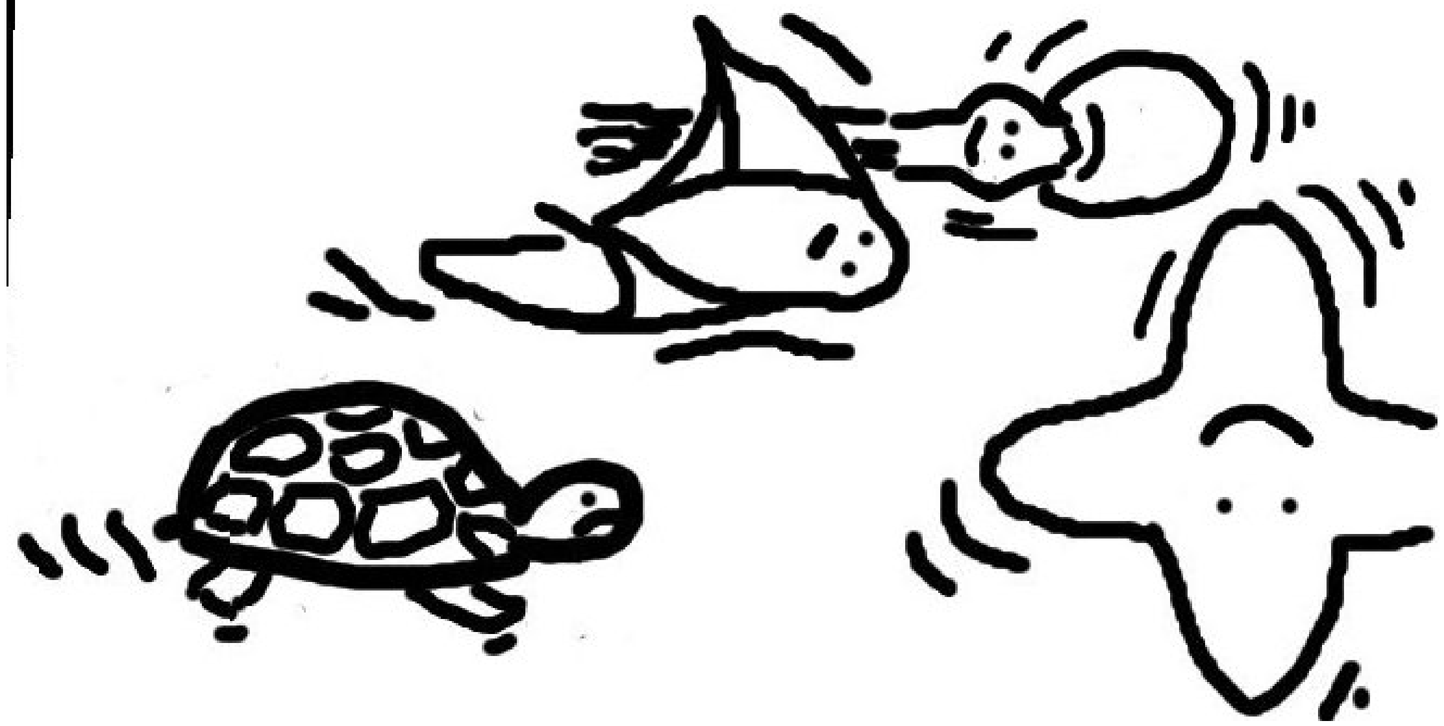

Feb


Mar


Feb


Mar

Mor



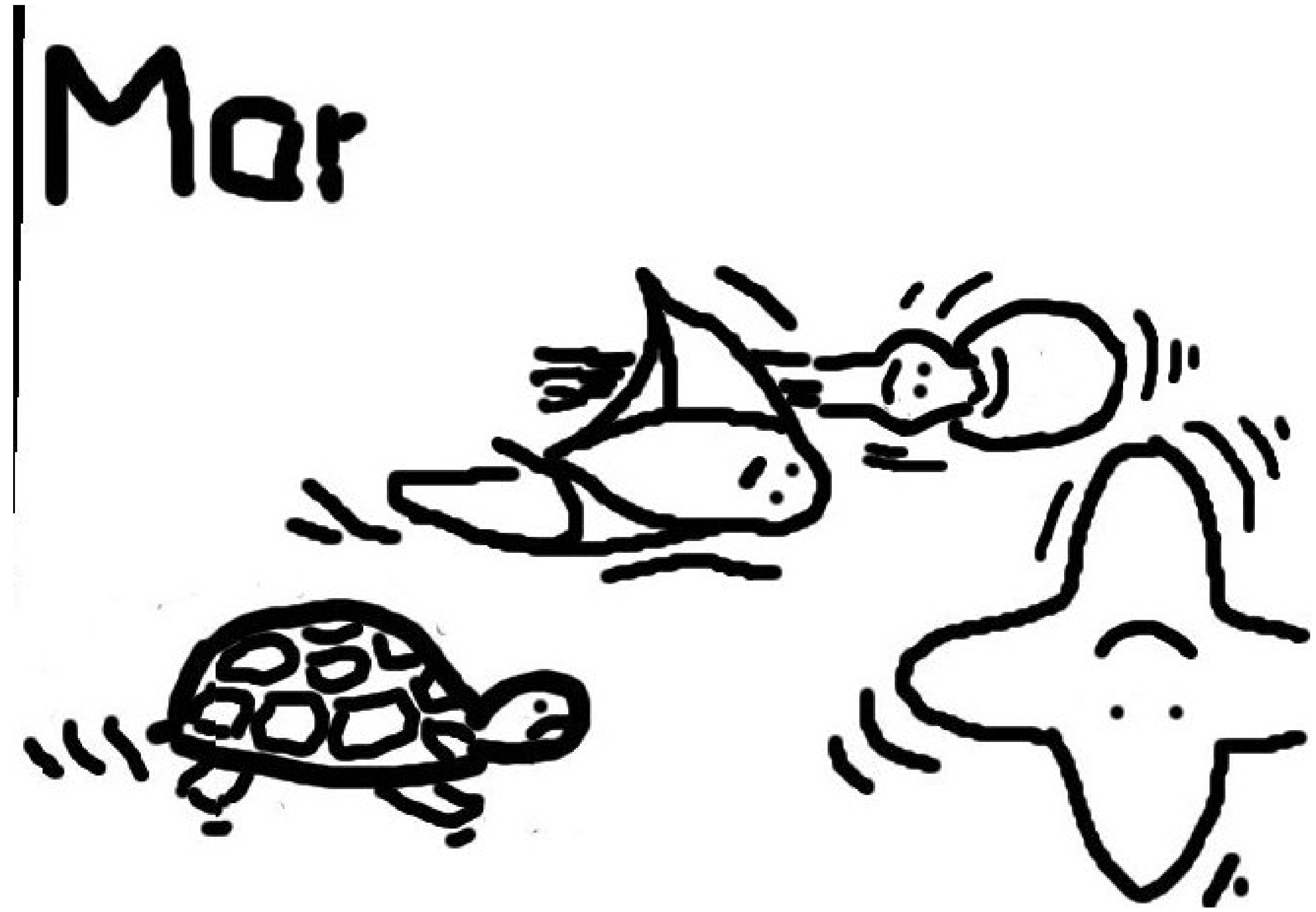
More ad

Explicit-

hyperel

EFD has
formulas
for ADD
in 51 rep
on 13 sh

Not yet
generalit
(e.g., He
complete
(e.g., ch



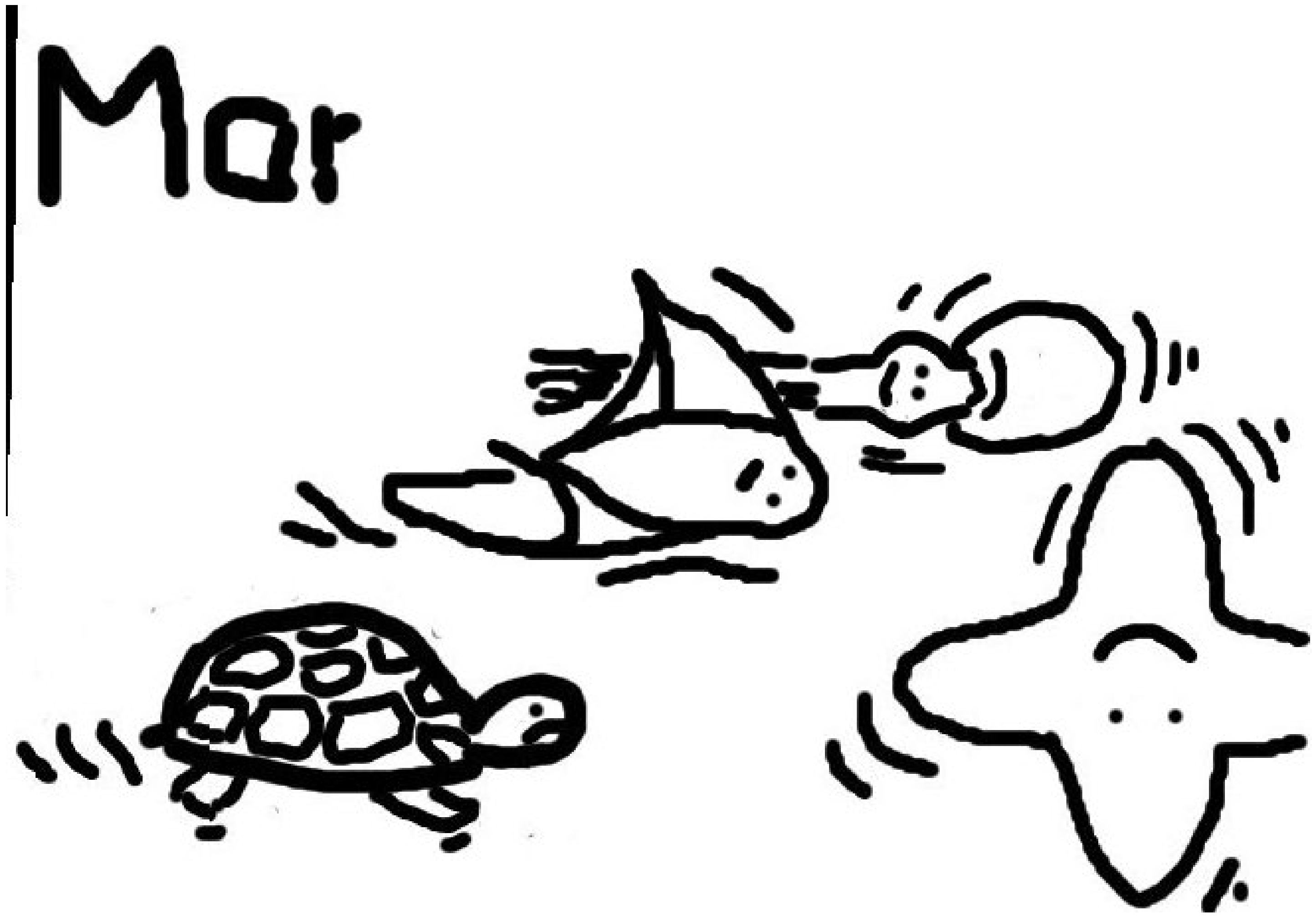
More addition forr

Explicit-Formulas

hyperelliptic.c

EFD has 583 com
formulas and oper
for ADD, DBL, etc
in 51 representatio
on 13 shapes of el

Not yet handled by
generality of curve
(e.g., Hessian orde
complete addition
(e.g., checking for

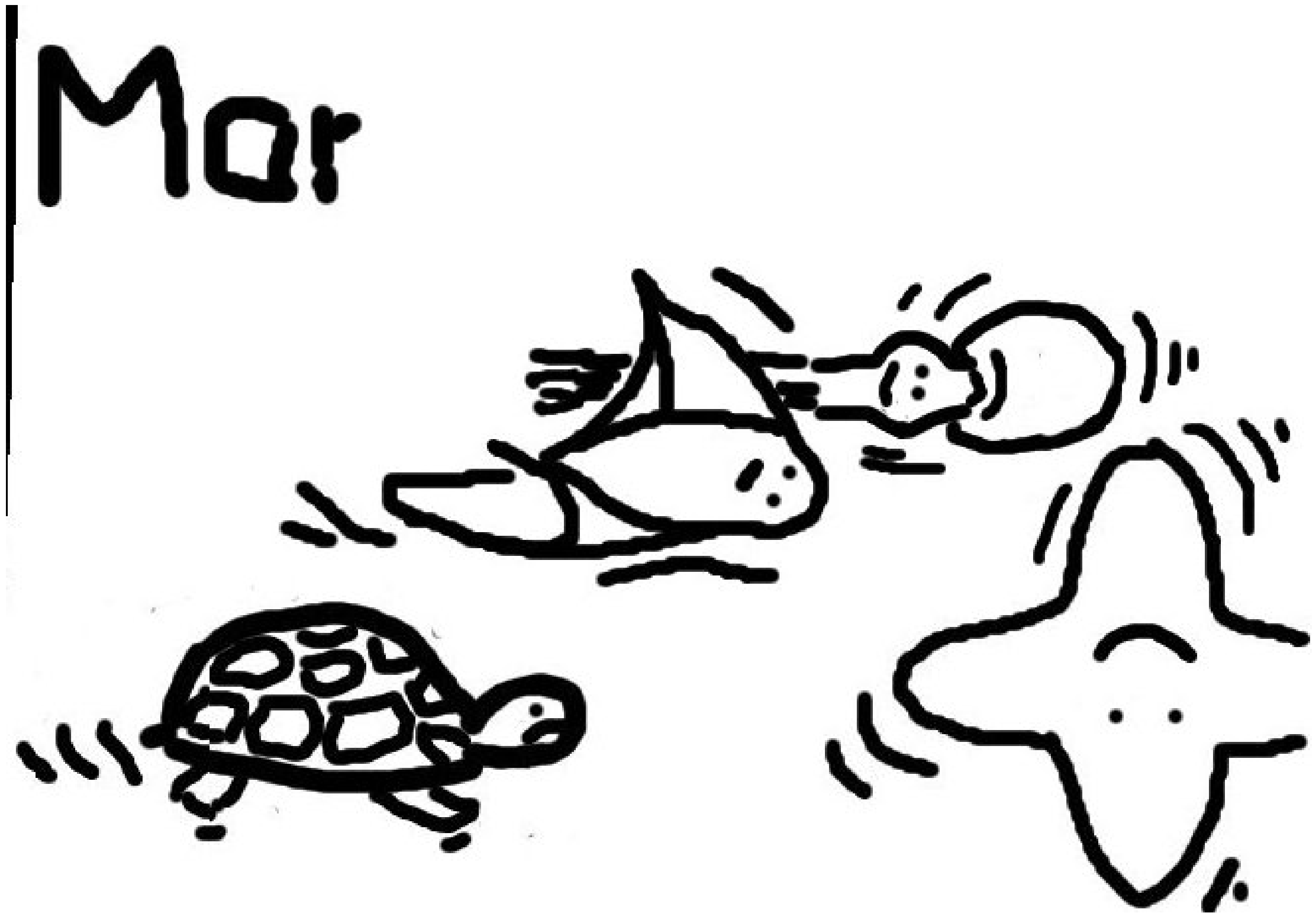


More addition formulas

Explicit-Formulas Database:
hyperelliptic.org/EFD

EFD has 583 computer-verif
formulas and operation coun
for ADD, DBL, etc.
in 51 representations
on 13 shapes of elliptic curv

Not yet handled by compute
generality of curve shapes
(e.g., Hessian order $\in 3\mathbf{Z}$);
complete addition algorithm
(e.g., checking for ∞).

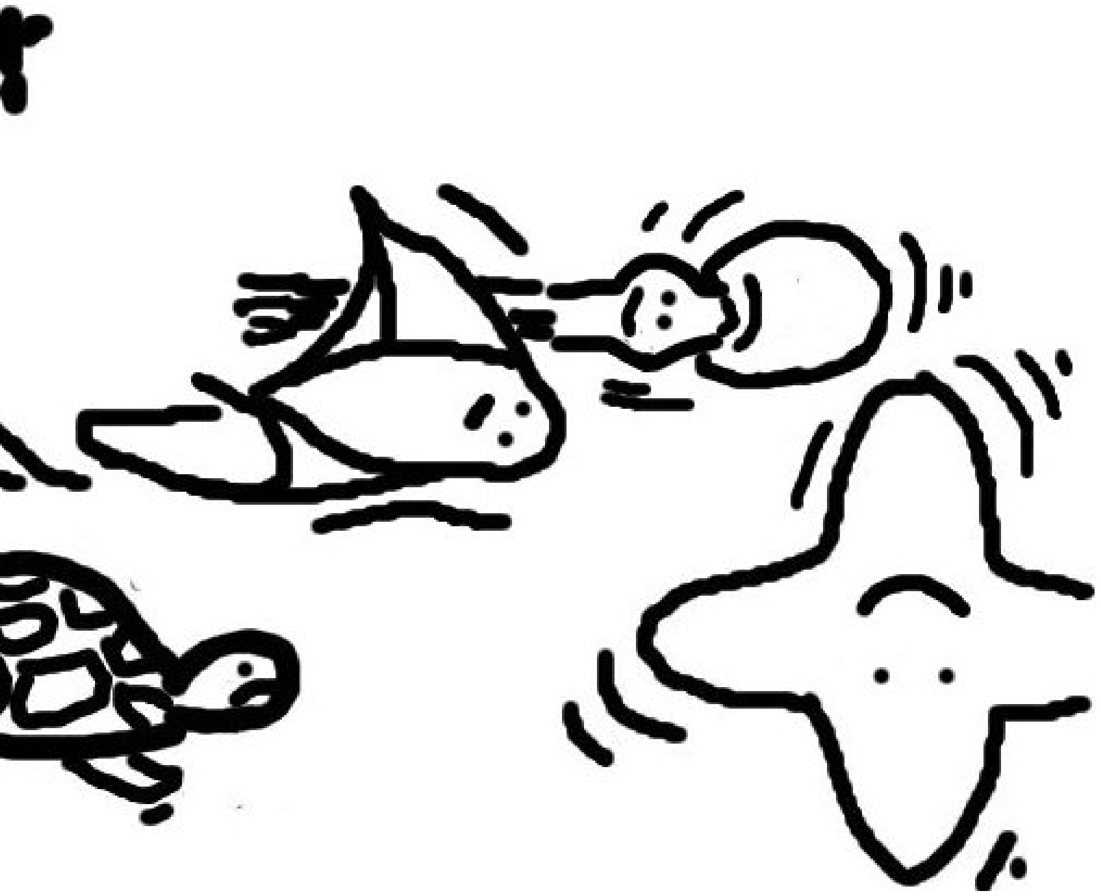


More addition formulas

Explicit-Formulas Database:
`hyperelliptic.org/EFD`

EFD has 583 computer-verified formulas and operation counts for ADD, DBL, etc.
in 51 representations
on 13 shapes of elliptic curves.

Not yet handled by computer:
generality of curve shapes
(e.g., Hessian order $\in 3\mathbf{Z}$);
complete addition algorithms
(e.g., checking for ∞).



More addition formulas

Explicit-Formulas Database:
hyperelliptic.org/EFD

EFD has 583 computer-verified formulas and operation counts for ADD, DBL, etc.
in 51 representations
on 13 shapes of elliptic curves.

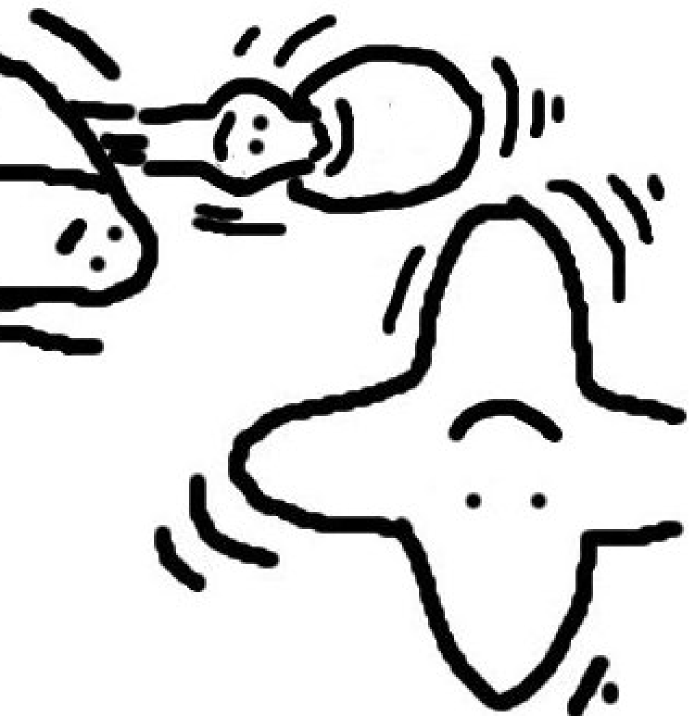
Not yet handled by computer: generality of curve shapes (e.g., Hessian order $\in 3\mathbf{Z}$); complete addition algorithms (e.g., checking for ∞).

How to

Standard
with coe
to repres

Example
$839 = 8$
value (a
$8t^2 + 3t$

Convenie
inside co
(or $9, 3,$
“p[0] =



More addition formulas

Explicit-Formulas Database:

hyperelliptic.org/EFD

EFD has 583 computer-verified formulas and operation counts for ADD, DBL, etc. in 51 representations on 13 shapes of elliptic curves.

Not yet handled by computer: generality of curve shapes (e.g., Hessian order $\in 3\mathbf{Z}$); complete addition algorithms (e.g., checking for ∞).

How to multiply b

Standard idea: Us
with coefficients in
to represent intege

Example of represe
$839 = 8 \cdot 10^2 + 3 \cdot$
value (at $t = 10$) o
$8t^2 + 3t^1 + 9t^0$.

Convenient to exp
inside computer as
(or 9, 3, 8, 0 or 9, 3
“p[0] = 9; p[1]

More addition formulas

Explicit-Formulas Database:
`hyperelliptic.org/EFD`

EFD has 583 computer-verified formulas and operation counts for ADD, DBL, etc. in 51 representations on 13 shapes of elliptic curves.

Not yet handled by computer: generality of curve shapes (e.g., Hessian order $\in 3\mathbf{Z}$); complete addition algorithms (e.g., checking for ∞).

How to multiply big integers

Standard idea: Use polynom
with coefficients in $\{0, 1, \ldots$
to represent integer in radix

Example of representation:
$839 = 8 \cdot 10^2 + 3 \cdot 10^1 + 9 \cdot$
value (at $t = 10$) of polynon
$8t^2 + 3t^1 + 9t^0$.

Convenient to express polyn
inside computer as array $9, 3$
(or $9, 3, 8, 0$ or $9, 3, 8, 0, 0$ or
"`p[0] = 9; p[1] = 3; p[2]`

More addition formulas

Explicit-Formulas Database:

`hyperelliptic.org/EFD`

EFD has 583 computer-verified formulas and operation counts for ADD, DBL, etc. in 51 representations on 13 shapes of elliptic curves.

Not yet handled by computer: generality of curve shapes (e.g., Hessian order $\in 3\mathbf{Z}$); complete addition algorithms (e.g., checking for ∞).

How to multiply big integers

Standard idea: Use polynomial with coefficients in $\{0, 1, \ldots, 9\}$ to represent integer in radix 10.

Example of representation: $839 = 8 \cdot 10^2 + 3 \cdot 10^1 + 9 \cdot 10^0 =$ value (at $t = 10$) of polynomial $8t^2 + 3t^1 + 9t^0$.

Convenient to express polynomial inside computer as array $9, 3, 8$ (or $9, 3, 8, 0$ or $9, 3, 8, 0, 0$ or $\ldots$): "`p[0] = 9; p[1] = 3; p[2] = 8`" dition formulas

Formulas Database:

lliptic.org/EFD

s 583 computer-verified

s and operation counts

), DBL, etc.

presentations

hapes of elliptic curves.

handled by computer:

ty of curve shapes

essian order $\in 3\mathbf{Z}$);

e addition algorithms

ecking for ∞).

How to multiply big integers

Standard idea: Use polynomial with coefficients in $\{0, 1, \ldots, 9\}$ to represent integer in radix 10.

Example of representation:
$839 = 8 \cdot 10^2 + 3 \cdot 10^1 + 9 \cdot 10^0 =$ value (at $t = 10$) of polynomial $8t^2 + 3t^1 + 9t^0$.

Convenient to express polynomial inside computer as array $9, 3, 8$ (or $9, 3, 8, 0$ or $9, 3, 8, 0, 0$ or $\ldots$):
"`p[0] = 9; p[1] = 3; p[2] = 8`"

Multiply

by multi

that rep

Polynom

involves

Have sp

into ma

Example

$(8t^2 + 3$

$64t^4 + 4$

mulas

Database:

org/EFD

puter-verified
ation counts
c.
ons
liptic curves.

y computer:
e shapes
er $\in 3\mathbf{Z}$);
algorithms
∞).

How to multiply big integers

Standard idea: Use polynomial with coefficients in $\{0, 1, \ldots, 9\}$ to represent integer in radix 10.

Example of representation: $839 = 8 \cdot 10^2 + 3 \cdot 10^1 + 9 \cdot 10^0 =$ value (at $t = 10$) of polynomial $8t^2 + 3t^1 + 9t^0$.

Convenient to express polynomial inside computer as array 9, 3, 8 (or 9, 3, 8, 0 or 9, 3, 8, 0, 0 or . . .): "`p[0] = 9; p[1] = 3; p[2] = 8`"

Multiply two integ
by multiplying pol
that represent the

Polynomial multip
involves *small* inte
Have split one big
into many small o

Example, squaring
$(8t^2 + 3t^1 + 9t^0)^2$
$64t^4 + 48t^3 + 153t$

fied

nts

es.

er:

s

How to multiply big integers

Standard idea: Use polynomial with coefficients in $\{0, 1, \ldots, 9\}$ to represent integer in radix 10.

Example of representation:
$839 = 8 \cdot 10^2 + 3 \cdot 10^1 + 9 \cdot 10^0 =$ value (at $t = 10$) of polynomial $8t^2 + 3t^1 + 9t^0$.

Convenient to express polynomial inside computer as array $9, 3, 8$ (or $9, 3, 8, 0$ or $9, 3, 8, 0, 0$ or $\ldots$): “`p[0] = 9; p[1] = 3; p[2] = 8`”

Multiply two integers by multiplying polynomials that represent the integers.

Polynomial multiplication involves *small* integer coeffic
Have split one big multiplica
into many small operations.

Example, squaring 839:
$(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1$

How to multiply big integers

Standard idea: Use polynomial with coefficients in $\{0, 1, \ldots, 9\}$ to represent integer in radix 10.

Example of representation: $839 = 8 \cdot 10^2 + 3 \cdot 10^1 + 9 \cdot 10^0 =$ value (at $t = 10$) of polynomial $8t^2 + 3t^1 + 9t^0$.

Convenient to express polynomial inside computer as array $9, 3, 8$ (or $9, 3, 8, 0$ or $9, 3, 8, 0, 0$ or $\ldots$): "`p[0] = 9; p[1] = 3; p[2] = 8`"

Multiply two integers by multiplying polynomials that represent the integers.

Polynomial multiplication involves *small* integer coefficients. Have split one big multiplication into many small operations.

Example, squaring 839:
$(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$.

multiply big integers

d idea: Use polynomial

efficients in $\{0, 1, \ldots, 9\}$

sent integer in radix 10.

e of representation:

$\cdot 10^2 + 3 \cdot 10^1 + 9 \cdot 10^0 =$

t $t = 10$) of polynomial

$^1 + 9t^0$.

ent to express polynomial

omputer as array 9, 3, 8

8, 0 or 9, 3, 8, 0, 0 or . . .):

9; p[1] = 3; p[2] = 8”

Multiply two integers

by multiplying polynomials

that represent the integers.

Polynomial multiplication

involves *small* integer coefficients.

Have split one big multiplication

into many small operations.

Example, squaring 839:

$(8t^2 + 3t^1 + 9t^0)^2 =$

$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$.

Oops, pr

usually h

So “carr

$ct^j \to \lfloor c$

Example

$64t^4 + 4$

$64t^4 + 4$

$64t^4 + 4$

$64t^4 + 6$

$70t^4 + 3$

$7t^5 + 0t$

In other

ig integers

e polynomial
n $\{0, 1, \ldots, 9\}$
er in radix 10.

entation:
$10^1 + 9 \cdot 10^0 =$
of polynomial

ress polynomial
s array $9, 3, 8$
$3, 8, 0, 0$ or $\ldots$):
= 3; p[2] = 8"

Multiply two integers
by multiplying polynomials
that represent the integers.

Polynomial multiplication
involves *small* integer coefficients.
Have split one big multiplication
into many small operations.

Example, squaring 839:
$(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$.

Oops, product pol
usually has coeffic
So "carry" extra d
$ct^j \to \lfloor c/10 \rfloor t^{j+1}$

Example, squaring
$64t^4 + 48t^3 + 153t$
$64t^4 + 48t^3 + 153$
$64t^4 + 48t^3 + 159$
$64t^4 + 63t^3 + 9t^2$
$70t^4 + 3t^3 + 9t^2 +$
$7t^5 + 0t^4 + 3t^3 +$

In other words, 83

s

ial

$, 9\}$

10.

$10^0 =$

nial

omial

$3, 8$

$\ldots):$

= 8”

Multiply two integers
by multiplying polynomials
that represent the integers.

Polynomial multiplication
involves *small* integer coefficients.
Have split one big multiplication
into many small operations.

Example, squaring 839:

$(8t^2 + 3t^1 + 9t^0)^2 =$

$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0.$

Oops, product polynomial
usually has coefficients > 9.
So “carry” extra digits:

$ct^j \to \lfloor c/10 \rfloor t^{j+1} + (c \bmod$

Example, squaring 839:

$64t^4 + 48t^3 + 153t^2 + 54t^1 +$

$64t^4 + 48t^3 + 153t^2 + 62t^1$

$64t^4 + 48t^3 + 159t^2 + 2t^1 +$

$64t^4 + 63t^3 + 9t^2 + 2t^1 + 1$

$70t^4 + 3t^3 + 9t^2 + 2t^1 + 1t$

$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1$

In other words, $839^2 = 7039$

Multiply two integers
by multiplying polynomials
that represent the integers.

Polynomial multiplication
involves *small* integer coefficients.
Have split one big multiplication
into many small operations.

Example, squaring 839:
$(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$.

Oops, product polynomial
usually has coefficients > 9.
So "carry" extra digits:
$ct^j \to \lfloor c/10 \rfloor t^{j+1} + (c \bmod 10)t^j$.

Example, squaring 839:
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;
$64t^4 + 48t^3 + 153t^2 + 62t^1 + 1t^0$;
$64t^4 + 48t^3 + 159t^2 + 2t^1 + 1t^0$;
$64t^4 + 63t^3 + 9t^2 + 2t^1 + 1t^0$;
$70t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$;
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

In other words, $839^2 = 703921$.

two integers
plying polynomials
resent the integers.

nial multiplication
small integer coefficients.
lit one big multiplication
ny small operations.

e, squaring 839:
$t^1 + 9t^0)^2 =$
$8t^3 + 153t^2 + 54t^1 + 81t^0$.

Oops, product polynomial
usually has coefficients > 9.
So "carry" extra digits:
$ct^j \to \lfloor c/10 \rfloor t^{j+1} + (c \bmod 10)t^j$.

Example, squaring 839:
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;
$64t^4 + 48t^3 + 153t^2 + 62t^1 + 1t^0$;
$64t^4 + 48t^3 + 159t^2 + 2t^1 + 1t^0$;
$64t^4 + 63t^3 + 9t^2 + 2t^1 + 1t^0$;
$70t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$;
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

In other words, $839^2 = 703921$.

What op

divide b

gers
ynomials
integers.

lication
eger coefficients.
multiplication
perations.

839:

$=$

$t^2 + 54t^1 + 81t^0$.

Oops, product polynomial usually has coefficients > 9.

So "carry" extra digits:

$$ct^j \to \lfloor c/10 \rfloor \, t^{j+1} + (c \bmod 10)t^j.$$

Example, squaring 839:

$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;

$64t^4 + 48t^3 + 153t^2 + 62t^1 + 1t^0$;

$64t^4 + 48t^3 + 159t^2 + 2t^1 + 1t^0$;

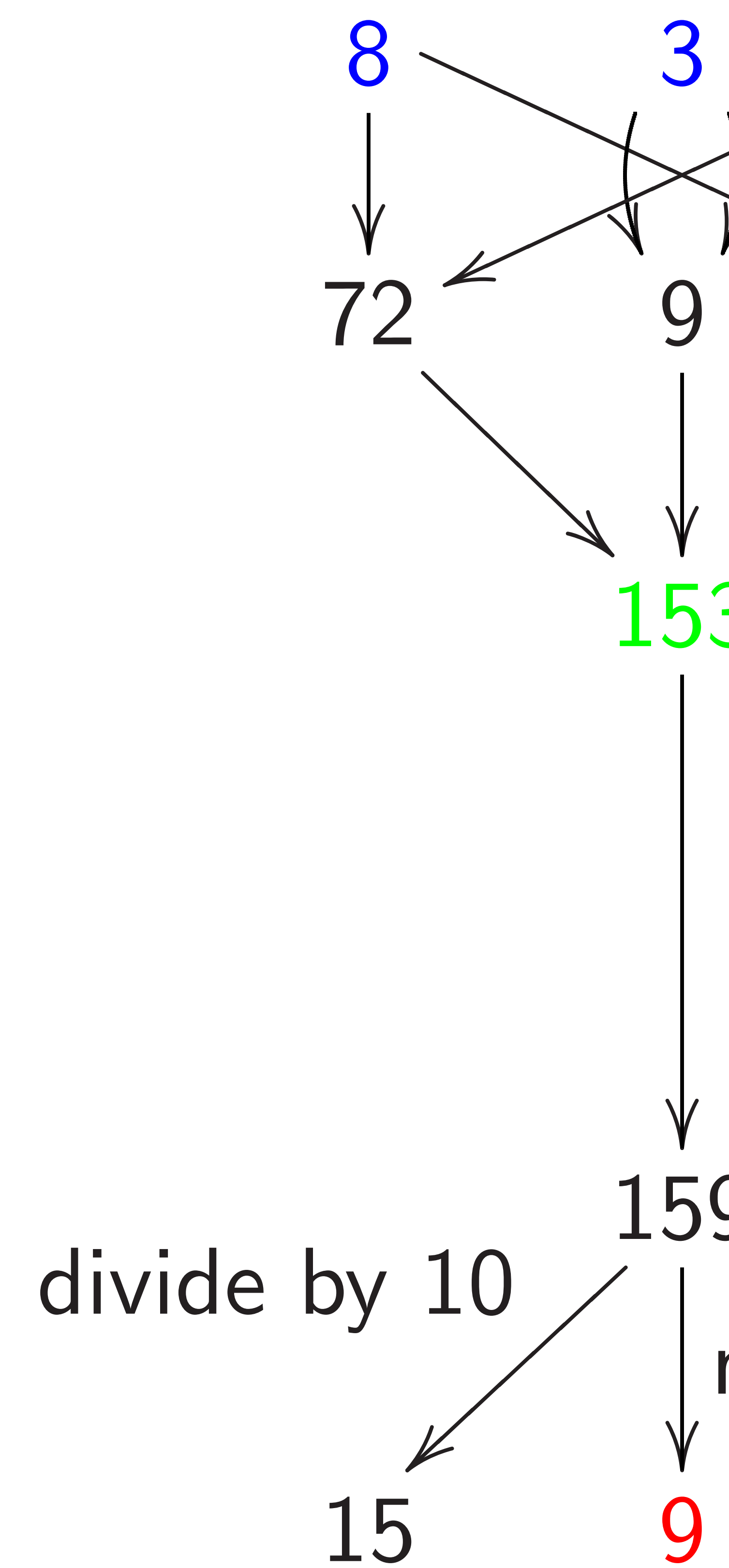
$64t^4 + 63t^3 + 9t^2 + 2t^1 + 1t^0$;

$70t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$;

$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

In other words, $839^2 = 703921$.

What operations w



cients.

ation

$+\ 81t^0.$

Oops, product polynomial usually has coefficients > 9.

So "carry" extra digits:

$ct^j \to \lfloor c/10 \rfloor\, t^{j+1} + (c \bmod 10)t^j.$

Example, squaring 839:

$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0;$

$64t^4 + 48t^3 + 153t^2 + 62t^1 + 1t^0;$

$64t^4 + 48t^3 + 159t^2 + 2t^1 + 1t^0;$

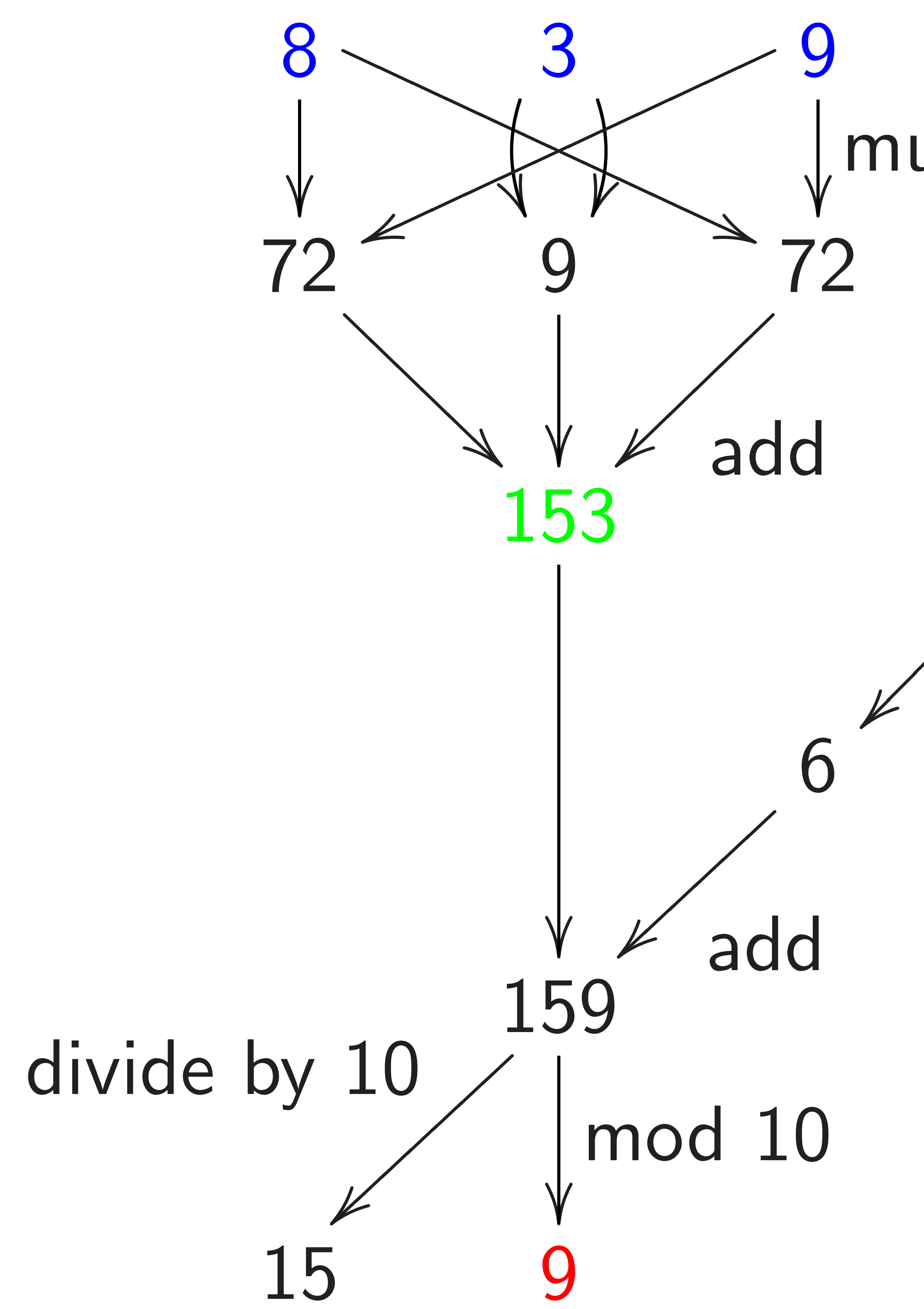
$64t^4 + 63t^3 + 9t^2 + 2t^1 + 1t^0;$

$70t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0;$

$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0.$

In other words, $839^2 = 703921$.

What operations were used



Oops, product polynomial usually has coefficients > 9.

So "carry" extra digits:

$ct^j \to \lfloor c/10 \rfloor t^{j+1} + (c \bmod 10)t^j$.

Example, squaring 839:

$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;

$64t^4 + 48t^3 + 153t^2 + 62t^1 + 1t^0$;

$64t^4 + 48t^3 + 159t^2 + 2t^1 + 1t^0$;

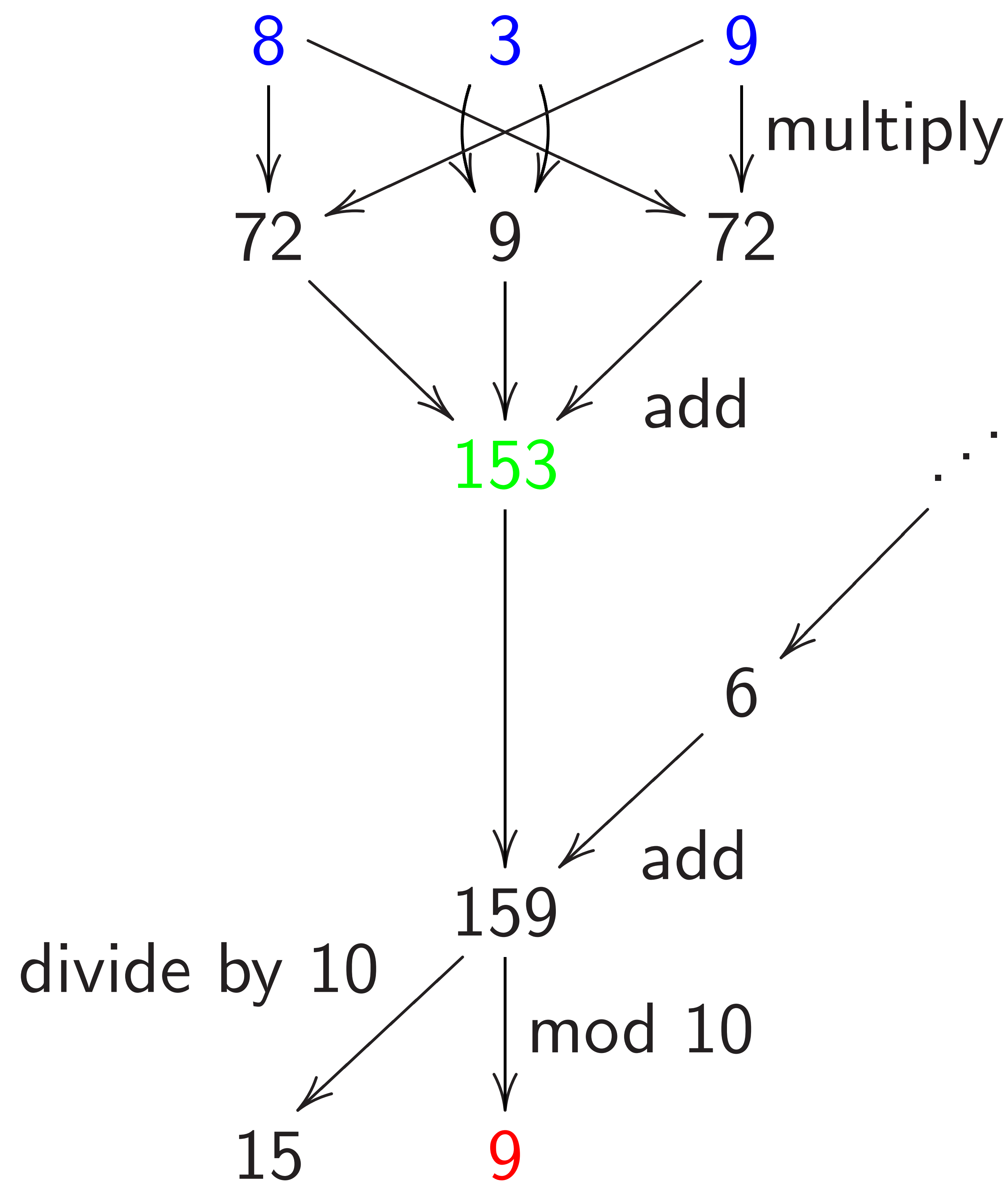
$64t^4 + 63t^3 + 9t^2 + 2t^1 + 1t^0$;

$70t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$;

$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

In other words, $839^2 = 703921$.

What operations were used here?



roduct polynomial

has coefficients > 9.

y" extra digits:

$c/10\rfloor\, t^{j+1} + (c \bmod 10)t^j$.

e, squaring 839:

$8t^3 + 153t^2 + 54t^1 + 81t^0$;

$8t^3 + 153t^2 + 62t^1 + 1t^0$;

$8t^3 + 159t^2 + 2t^1 + 1t^0$;

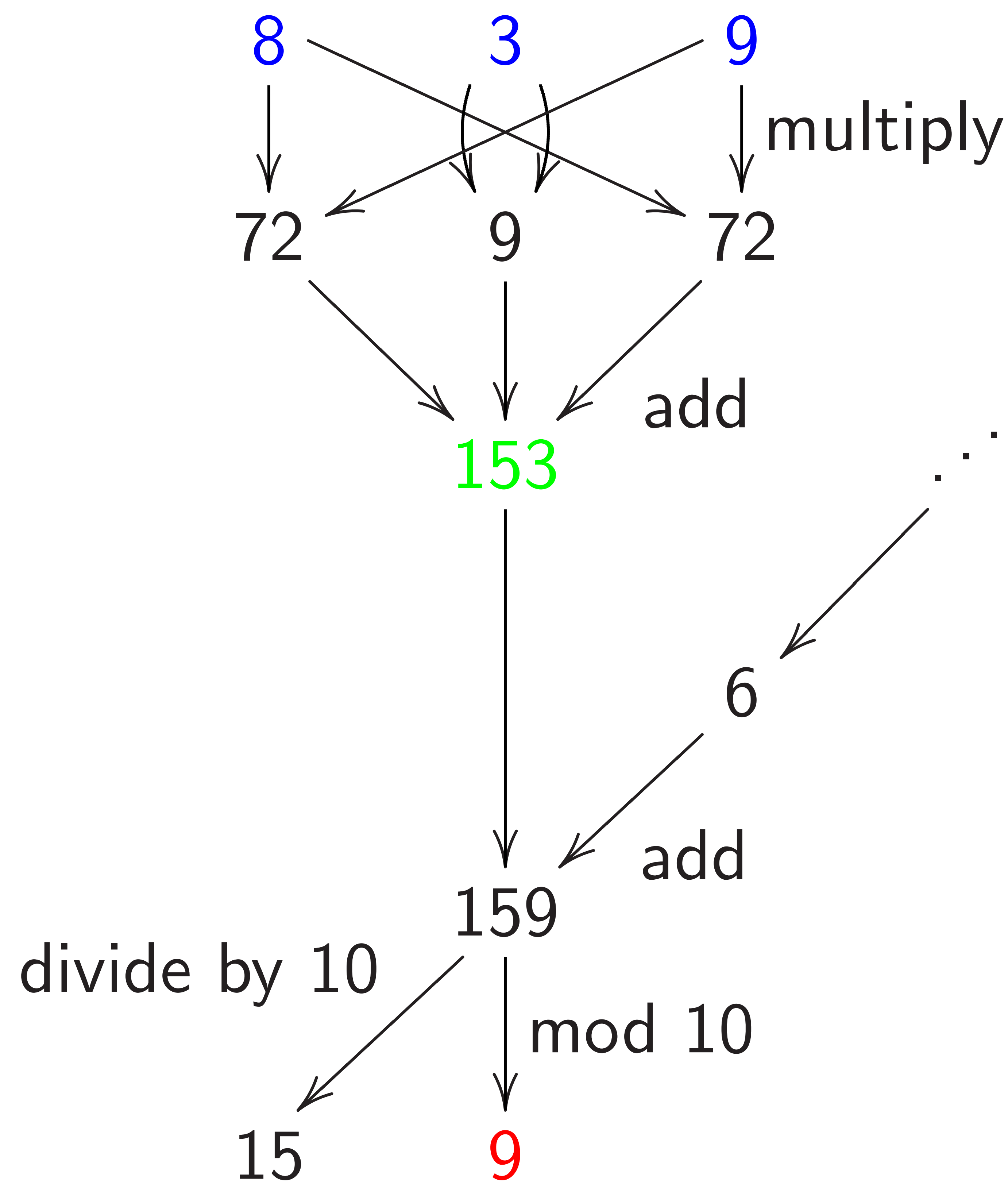
$63t^3 + 9t^2 + 2t^1 + 1t^0$;

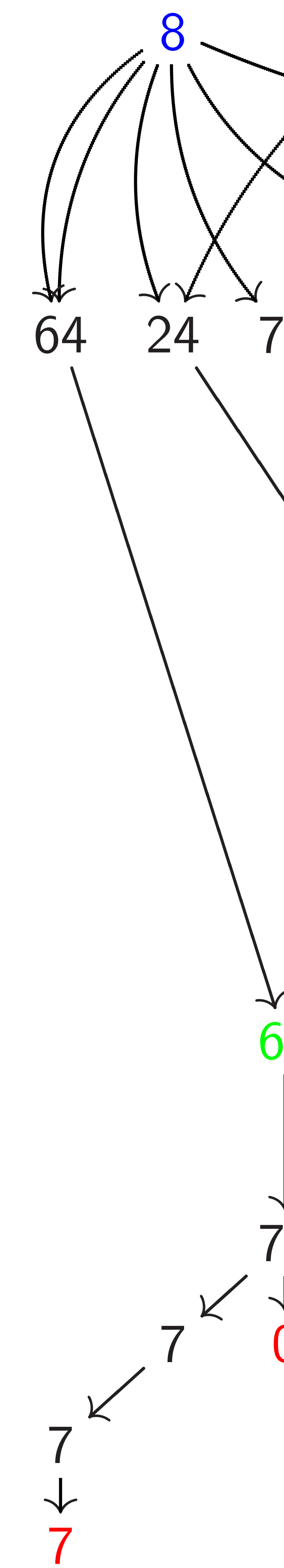
$t^3 + 9t^2 + 2t^1 + 1t^0$;

$^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

words, $839^2 = 703921$.

What operations were used here?





ynomial

ients > 9.

ligits:

$+ (c \bmod 10)t^j$.

839:

$t^2 + 54t^1 + 81t^0$;

$t^2 + 62t^1 + 1t^0$;

$t^2 + 2t^1 + 1t^0$;

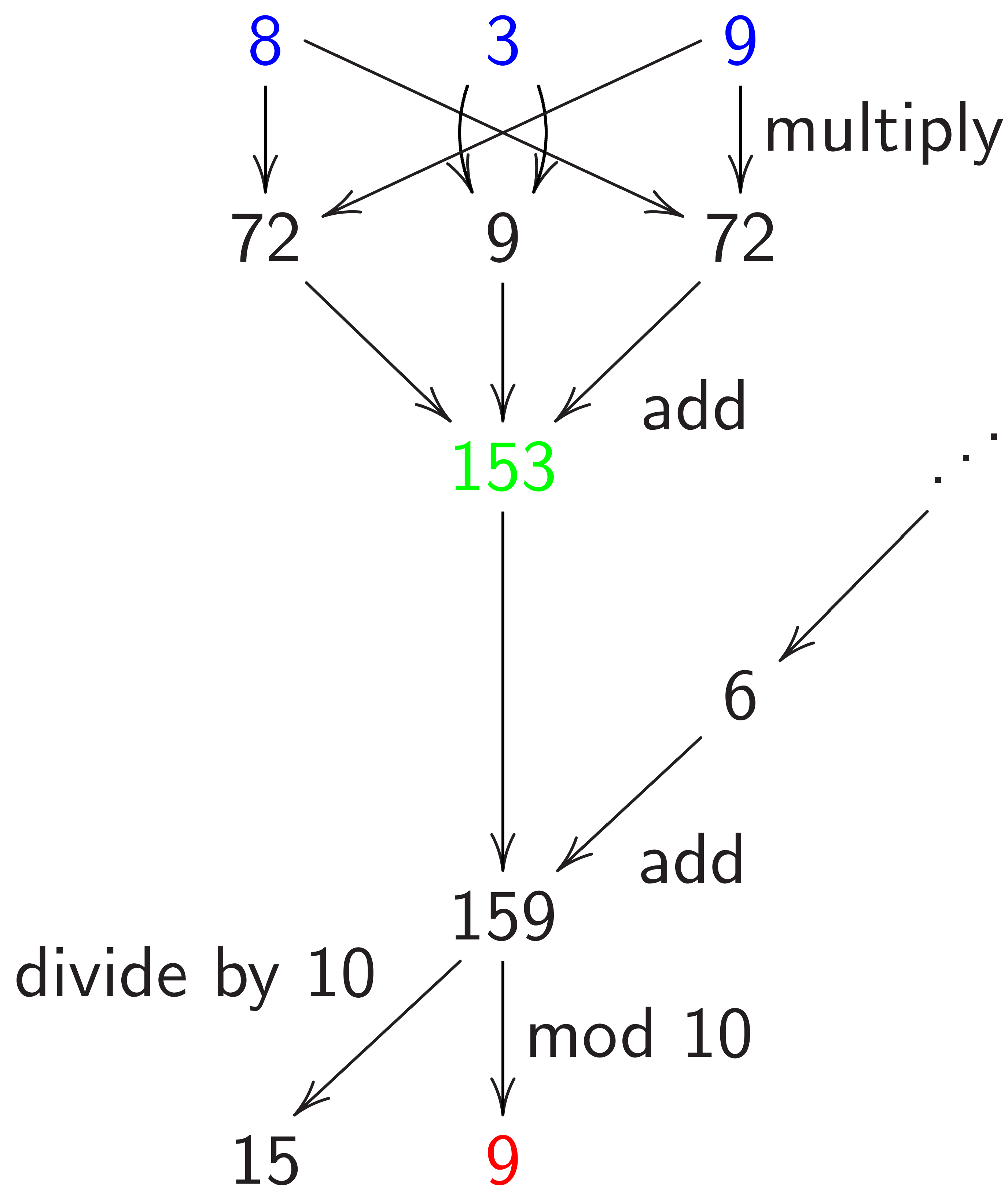
$+ 2t^1 + 1t^0$;

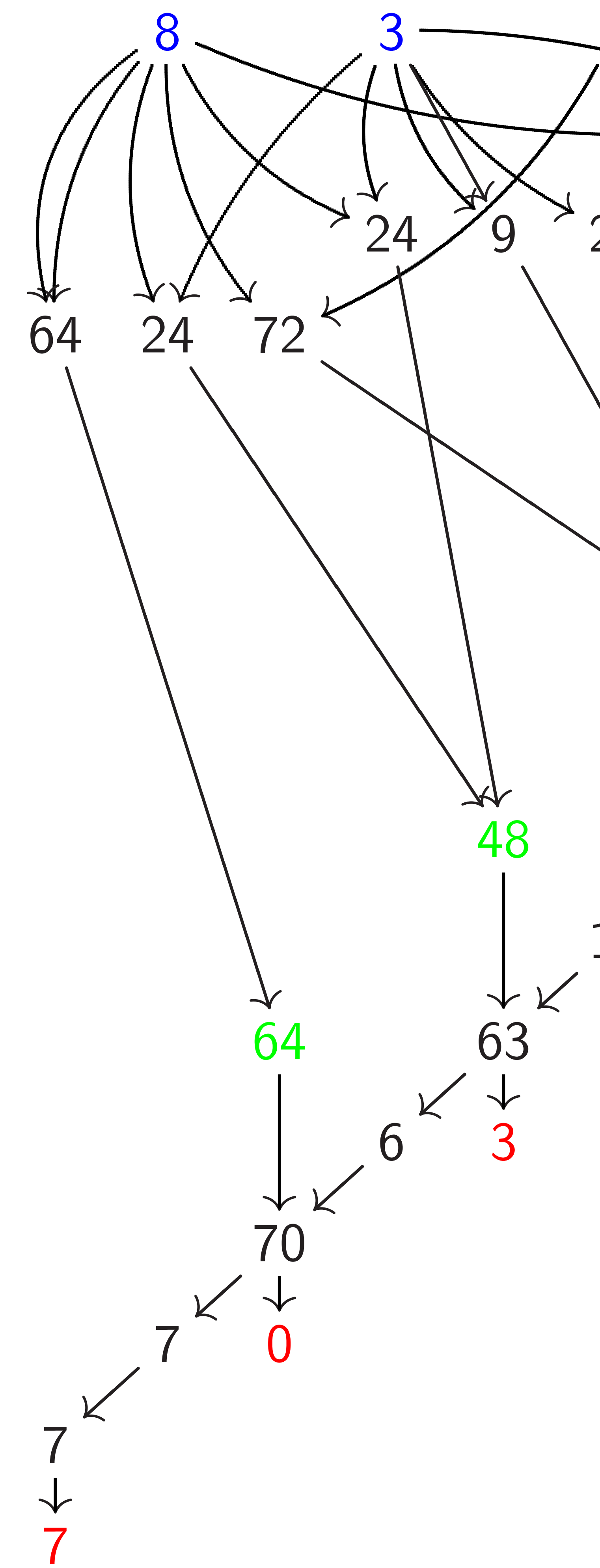
$- 2t^1 + 1t^0$;

$9t^2 + 2t^1 + 1t^0$.

$9^2 = 703921$.

What operations were used here?





$10)t^j$.

$+\ 81t^0$;

$+\ 1t^0$;

$-\ 1t^0$;

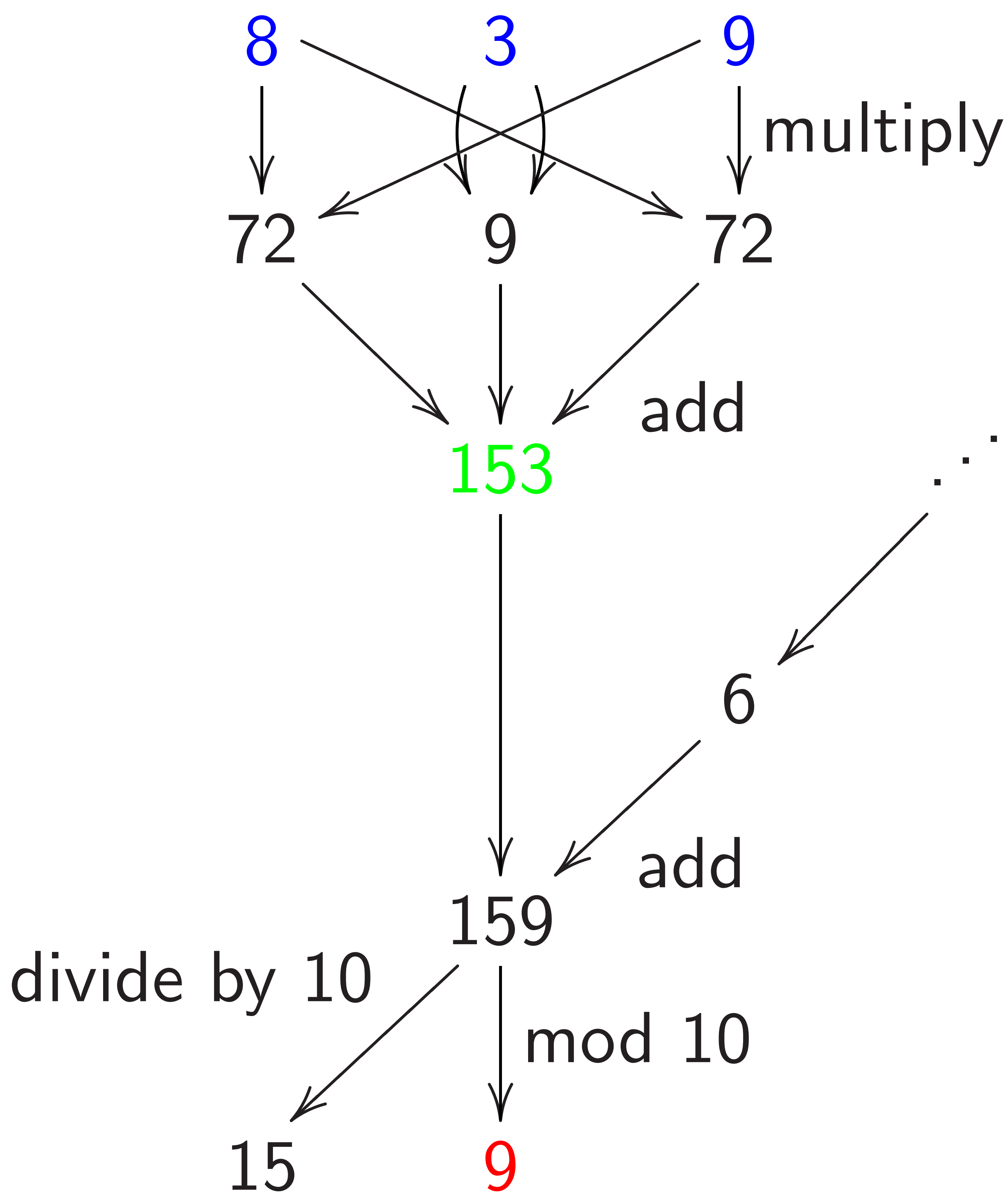
t^0;

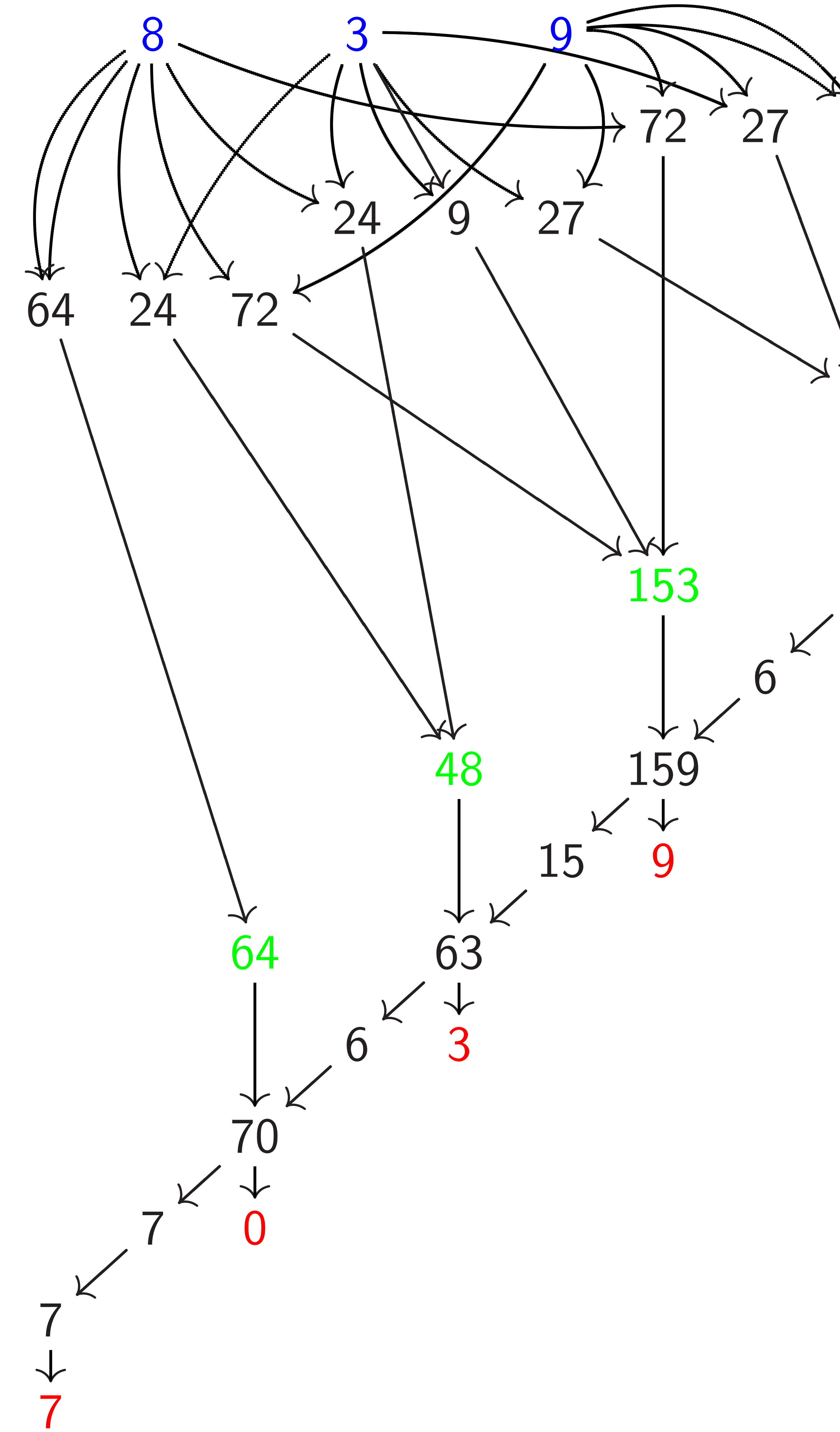
^{0};

$+\ 1t^0$.

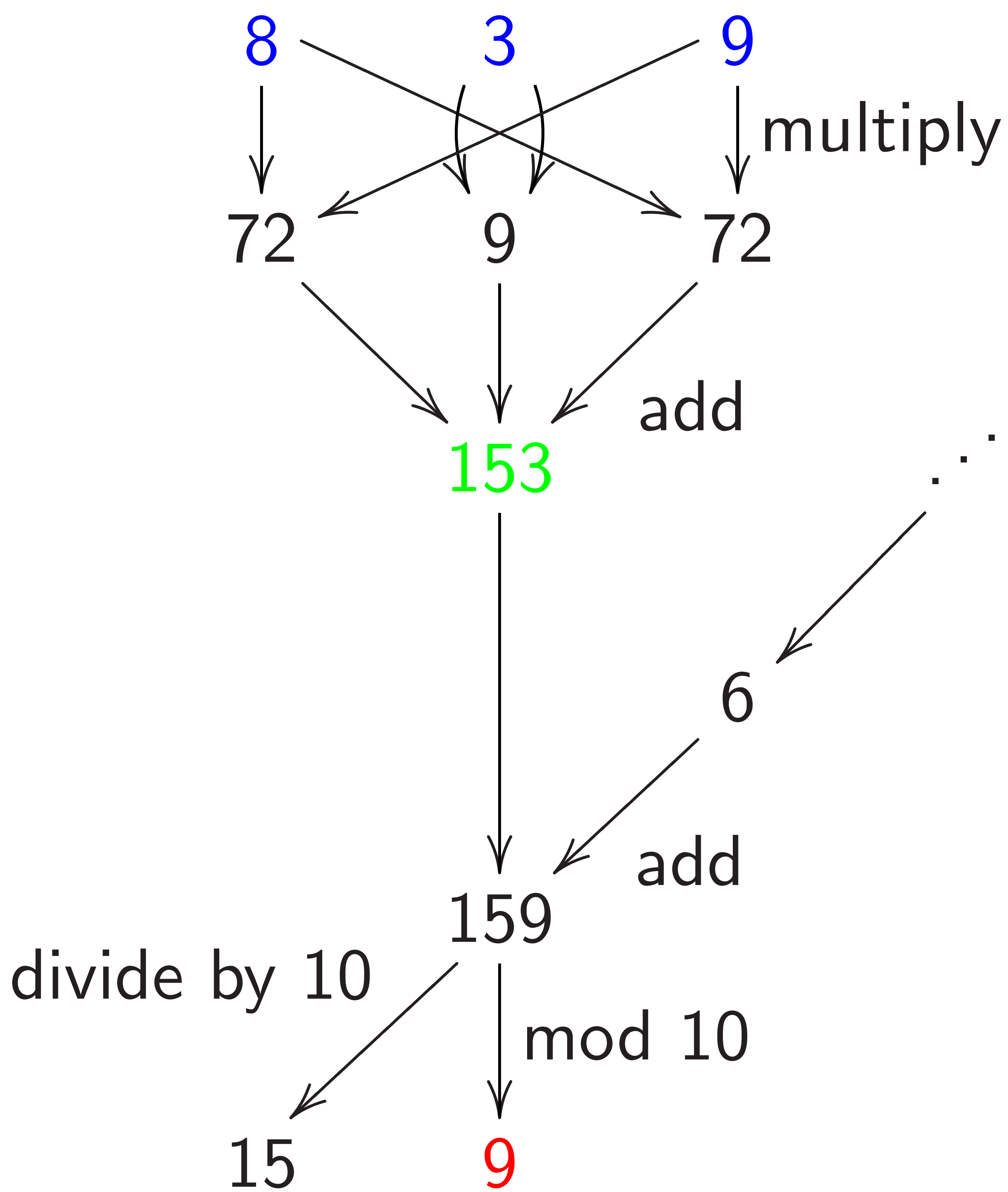
921.

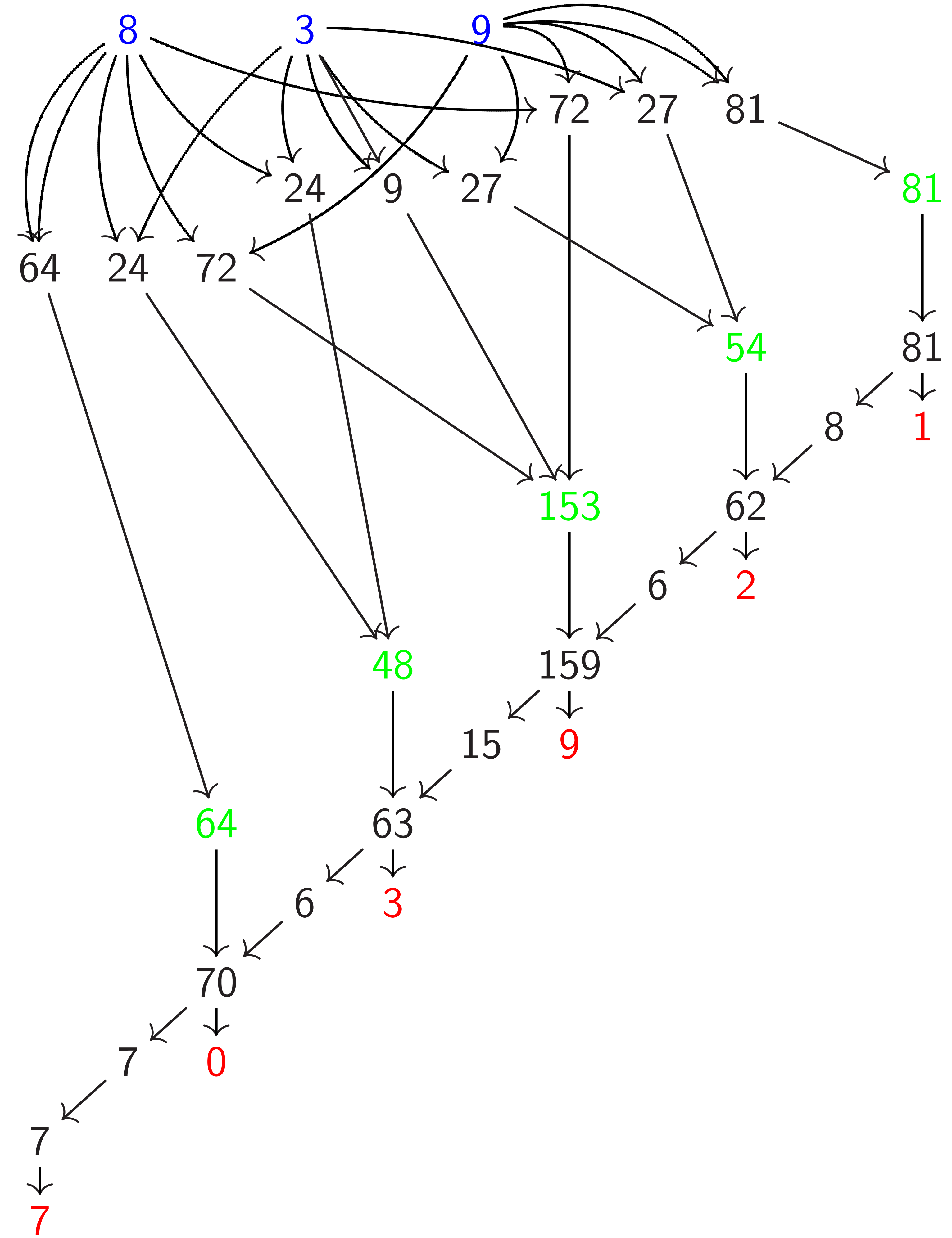
What operations were used here?



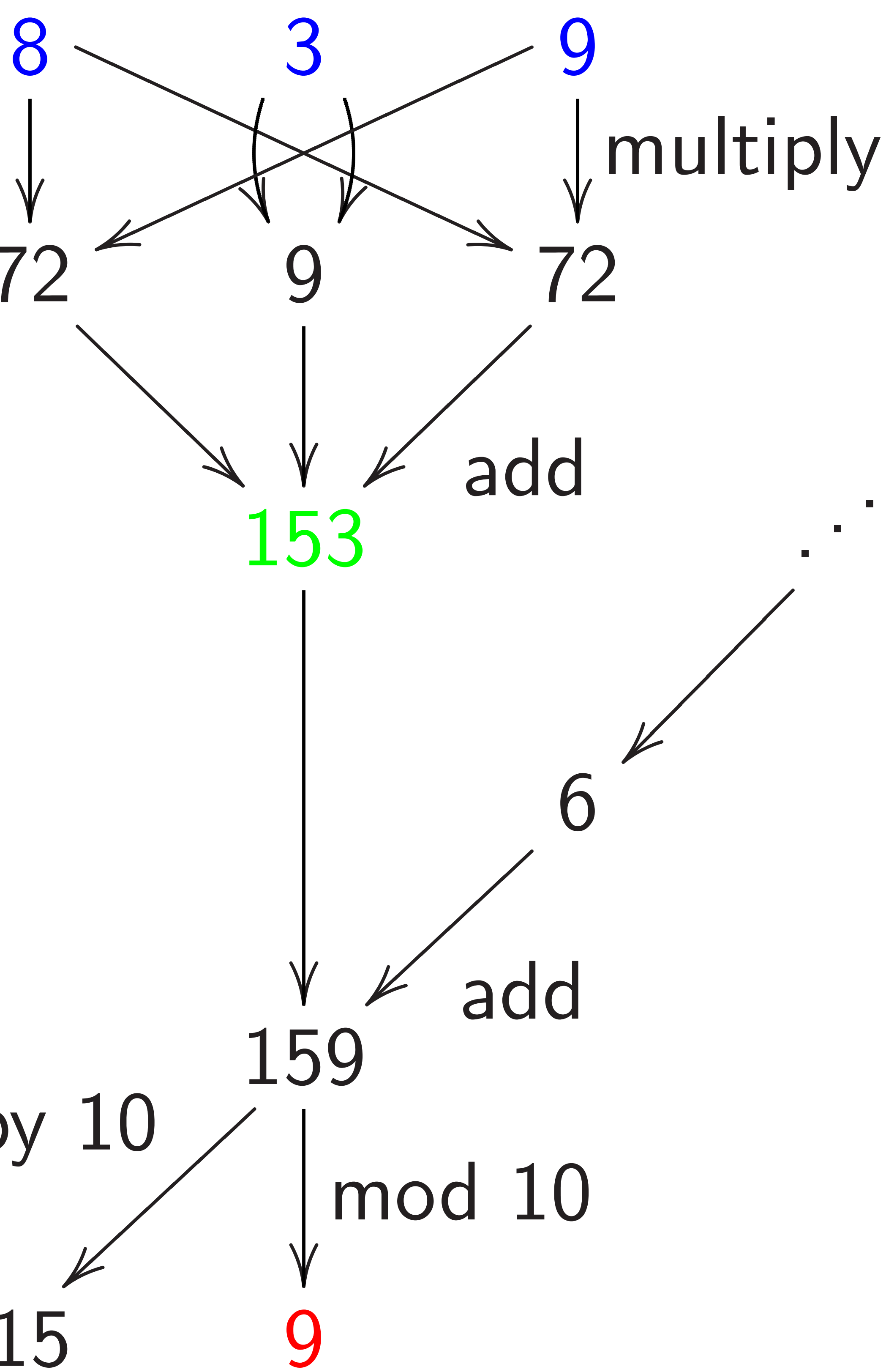


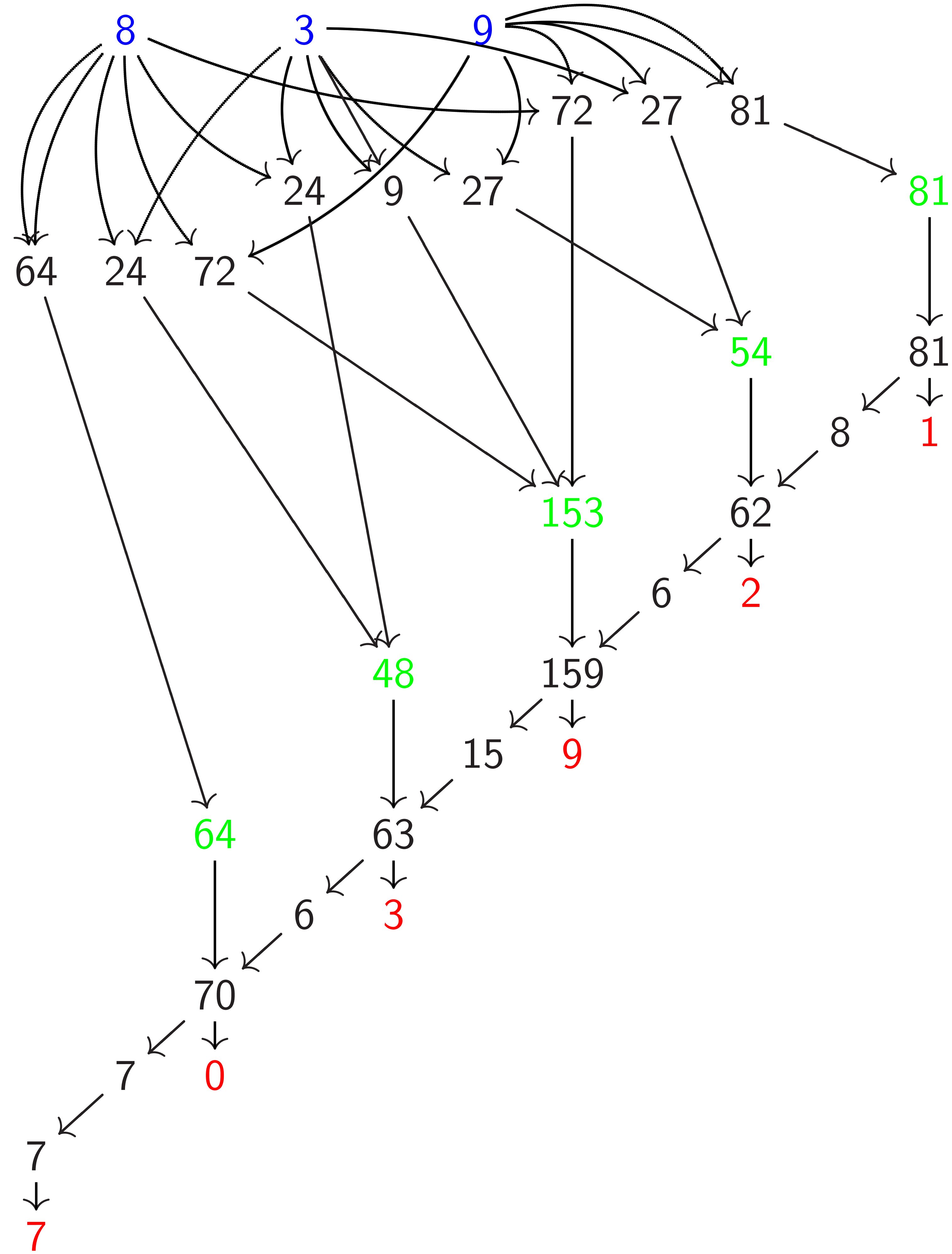
What operations were used here?





perations were used here?





The scal

839 = 8

value (a

$800t^2 +$

Squaring

640000t

$540t^1 +$

Carrying

640000t

$540t^1 +$

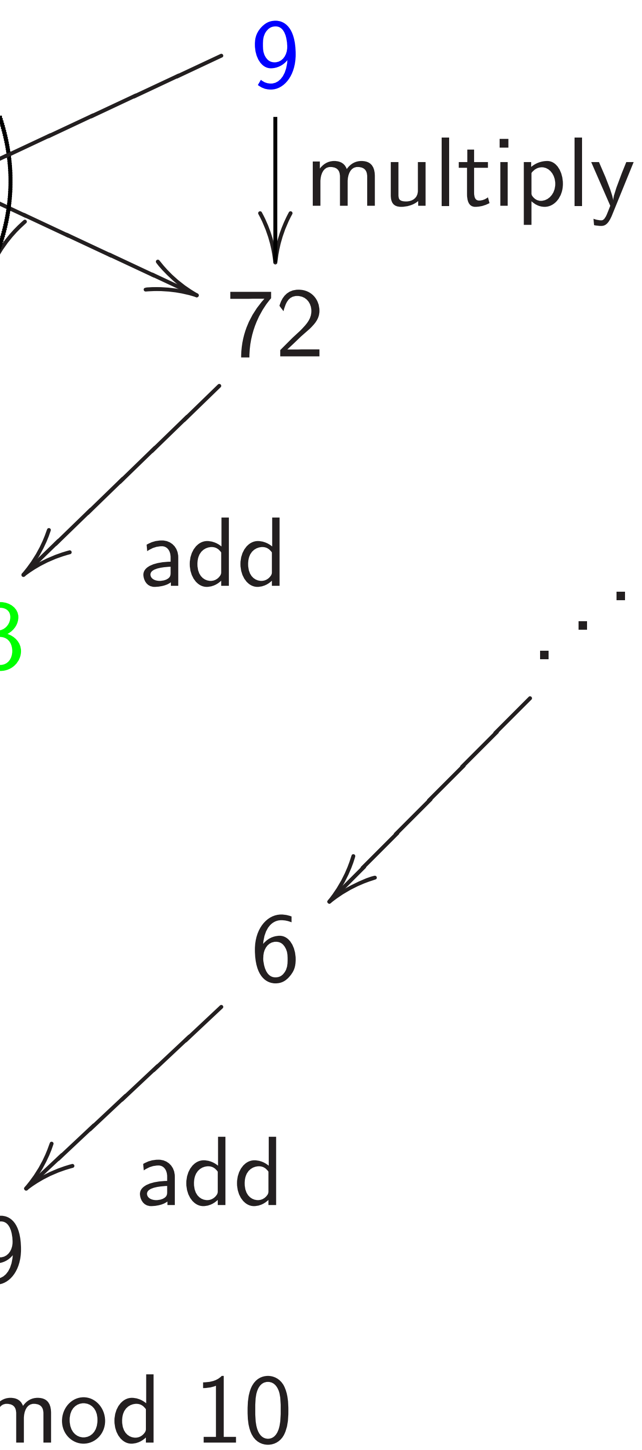
640000t

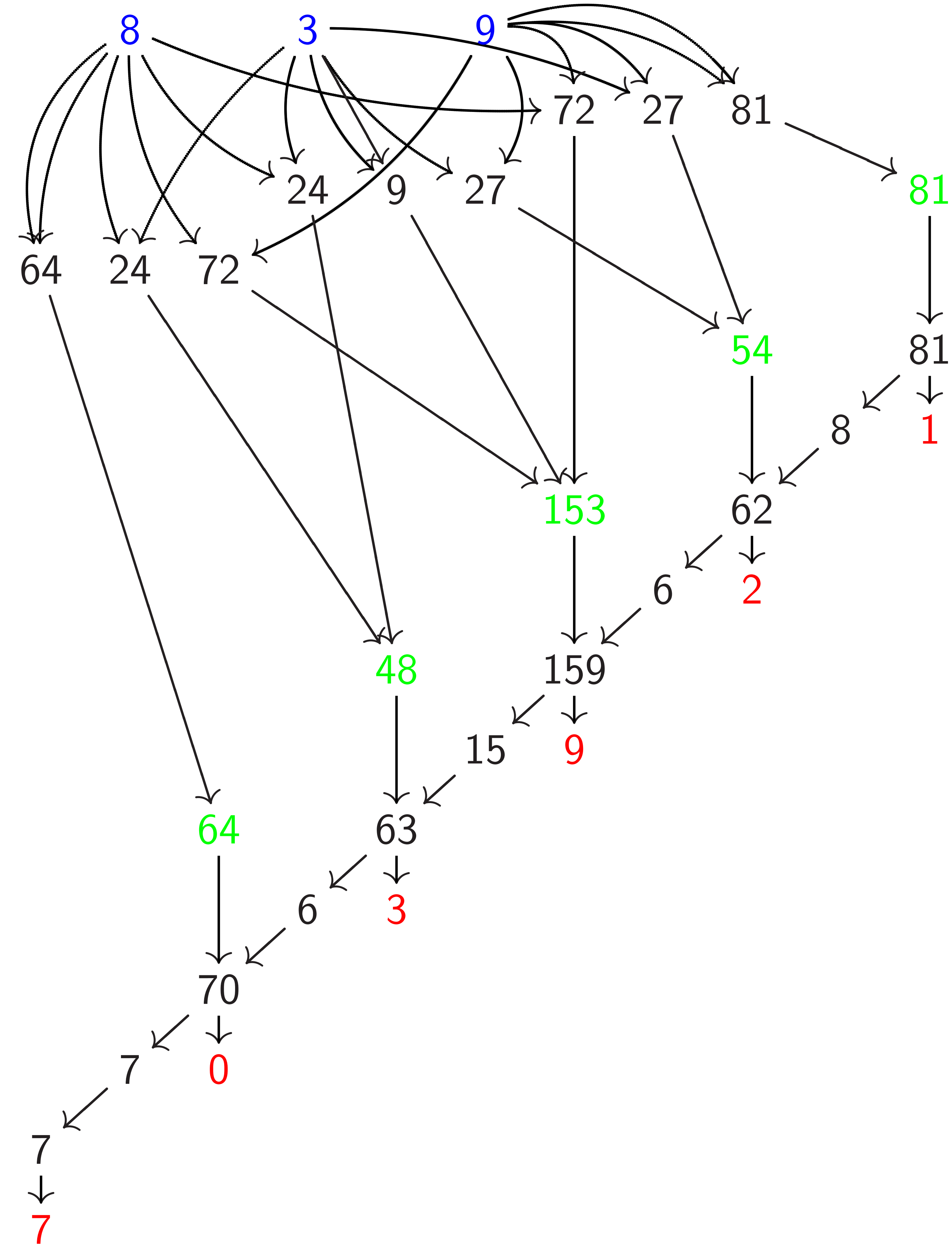
$620t^1 +$

700000t

$20t^1 + 1$

were used here?





<u>The scaled variatio</u>

$839 = 800 + 30 +$

value (at $t = 1$) of

$800t^2 + 30t^1 + 9t^0$

Squaring: $(800t^2 -$

$640000t^4 + 48000$

$540t^1 + 81t^0.$

Carrying:

$640000t^4 + 48000$

$540t^1 + 81t^0;$

$640000t^4 + 48000$

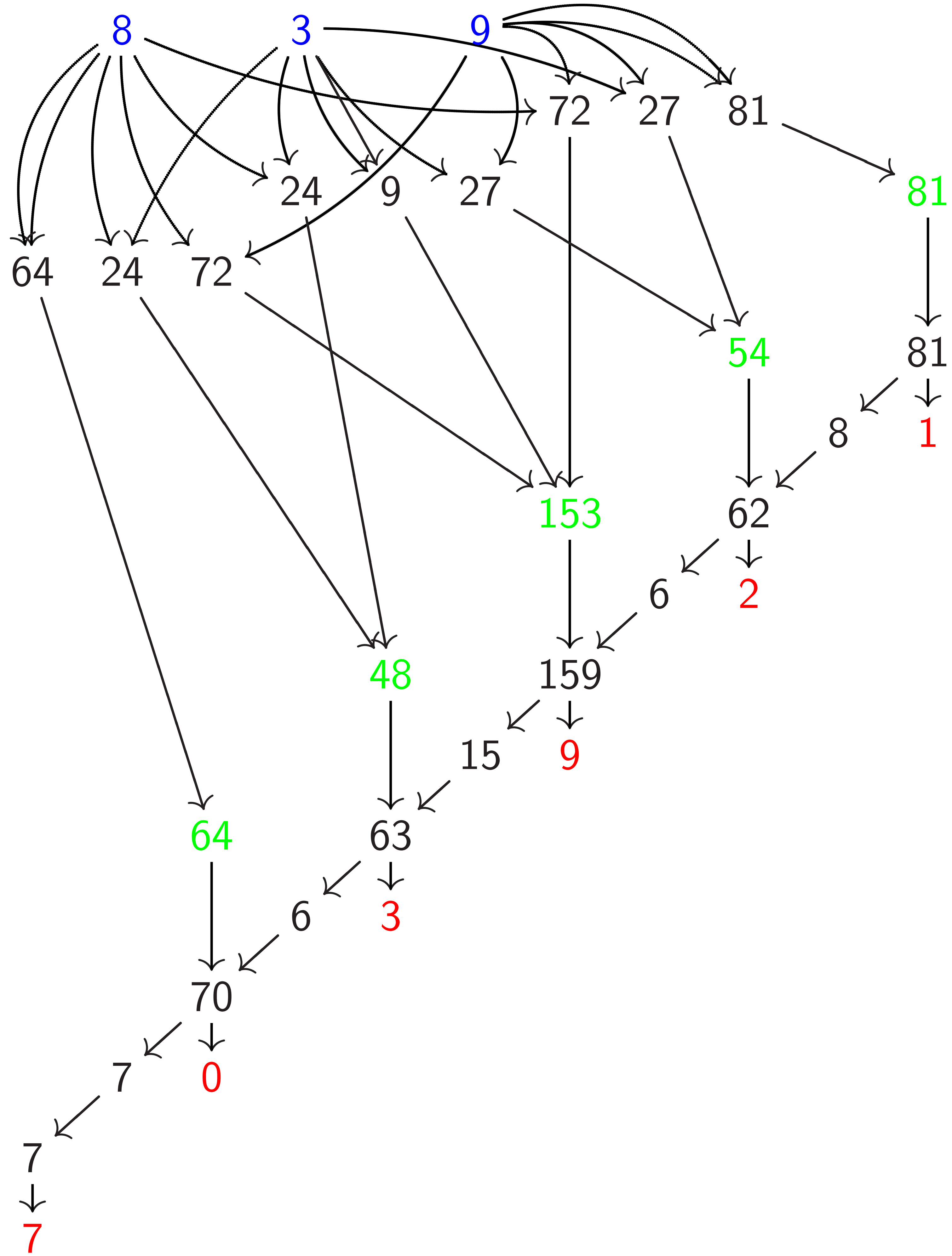
$620t^1 + 1t^0;$.

$700000t^5 + 0t^4 + 3$

$20t^1 + 1t^0.$

here?

ultiply



The scaled variation

$839 = 800 + 30 + 9 =$

value (at $t = 1$) of polynom

$800t^2 + 30t^1 + 9t^0$.

Squaring: $(800t^2 + 30t^1 + 9t$

$640000t^4 + 48000t^3 + 1530$

$540t^1 + 81t^0$.

Carrying:

$640000t^4 + 48000t^3 + 1530$

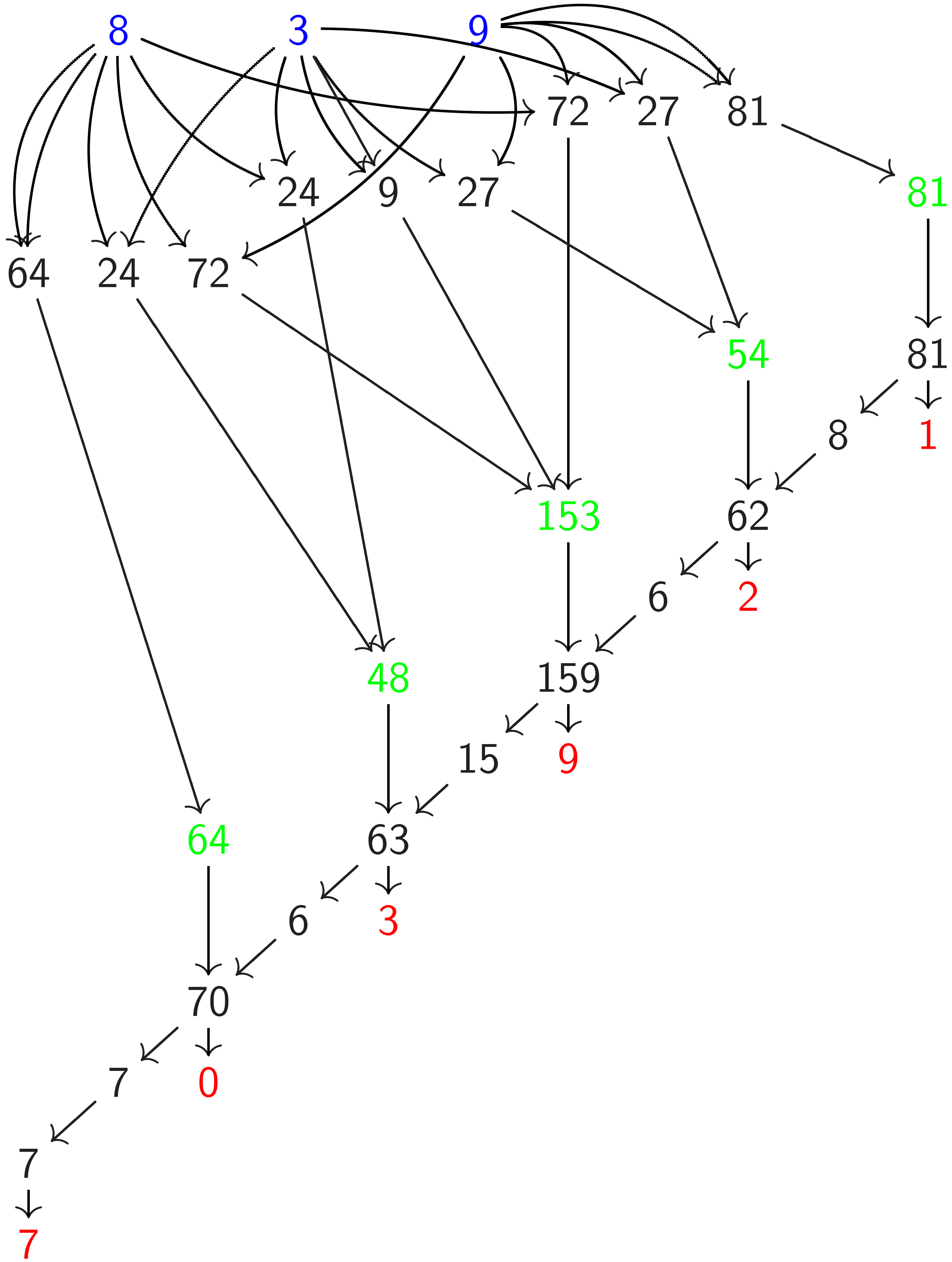
$540t^1 + 81t^0$;

$640000t^4 + 48000t^3 + 1530$

$620t^1 + 1t^0$; $\ldots$

$700000t^5 + 0t^4 + 3000t^3 + 9$

$20t^1 + 1t^0$.

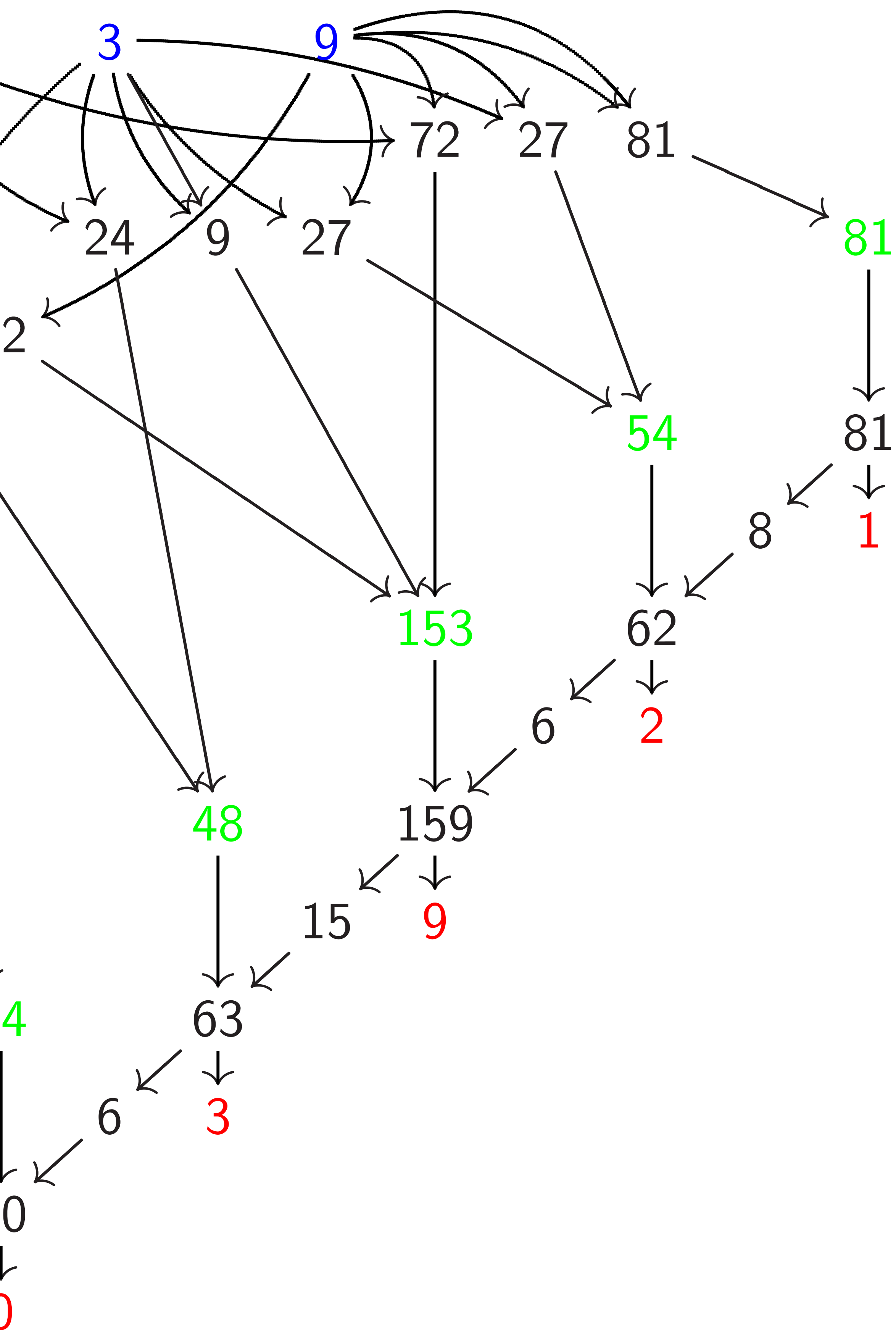


The scaled variation

$839 = 800 + 30 + 9 =$
value (at $t = 1$) of polynomial
$800t^2 + 30t^1 + 9t^0$.

Squaring: $(800t^2 + 30t^1 + 9t^0)^2 =$
$640000t^4 + 48000t^3 + 15300t^2 + 540t^1 + 81t^0$.

Carrying:
$640000t^4 + 48000t^3 + 15300t^2 + 540t^1 + 81t^0$;
$640000t^4 + 48000t^3 + 15300t^2 + 620t^1 + 1t^0$; $\quad\ldots$
$700000t^5 + 0t^4 + 3000t^3 + 900t^2 + 20t^1 + 1t^0$.



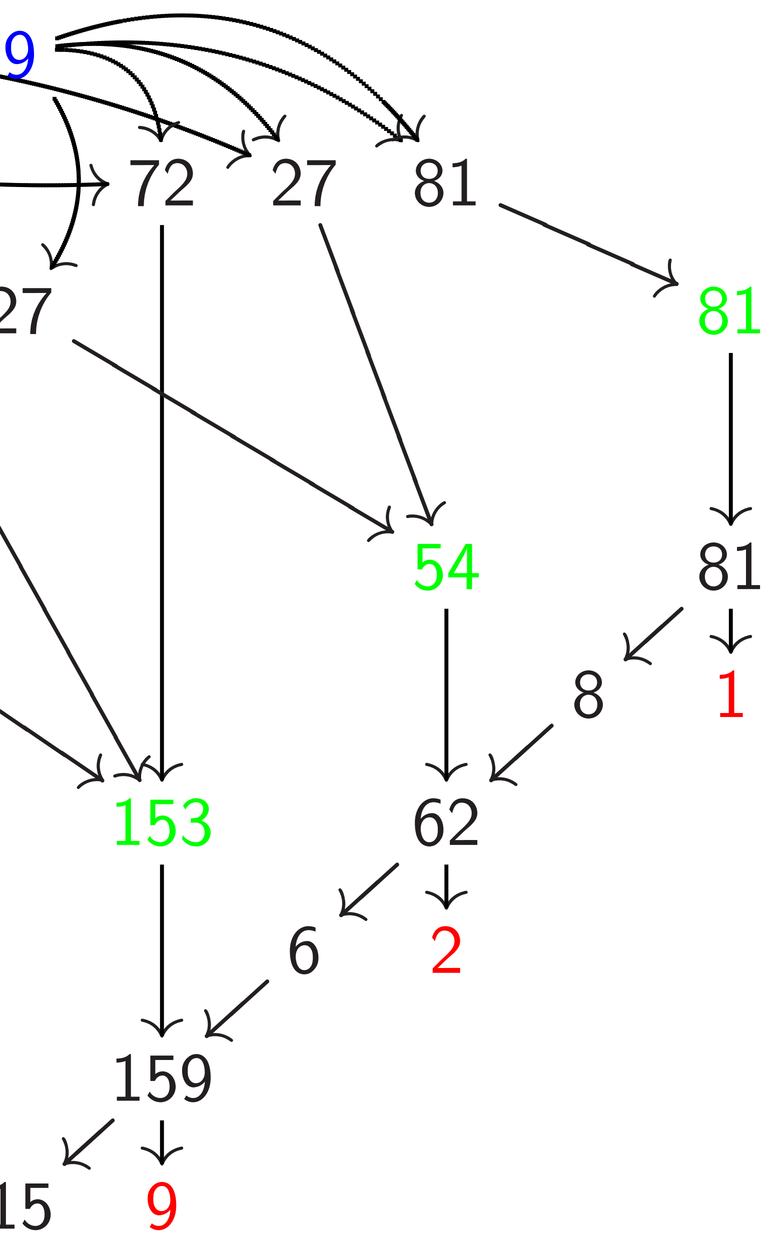
The scaled variation

$839 = 800 + 30 + 9 =$
value (at $t = 1$) of polynomial
$800t^2 + 30t^1 + 9t^0$.

Squaring: $(800t^2 + 30t^1 + 9t^0)^2 =$
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$.

Carrying:
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$;
$640000t^4 + 48000t^3 + 15300t^2 +$
$620t^1 + 1t^0$; ...
$700000t^5 + 0t^4 + 3000t^3 + 900t^2 +$
$20t^1 + 1t^0$.

What op



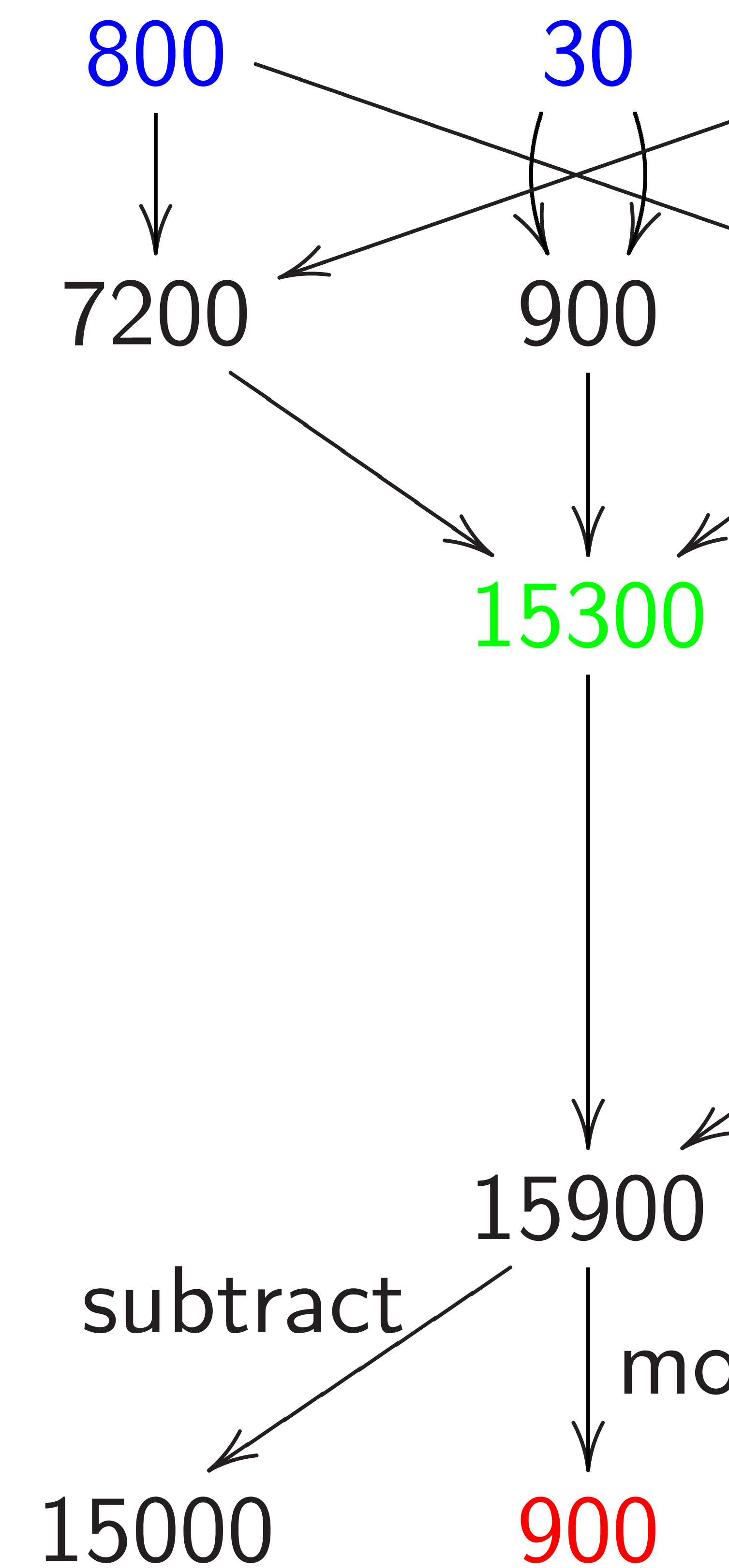
The scaled variation

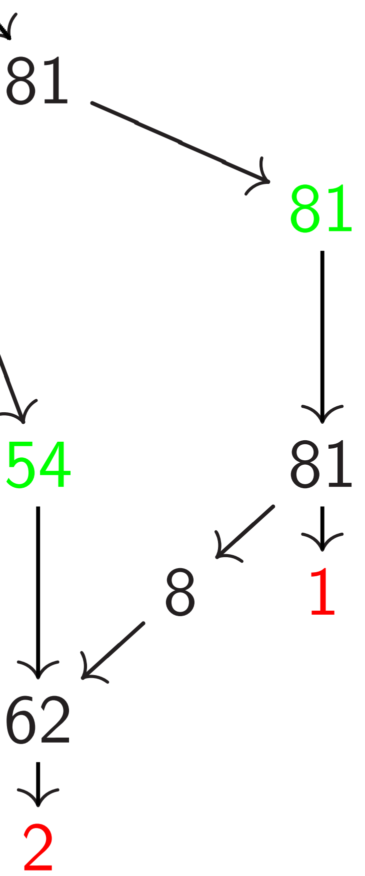
$839 = 800 + 30 + 9 =$
value (at $t = 1$) of polynomial
$800t^2 + 30t^1 + 9t^0$.

Squaring: $(800t^2 + 30t^1 + 9t^0)^2 =$
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$.

Carrying:
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$;
$640000t^4 + 48000t^3 + 15300t^2 +$
$620t^1 + 1t^0$; $\quad \ldots$
$700000t^5 + 0t^4 + 3000t^3 + 900t^2 +$
$20t^1 + 1t^0$.

What operations w





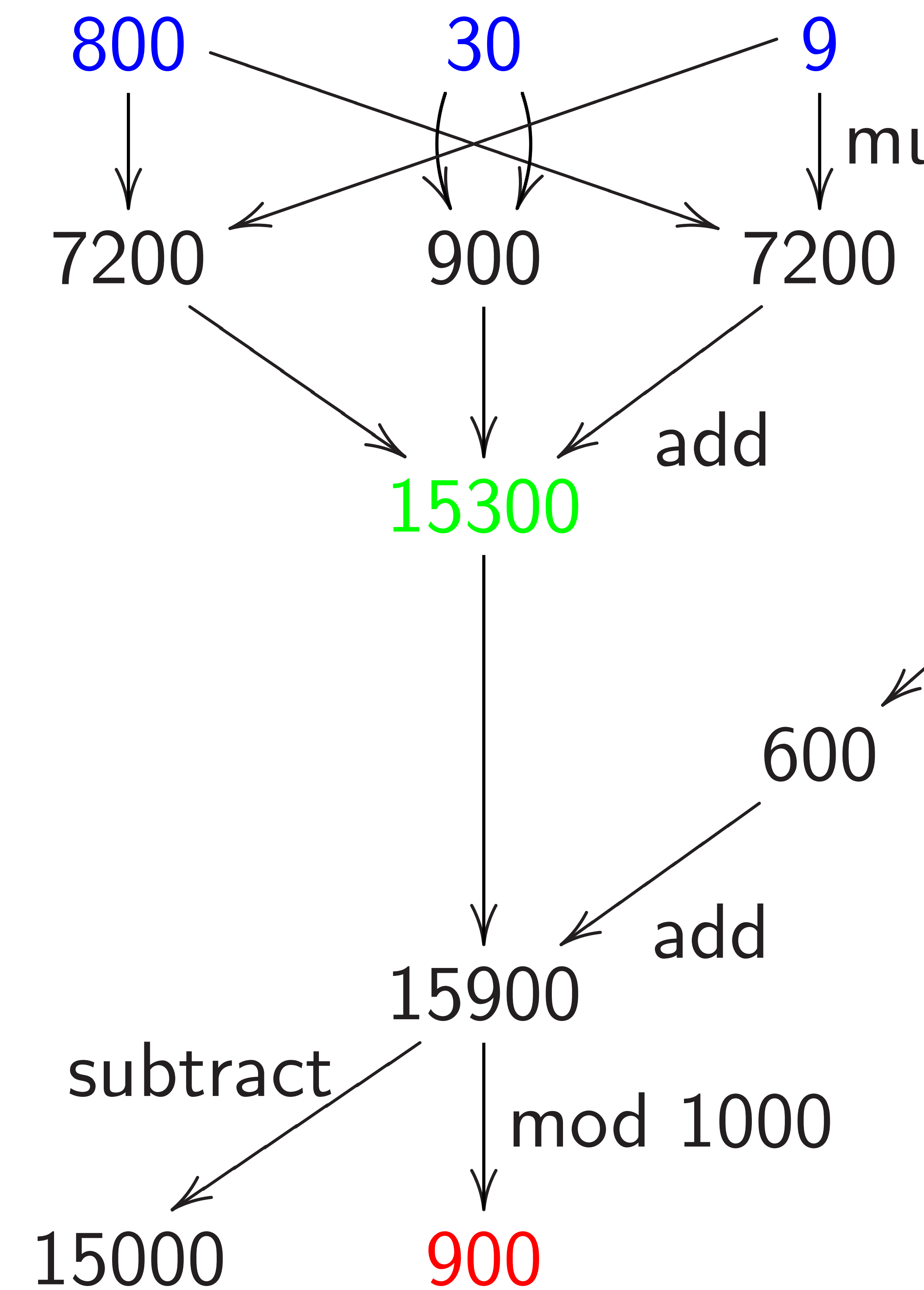
The scaled variation

$839 = 800 + 30 + 9 =$
value (at $t = 1$) of polynomial
$800t^2 + 30t^1 + 9t^0$.

Squaring: $(800t^2 + 30t^1 + 9t^0)^2 =$
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$.

Carrying:
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$;
$640000t^4 + 48000t^3 + 15300t^2 +$
$620t^1 + 1t^0$; $\quad \ldots$
$700000t^5 + 0t^4 + 3000t^3 + 900t^2 +$
$20t^1 + 1t^0$.

What operations were used



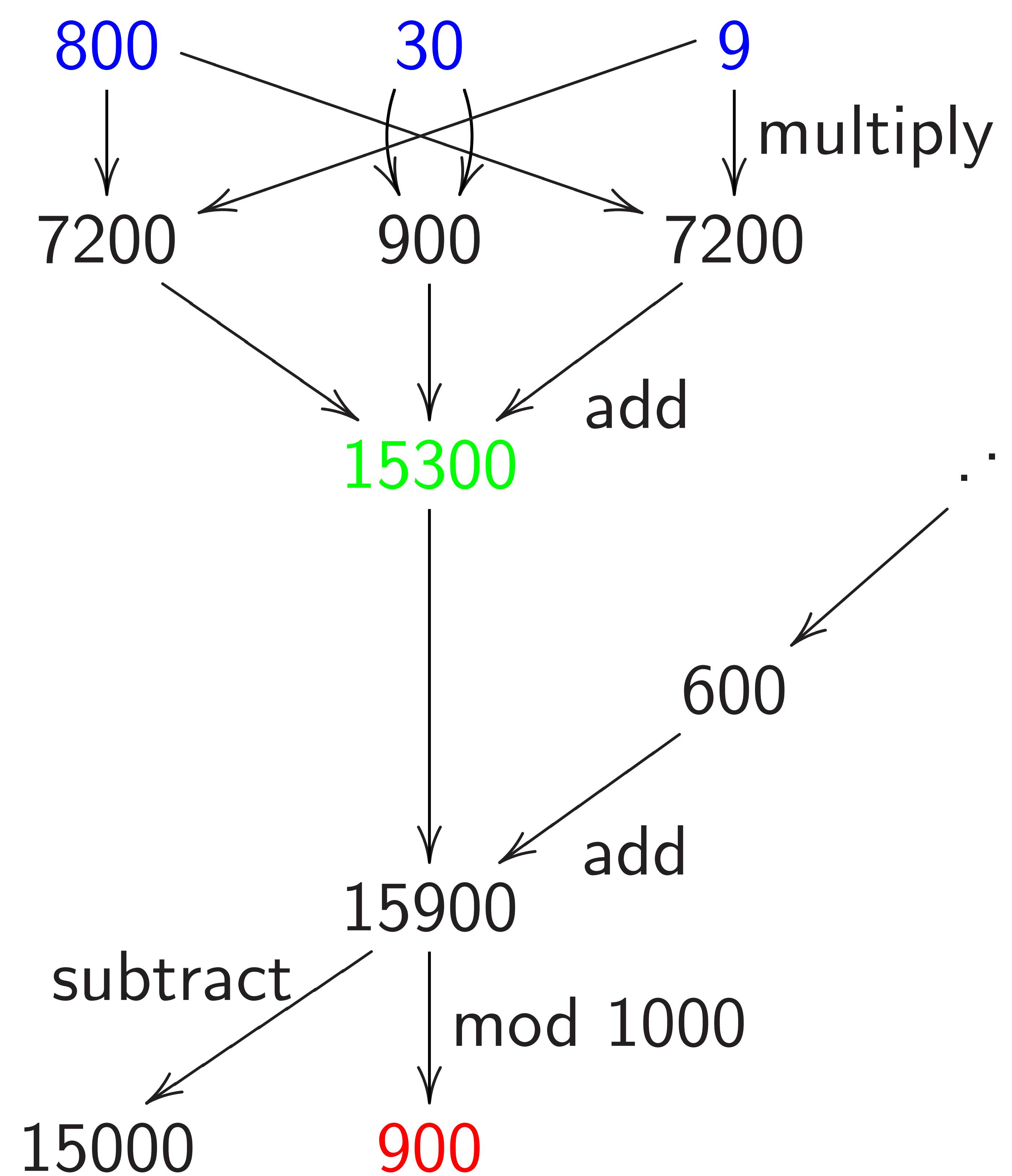
The scaled variation

$839 = 800 + 30 + 9 =$
value (at $t = 1$) of polynomial
$800t^2 + 30t^1 + 9t^0$.

Squaring: $(800t^2 + 30t^1 + 9t^0)^2 =$
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$.

Carrying:
$640000t^4 + 48000t^3 + 15300t^2 +$
$540t^1 + 81t^0$;
$640000t^4 + 48000t^3 + 15300t^2 +$
$620t^1 + 1t^0$; $\quad \ldots$
$700000t^5 + 0t^4 + 3000t^3 + 900t^2 +$
$20t^1 + 1t^0$.

What operations were used here?



led variation

$00 + 30 + 9 =$

t $t = 1$) of polynomial

$30t^1 + 9t^0$.

g: $(800t^2 + 30t^1 + 9t^0)^2 =$

$^4 + 48000t^3 + 15300t^2 +$

$81t^0$.

g:

$^4 + 48000t^3 + 15300t^2 +$

$81t^0$;

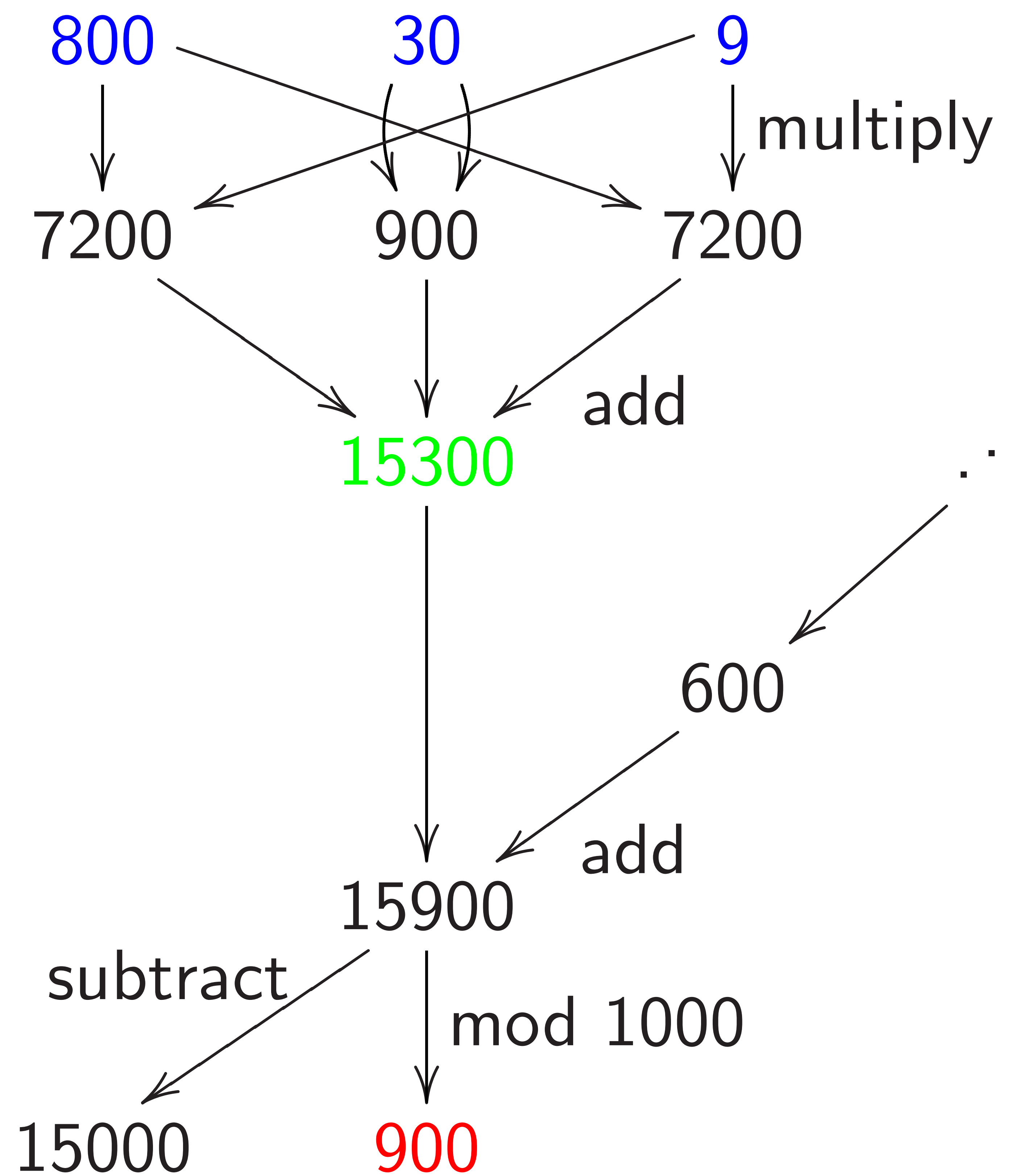
$^4 + 48000t^3 + 15300t^2 +$

$1t^0$; $\ldots$

$^5 + 0t^4 + 3000t^3 + 900t^2 +$

t^0.

What operations were used here?



Speedup

$(\cdots + f_2$

has coef

$f_4 f_0 + f$

5 mults,

on

$9 =$

f polynomial

^{0}.

$+30t^1+9t^0)^2 =$

$t^3 + 15300t^2 +$

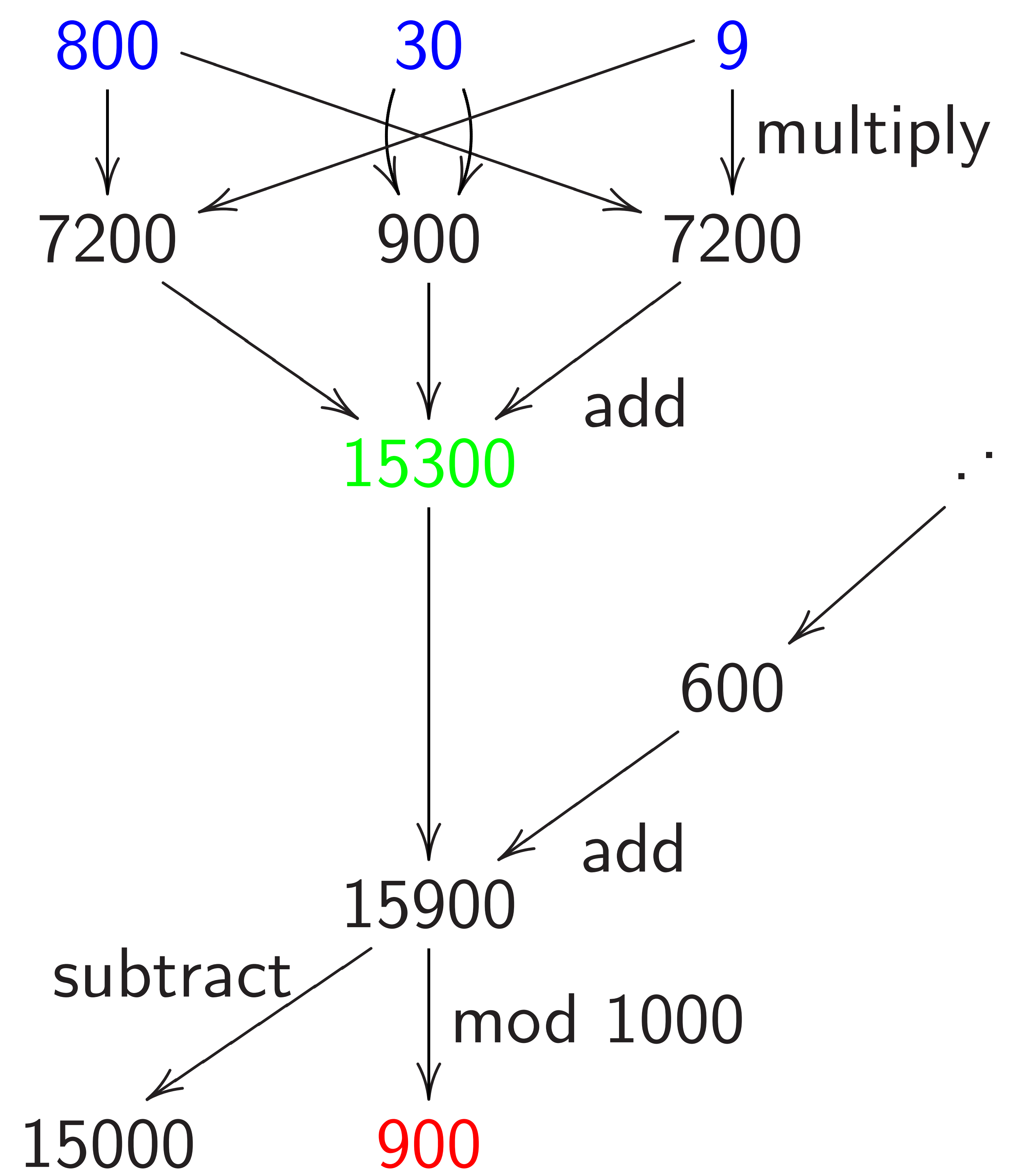
$t^3 + 15300t^2 +$

$t^3 + 15300t^2 +$

. .

$3000t^3 + 900t^2 +$

What operations were used here?



Speedup: double i

$(\cdots + f_2t^2 + f_1t^1$

has coefficients su

$f_4f_0 + f_3f_1 + f_2f$

5 mults, 4 adds.

ial

$^0)^2 =$

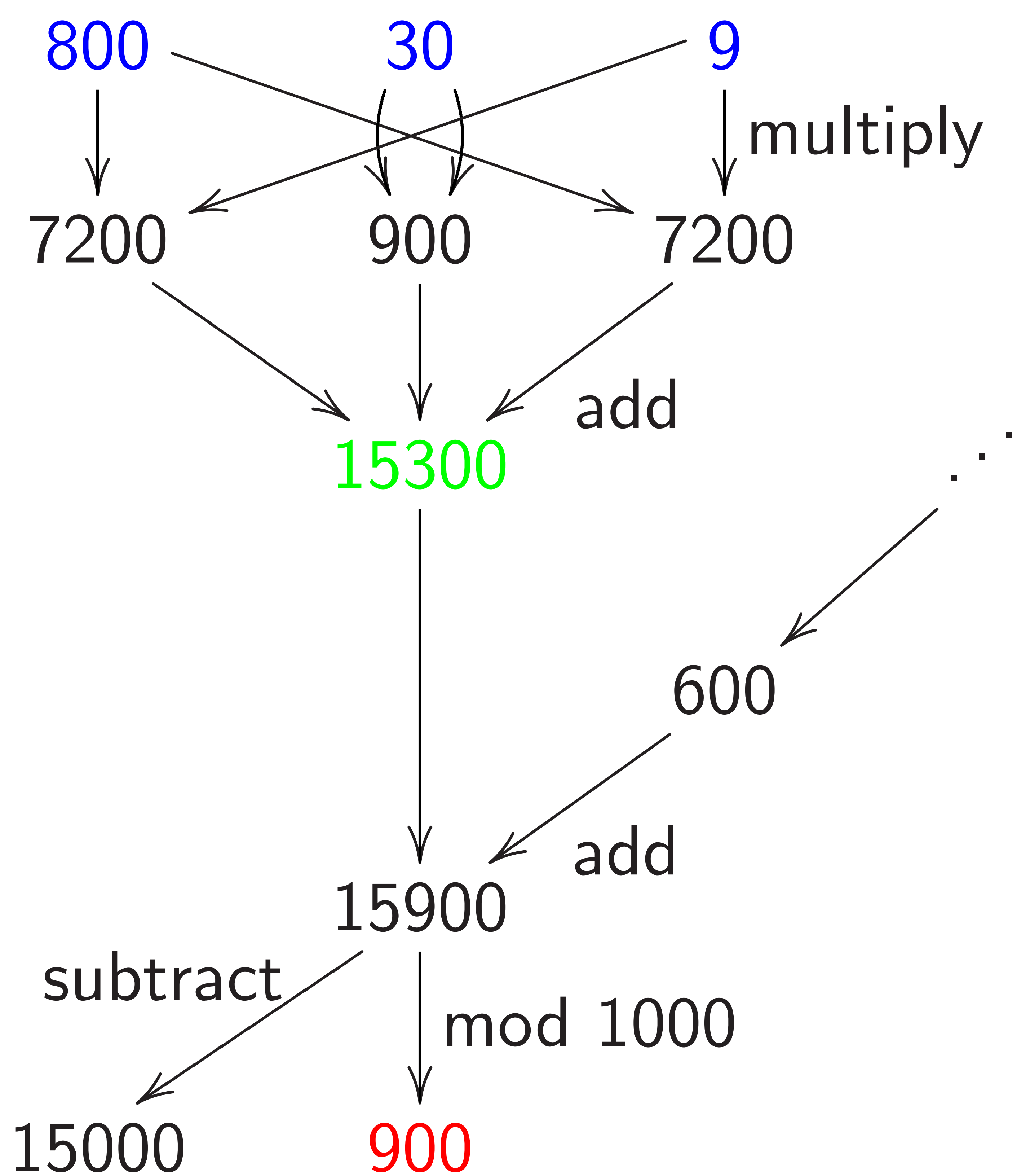
$0t^2 +$

$0t^2 +$

$0t^2 +$

$00t^2 +$

What operations were used here?



<u>Speedup: double inside squa</u>

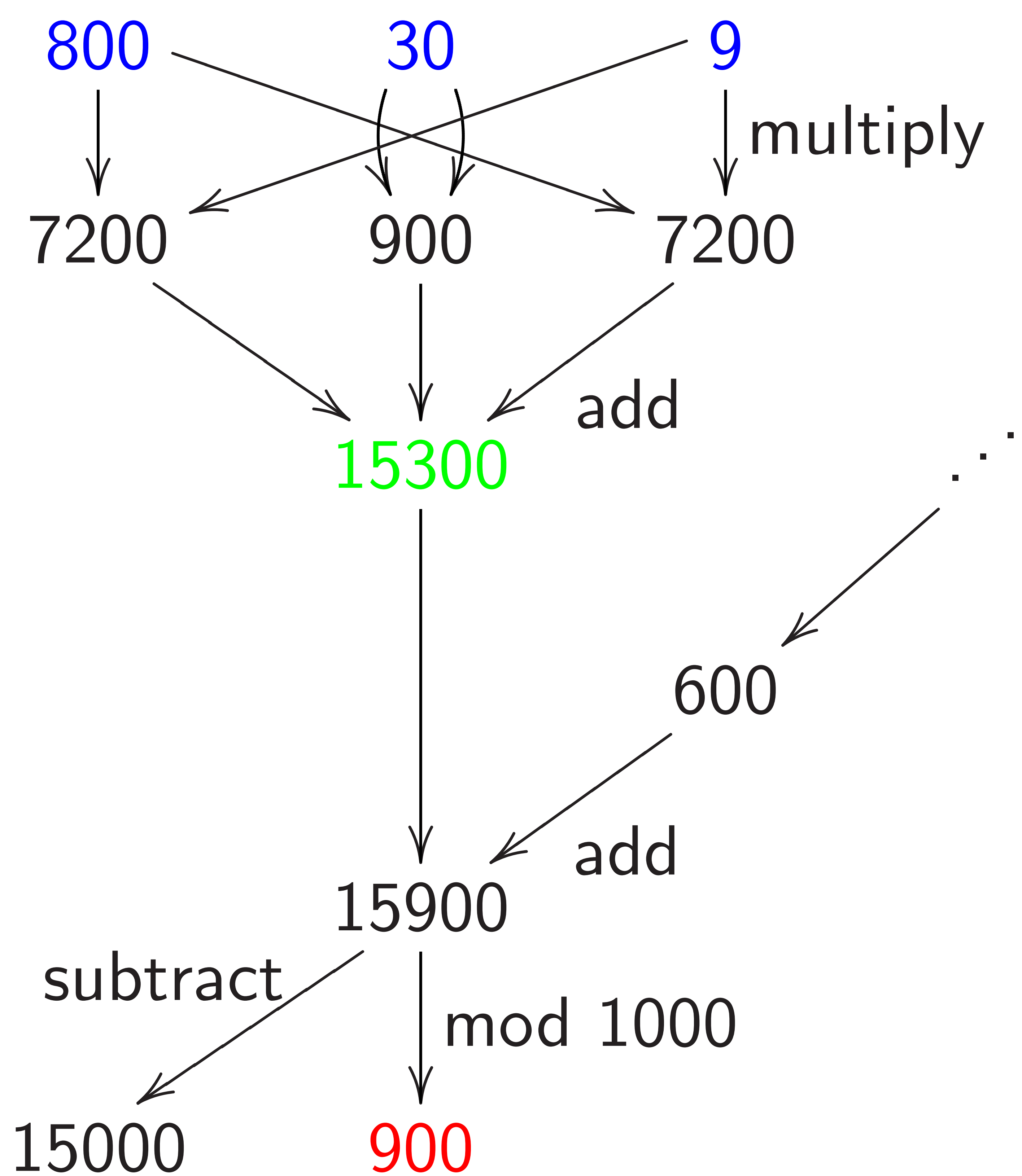
$(\cdots + f_2 t^2 + f_1 t^1 + f_0 t^0)^2$

has coefficients such as

$f_4 f_0 + f_3 f_1 + f_2 f_2 + f_1 f_3 +$

5 mults, 4 adds.

What operations were used here?



<u>Speedup: double inside squaring</u>

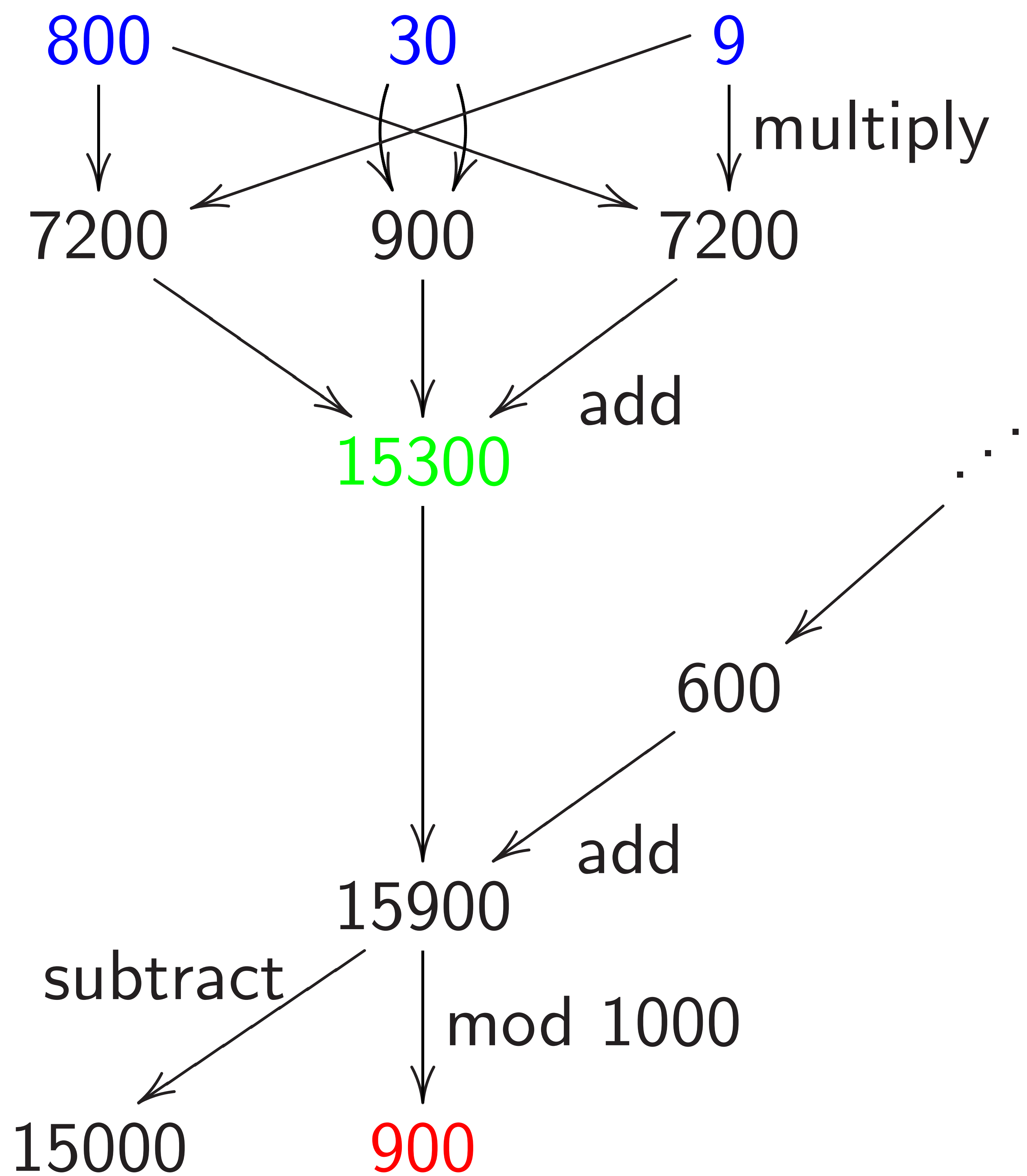
$(\cdots + f_2t^2 + f_1t^1 + f_0t^0)^2$

has coefficients such as

$f_4f_0 + f_3f_1 + f_2f_2 + f_1f_3 + f_0f_4$.

5 mults, 4 adds.

What operations were used here?



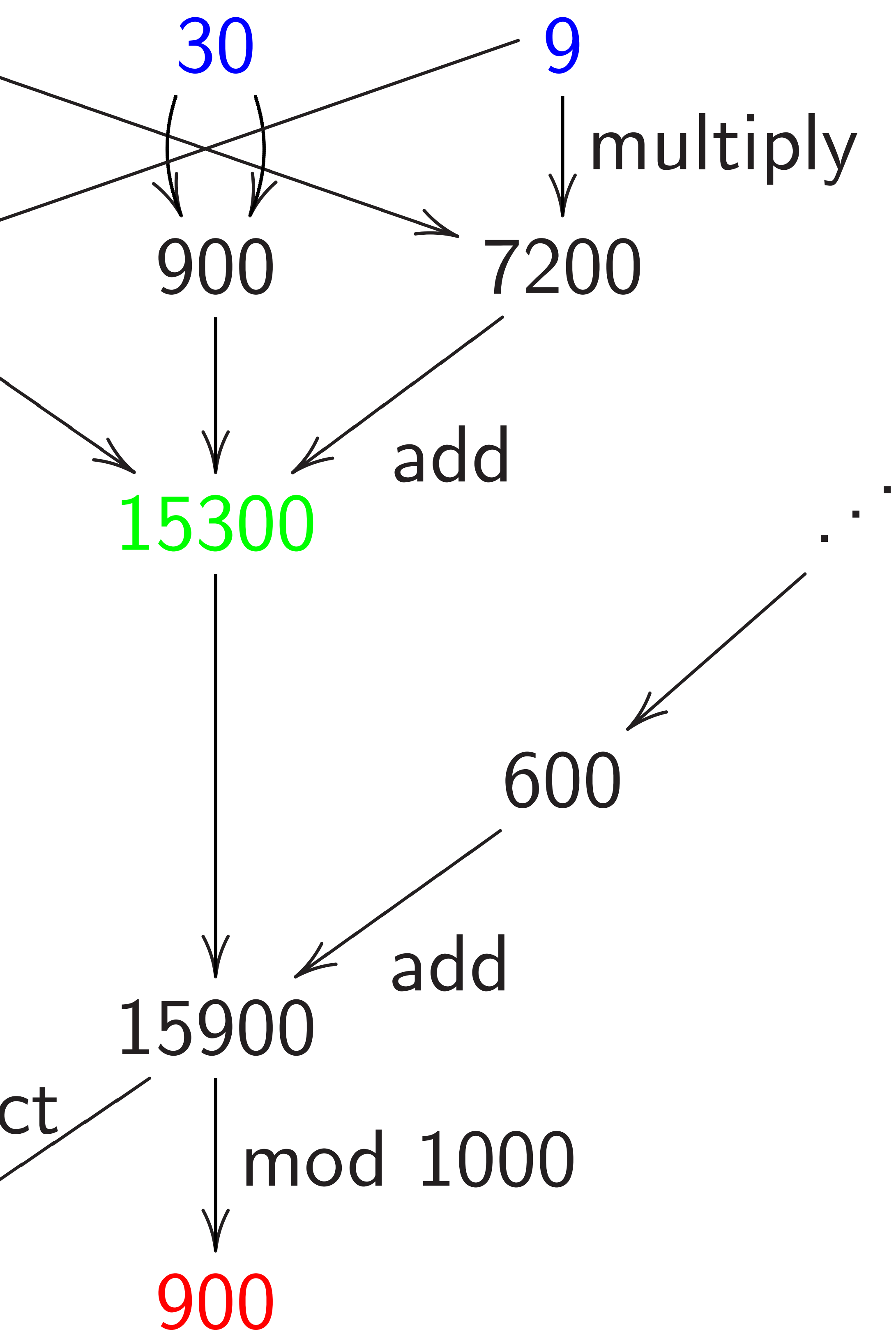
Speedup: double inside squaring

$(\cdots + f_2t^2 + f_1t^1 + f_0t^0)^2$
has coefficients such as
$f_4f_0 + f_3f_1 + f_2f_2 + f_1f_3 + f_0f_4$.
5 mults, 4 adds.

Compute more efficiently as
$2f_4f_0 + 2f_3f_1 + f_2f_2$.
3 mults, 2 adds, 2 doublings.

Save $\approx 1/2$ of the mults
if there are many coefficients.

perations were used here?



Speedup: double inside squaring

$(\cdots + f_2 t^2 + f_1 t^1 + f_0 t^0)^2$
has coefficients such as
$f_4 f_0 + f_3 f_1 + f_2 f_2 + f_1 f_3 + f_0 f_4$.
5 mults, 4 adds.

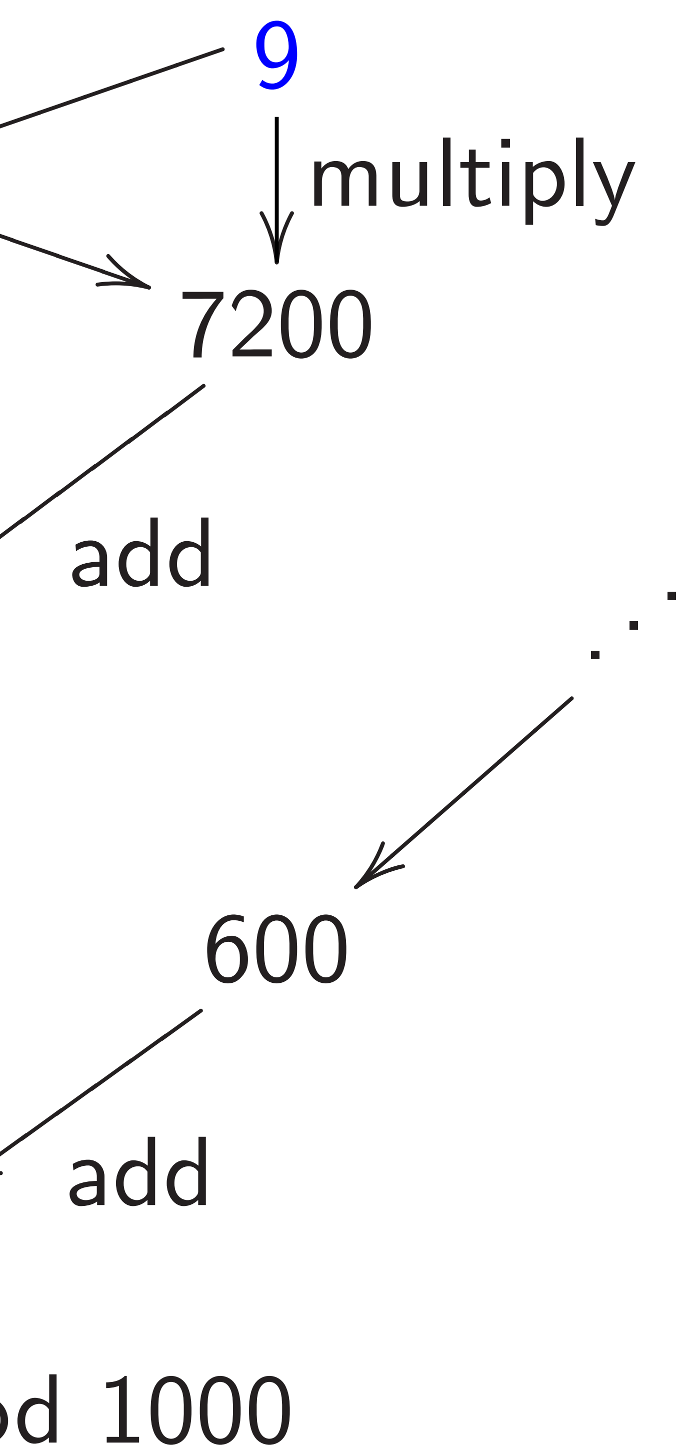
Compute more efficiently as
$2 f_4 f_0 + 2 f_3 f_1 + f_2 f_2$.
3 mults, 2 adds, 2 doublings.

Save $\approx 1/2$ of the mults
if there are many coefficients.

Faster a
$2(f_4 f_0 +$
3 mults,

Save $\approx$
if there a

were used here?



<u>Speedup: double inside squaring</u>

$(\cdots + f_2 t^2 + f_1 t^1 + f_0 t^0)^2$
has coefficients such as
$f_4 f_0 + f_3 f_1 + f_2 f_2 + f_1 f_3 + f_0 f_4$.
5 mults, 4 adds.

Compute more efficiently as
$2 f_4 f_0 + 2 f_3 f_1 + f_2 f_2$.
3 mults, 2 adds, 2 doublings.

Save $\approx 1/2$ of the mults
if there are many coefficients.

Faster alternative:
$2(f_4 f_0 + f_3 f_1) +$
3 mults, 2 adds, 1

Save $\approx 1/2$ of the
if there are many

here?

ultiply

Speedup: double inside squaring

$(\cdots + f_2t^2 + f_1t^1 + f_0t^0)^2$
has coefficients such as
$f_4f_0 + f_3f_1 + f_2f_2 + f_1f_3 + f_0f_4$.
5 mults, 4 adds.

Compute more efficiently as
$2f_4f_0 + 2f_3f_1 + f_2f_2$.
3 mults, 2 adds, 2 doublings.

Save $\approx 1/2$ of the mults
if there are many coefficients.

Faster alternative:
$2(f_4f_0 + f_3f_1) + f_2f_2$.
3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds
if there are many coefficient

Speedup: double inside squaring

$(\cdots + f_2 t^2 + f_1 t^1 + f_0 t^0)^2$
has coefficients such as
$f_4 f_0 + f_3 f_1 + f_2 f_2 + f_1 f_3 + f_0 f_4$.
5 mults, 4 adds.

Compute more efficiently as
$2 f_4 f_0 + 2 f_3 f_1 + f_2 f_2$.
3 mults, 2 adds, 2 doublings.

Save $\approx 1/2$ of the mults
if there are many coefficients.

Faster alternative:
$2(f_4 f_0 + f_3 f_1) + f_2 f_2$.
3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds
if there are many coefficients.

Speedup: double inside squaring

$(\cdots + f_2t^2 + f_1t^1 + f_0t^0)^2$
has coefficients such as
$f_4f_0 + f_3f_1 + f_2f_2 + f_1f_3 + f_0f_4$.
5 mults, 4 adds.

Compute more efficiently as
$2f_4f_0 + 2f_3f_1 + f_2f_2$.
3 mults, 2 adds, 2 doublings.

Save $\approx 1/2$ of the mults
if there are many coefficients.

Faster alternative:
$2(f_4f_0 + f_3f_1) + f_2f_2$.
3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds
if there are many coefficients.

Even faster alternative:
$(2f_0)f_4 + (2f_1)f_3 + f_2f_2$,
after precomputing $2f_0, 2f_1, \ldots$.

3 mults, 2 adds, 0 doublings.
Precomputation ≈ 0.5 doublings.

: double inside squaring

$_2t^2 + f_1t^1 + f_0t^0)^2$

fficients such as

$f_3f_1 + f_2f_2 + f_1f_3 + f_0f_4$.

4 adds.

e more efficiently as

$2f_3f_1 + f_2f_2$.

2 adds, 2 doublings.

1/2 of the mults

are many coefficients.

Faster alternative:

$2(f_4f_0 + f_3f_1) + f_2f_2$.

3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds

if there are many coefficients.

Even faster alternative:

$(2f_0)f_4 + (2f_1)f_3 + f_2f_2$,

after precomputing $2f_0, 2f_1, \ldots$.

3 mults, 2 adds, 0 doublings.

Precomputation ≈ 0.5 doublings.

Speedup

Recall 1

Scaled:

Alternat

Scaled:

Use digi

instead o

Small di

Several s

easily ha

easily ha

reduce p

nside squaring

$+ f_0 t^0)^2$

ch as

$_2 + f_1 f_3 + f_0 f_4$.

iciently as

$_2 f_2$.

doublings.

mults

coefficients.

Faster alternative:

$2(f_4 f_0 + f_3 f_1) + f_2 f_2$.

3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds
if there are many coefficients.

Even faster alternative:

$(2f_0)f_4 + (2f_1)f_3 + f_2 f_2$,

after precomputing $2f_0, 2f_1, \ldots$.

3 mults, 2 adds, 0 doublings.

Precomputation ≈ 0.5 doublings.

Speedup: allow ne

Recall $159 \mapsto 15$, 9

Scaled: $15900 \mapsto$

Alternative: $159 \mapsto$

Scaled: $15900 \mapsto$

Use digits $\{-5, -$
instead of $\{0, 1, \ldots$
Small disadvantag
Several small adva
easily handle nega
easily handle subtr
reduce products a

aring

$- f_0 f_4$.

s.

s.

Faster alternative:

$2(f_4 f_0 + f_3 f_1) + f_2 f_2$.

3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds if there are many coefficients.

Even faster alternative:

$(2f_0)f_4 + (2f_1)f_3 + f_2 f_2$,

after precomputing $2f_0, 2f_1, \ldots$.

3 mults, 2 adds, 0 doublings.

Precomputation ≈ 0.5 doublings.

Speedup: allow negative coe

Recall $159 \mapsto 15, 9$.

Scaled: $15900 \mapsto 15000, 900$

Alternative: $159 \mapsto 16, -1$.

Scaled: $15900 \mapsto 16000, -1$

Use digits $\{-5, -4, \ldots, 4, 5$

instead of $\{0, 1, \ldots, 9\}$.

Small disadvantage: need $-$

Several small advantages:

easily handle negative intege

easily handle subtraction;

reduce products a bit.

Faster alternative:

$2(f_4f_0 + f_3f_1) + f_2f_2$.

3 mults, 2 adds, 1 doubling.

Save $\approx 1/2$ of the adds if there are many coefficients.

Even faster alternative:

$(2f_0)f_4 + (2f_1)f_3 + f_2f_2$,

after precomputing $2f_0, 2f_1, \ldots$.

3 mults, 2 adds, 0 doublings.

Precomputation ≈ 0.5 doublings.

Speedup: allow negative coeffs

Recall $159 \mapsto 15, 9$.

Scaled: $15900 \mapsto 15000, 900$.

Alternative: $159 \mapsto 16, -1$.

Scaled: $15900 \mapsto 16000, -100$.

Use digits $\{-5, -4, \ldots, 4, 5\}$ instead of $\{0, 1, \ldots, 9\}$.

Small disadvantage: need $-$.

Several small advantages:

easily handle negative integers;

easily handle subtraction;

reduce products a bit.

lternative:

$- f_3 f_1) + f_2 f_2$.

2 adds, 1 doubling.

1/2 of the adds

are many coefficients.

ster alternative:

$+ (2f_1)f_3 + f_2 f_2$,

ecomputing $2f_0, 2f_1, \ldots$.

2 adds, 0 doublings.

putation $\approx$ 0.5 doublings.

Speedup: allow negative coeffs

Recall 159 $\mapsto$ 15, 9.

Scaled: 15900 $\mapsto$ 15000, 900.

Alternative: 159 $\mapsto$ 16, -1.

Scaled: 15900 $\mapsto$ 16000, -100.

Use digits $\{-5, -4, \ldots, 4, 5\}$

instead of $\{0, 1, \ldots, 9\}$.

Small disadvantage: need $-$.

Several small advantages:

easily handle negative integers;

easily handle subtraction;

reduce products a bit.

Speedup

Computi

multiply

square c

e.g. $a =$

$(3t^2 + 1t$

$6t^4 + 23$

carry: 8

As befor

$64t^4 + 4$

$7t^5 + 0t$

$+$: $7t^5 +$

$7t^5 + 8t$

$f_2 f_2$.
doubling.

adds
coefficients.

ative:
$+ f_2 f_2$,
g $2f_0, 2f_1, \ldots$.

doublings.
$\approx$ 0.5 doublings.

Speedup: allow negative coeffs

Recall $159 \mapsto 15, 9$.
Scaled: $15900 \mapsto 15000, 900$.

Alternative: $159 \mapsto 16, -1$.
Scaled: $15900 \mapsto 16000, -100$.

Use digits $\{-5, -4, \ldots, 4, 5\}$
instead of $\{0, 1, \ldots, 9\}$.
Small disadvantage: need $-$.
Several small advantages:
easily handle negative integers;
easily handle subtraction;
reduce products a bit.

Speedup: delay ca

Computing (e.g.)
multiply a, b polyn
square c poly, carr

e.g. $a = 314$, $b =$
$(3t^2 + 1t^1 + 4t^0)(2t$
$6t^4 + 23t^3 + 18t^2$
carry: $8t^4 + 5t^3 +$

As before $(8t^2 + 3$
$64t^4 + 48t^3 + 153t$
$7t^5 + 0t^4 + 3t^3 +$

$+$: $7t^5 + 8t^4 + 8t^3 +$
$7t^5 + 8t^4 + 9t^3 +$

s.

, ...

s.

lings.

Speedup: allow negative coeffs

Recall $159 \mapsto 15, 9$.

Scaled: $15900 \mapsto 15000, 900$.

Alternative: $159 \mapsto 16, -1$.

Scaled: $15900 \mapsto 16000, -100$.

Use digits $\{-5, -4, \ldots, 4, 5\}$
instead of $\{0, 1, \ldots, 9\}$.
Small disadvantage: need $-$.
Several small advantages:
easily handle negative integers;
easily handle subtraction;
reduce products a bit.

Speedup: delay carries

Computing (e.g.) big $ab + c$
multiply a, b polynomials, ca
square c poly, carry, add, ca

e.g. $a = 314$, $b = 271$, $c =$
$(3t^2 + 1t^1 + 4t^0)(2t^2 + 7t^1 + 1$
$6t^4 + 23t^3 + 18t^2 + 29t^1 +$
carry: $8t^4 + 5t^3 + 0t^2 + 9t^1$

As before $(8t^2 + 3t^1 + 9t^0)^2$
$64t^4 + 48t^3 + 153t^2 + 54t^1 +$
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1$

$+$: $7t^5 + 8t^4 + 8t^3 + 9t^2 + 11t$
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1$

Speedup: allow negative coeffs

Recall $159 \mapsto 15, 9$.

Scaled: $15900 \mapsto 15000, 900$.

Alternative: $159 \mapsto 16, -1$.

Scaled: $15900 \mapsto 16000, -100$.

Use digits $\{-5, -4, \ldots, 4, 5\}$ instead of $\{0, 1, \ldots, 9\}$.

Small disadvantage: need $-$.

Several small advantages: easily handle negative integers; easily handle subtraction; reduce products a bit.

Speedup: delay carries

Computing (e.g.) big $ab + c^2$: multiply a, b polynomials, carry, square c poly, carry, add, carry.

e.g. $a = 314$, $b = 271$, $c = 839$:

$(3t^2+1t^1+4t^0)(2t^2+7t^1+1t^0) =$

$6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0$;

carry: $8t^4 + 5t^3 + 0t^2 + 9t^1 + 4t^0$.

As before $(8t^2 + 3t^1 + 9t^0)^2 =$

$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;

$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

$+$: $7t^5+8t^4+8t^3+9t^2+11t^1+5t^0$;

$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

p: allow negative coeffs

59 $\mapsto$ 15, 9.

15900 $\mapsto$ 15000, 900.

ive: 159 $\mapsto$ 16, -1.

15900 $\mapsto$ 16000, -100.

ts $\{-5, -4, \ldots, 4, 5\}$

of $\{0, 1, \ldots, 9\}$.

sadvantage: need $-$.

small advantages:

ndle negative integers;

ndle subtraction;

roducts a bit.

Speedup: delay carries

Computing (e.g.) big $ab + c^2$:
multiply a, b polynomials, carry,
square c poly, carry, add, carry.

e.g. $a = 314$, $b = 271$, $c = 839$:
$(3t^2 + 1t^1 + 4t^0)(2t^2 + 7t^1 + 1t^0) =$
$6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0$;
carry: $8t^4 + 5t^3 + 0t^2 + 9t^1 + 4t^0$.

As before $(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

$+$: $7t^5 + 8t^4 + 8t^3 + 9t^2 + 11t^1 + 5t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Faster: r

square c

$(6t^4 + 2$

$(64t^4 + 4$

$= 70t^4 +$

$7t^5 + 8t$

Eliminat

Outweig

slightly l

Importar

multiplic

to reduc

but carri

before a

egative coeffs

9.

15000, 900.

→ 16, −1.

16000, −100.

4, . . . , 4, 5}

. . , 9}.

e: need −.

ntages:

tive integers;

raction;

bit.

Speedup: delay carries

Computing (e.g.) big $ab + c^2$: multiply a, b polynomials, carry, square c poly, carry, add, carry.

e.g. $a = 314$, $b = 271$, $c = 839$:
$(3t^2 + 1t^1 + 4t^0)(2t^2 + 7t^1 + 1t^0) =$
$6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0$;
carry: $8t^4 + 5t^3 + 0t^2 + 9t^1 + 4t^0$.

As before $(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

$+$: $7t^5 + 8t^4 + 8t^3 + 9t^2 + 11t^1 + 5t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Faster: multiply a
square c polynomi

$(6t^4 + 23t^3 + 18t^2$
$(64t^4 + 48t^3 + 153$
$= 70t^4 + 71t^3 + 17$
$7t^5 + 8t^4 + 9t^3 +$

Eliminate intermed
Outweighs cost of
slightly larger coef

Important to carry
multiplications (ar
to reduce coefficie
but carries are usu
before additions, s

effs

0.

00.

}

.

ers;

Speedup: delay carries

Computing (e.g.) big $ab + c^2$:
multiply a, b polynomials, carry,
square c poly, carry, add, carry.

e.g. $a = 314$, $b = 271$, $c = 839$:
$(3t^2+1t^1+4t^0)(2t^2+7t^1+1t^0) =$
$6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0$;
carry: $8t^4 + 5t^3 + 0t^2 + 9t^1 + 4t^0$.

As before $(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

$+$: $7t^5+8t^4+8t^3+9t^2+11t^1+5t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Faster: multiply a, b polynor
square c polynomial, add, ca

$(6t^4 + 23t^3 + 18t^2 + 29t^1 +$
$(64t^4 + 48t^3 + 153t^2 + 54t^1 +$
$= 70t^4 + 71t^3 + 171t^2 + 83t^1 +$
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1$

Eliminate intermediate carrie
Outweighs cost of handling
slightly larger coefficients.

Important to carry between
multiplications (and squarin
to reduce coefficient size;
but carries are usually a bad
before additions, subtractior

Speedup: delay carries

Computing (e.g.) big $ab + c^2$: multiply a, b polynomials, carry, square c poly, carry, add, carry.

e.g. $a = 314$, $b = 271$, $c = 839$:
$(3t^2+1t^1+4t^0)(2t^2+7t^1+1t^0) =$
$6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0$;
carry: $8t^4 + 5t^3 + 0t^2 + 9t^1 + 4t^0$.

As before $(8t^2 + 3t^1 + 9t^0)^2 =$
$64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0$;
$7t^5 + 0t^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

$+$: $7t^5+8t^4+8t^3+9t^2+11t^1+5t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Faster: multiply a, b polynomials, square c polynomial, add, carry.

$(6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0) +$
$(64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0)$
$= 70t^4+71t^3+171t^2+83t^1+85t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Eliminate intermediate carries. Outweighs cost of handling slightly larger coefficients.

Important to carry between multiplications (and squarings) to reduce coefficient size; but carries are usually a bad idea before additions, subtractions, etc.

p: delay carries

ing (e.g.) big $ab + c^2$:
a, b polynomials, carry,
poly, carry, add, carry.

$= 314$, $b = 271$, $c = 839$:
$t^1 + 4t^0)(2t^2 + 7t^1 + 1t^0) =$
$t^3 + 18t^2 + 29t^1 + 4t^0$;
$t^4 + 5t^3 + 0t^2 + 9t^1 + 4t^0$.

re $(8t^2 + 3t^1 + 9t^0)^2 =$
$8t^3 + 153t^2 + 54t^1 + 81t^0$;
$^4 + 3t^3 + 9t^2 + 2t^1 + 1t^0$.

$8t^4 + 8t^3 + 9t^2 + 11t^1 + 5t^0$;
$^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Faster: multiply a, b polynomials,
square c polynomial, add, carry.

$(6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0) +$
$(64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0)$
$= 70t^4 + 71t^3 + 171t^2 + 83t^1 + 85t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Eliminate intermediate carries.
Outweighs cost of handling
slightly larger coefficients.

Important to carry between
multiplications (and squarings)
to reduce coefficient size;
but carries are usually a bad idea
before additions, subtractions, etc.

Speedup

How mu

$f = f_0$ -

$g = g_0$ +

Using th
400 coef

Faster:

$F_0 = f_0$

$F_1 = f_{10}$

Similarly

Then fg

$+ (F_0G_0$

rries

big $ab + c^2$:

nomials, carry,

y, add, carry.

271, $c = 839$:

$t^2 + 7t^1 + 1t^0) =$

$+ 29t^1 + 4t^0$;

$0t^2 + 9t^1 + 4t^0$.

$3t^1 + 9t^0)^2 =$

$t^2 + 54t^1 + 81t^0$;

$9t^2 + 2t^1 + 1t^0$.

$+9t^2 + 11t^1 + 5t^0$;

$0t^2 + 1t^1 + 5t^0$.

Faster: multiply a, b polynomials, square c polynomial, add, carry.

$(6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0) +$
$(64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0)$
$= 70t^4 + 71t^3 + 171t^2 + 83t^1 + 85t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Eliminate intermediate carries. Outweighs cost of handling slightly larger coefficients.

Important to carry between multiplications (and squarings) to reduce coefficient size; but carries are usually a bad idea before additions, subtractions, etc.

Speedup: polynom

How much work t

$f = f_0 + f_1 t + \cdot\cdot$

$g = g_0 + g_1 t + \cdots$

Using the obvious

400 coeff mults, 3

Faster: Write f as

$F_0 = f_0 + f_1 t + \cdot$

$F_1 = f_{10} + f_{11} t +$

Similarly write g a

Then $fg = (F_0 +$

$+ (F_0 G_0 - F_1 G_1 t^1$

^{2}:

rry,

rry.

839:

$t^0) =$

$4t^0$;

$+ 4t^0$.

$=$

$+ 81t^0$;

$+ 1t^0$.

$^1 + 5t^0$;

$+ 5t^0$.

Faster: multiply a, b polynomials, square c polynomial, add, carry.

$(6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0) +$
$(64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0)$
$= 70t^4 + 71t^3 + 171t^2 + 83t^1 + 85t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Eliminate intermediate carries. Outweighs cost of handling slightly larger coefficients.

Important to carry between multiplications (and squarings) to reduce coefficient size; but carries are usually a bad idea before additions, subtractions, etc.

Speedup: polynomial Karats

How much work to multiply
$f = f_0 + f_1 t + \cdots + f_{19} t^{19}$,
$g = g_0 + g_1 t + \cdots + g_{19} t^{19}$?

Using the obvious method:
400 coeff mults, 361 coeff a

Faster: Write f as $F_0 + F_1 t$
$F_0 = f_0 + f_1 t + \cdots + f_9 t^9$;
$F_1 = f_{10} + f_{11} t + \cdots + f_{19} t$
Similarly write g as $G_0 + G_1$

Then $fg = (F_0 + F_1)(G_0 +$
$+ (F_0 G_0 - F_1 G_1 t^{10})(1 - t^{10}$

Faster: multiply a, b polynomials, square c polynomial, add, carry.

$(6t^4 + 23t^3 + 18t^2 + 29t^1 + 4t^0) +$
$(64t^4 + 48t^3 + 153t^2 + 54t^1 + 81t^0)$
$= 70t^4 + 71t^3 + 171t^2 + 83t^1 + 85t^0$;
$7t^5 + 8t^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

Eliminate intermediate carries. Outweighs cost of handling slightly larger coefficients.

Important to carry between multiplications (and squarings) to reduce coefficient size;
but carries are usually a bad idea before additions, subtractions, etc.

Speedup: polynomial Karatsuba

How much work to multiply polys
$f = f_0 + f_1 t + \cdots + f_{19} t^{19}$,
$g = g_0 + g_1 t + \cdots + g_{19} t^{19}$?

Using the obvious method:
400 coeff mults, 361 coeff adds.

Faster: Write f as $F_0 + F_1 t^{10}$;
$F_0 = f_0 + f_1 t + \cdots + f_9 t^9$;
$F_1 = f_{10} + f_{11} t + \cdots + f_{19} t^9$.
Similarly write g as $G_0 + G_1 t^{10}$.

Then $fg = (F_0 + F_1)(G_0 + G_1) t^{10}$
$+ (F_0 G_0 - F_1 G_1 t^{10})(1 - t^{10})$.

multiply a, b polynomials,
polynomial, add, carry.

$3t^3 + 18t^2 + 29t^1 + 4t^0) +$
$48t^3 + 153t^2 + 54t^1 + 81t^0)$
$71t^3 + 171t^2 + 83t^1 + 85t^0$;
$^4 + 9t^3 + 0t^2 + 1t^1 + 5t^0$.

e intermediate carries.
hs cost of handling
arger coefficients.

nt to carry between
cations (and squarings)
e coefficient size;
ies are usually a bad idea
dditions, subtractions, etc.

Speedup: polynomial Karatsuba

How much work to multiply polys
$f = f_0 + f_1 t + \cdots + f_{19} t^{19}$,
$g = g_0 + g_1 t + \cdots + g_{19} t^{19}$?

Using the obvious method:
400 coeff mults, 361 coeff adds.

Faster: Write f as $F_0 + F_1 t^{10}$;
$F_0 = f_0 + f_1 t + \cdots + f_9 t^9$;
$F_1 = f_{10} + f_{11} t + \cdots + f_{19} t^9$.
Similarly write g as $G_0 + G_1 t^{10}$.

Then $fg = (F_0 + F_1)(G_0 + G_1) t^{10}$
$+ (F_0 G_0 - F_1 G_1 t^{10})(1 - t^{10})$.

20 adds
300 mul
$F_0 G_0$, F
243 add
9 adds f
with sub
and with
19 adds
19 adds

Total 30
Larger c
still save

Can app
as poly

, b polynomials,
al, add, carry.

$+ 29t^1 + 4t^0) +$
$t^2 + 54t^1 + 81t^0)$
$1t^2 + 83t^1 + 85t^0;$
$0t^2 + 1t^1 + 5t^0.$

diate carries.
handling
fficients.

between
nd squarings)
nt size;
ally a bad idea
ubtractions, etc.

Speedup: polynomial Karatsuba

How much work to multiply polys
$f = f_0 + f_1 t + \cdots + f_{19} t^{19}$,
$g = g_0 + g_1 t + \cdots + g_{19} t^{19}$?

Using the obvious method:
400 coeff mults, 361 coeff adds.

Faster: Write f as $F_0 + F_1 t^{10}$;
$F_0 = f_0 + f_1 t + \cdots + f_9 t^9$;
$F_1 = f_{10} + f_{11} t + \cdots + f_{19} t^9$.
Similarly write g as $G_0 + G_1 t^{10}$.

Then $fg = (F_0 + F_1)(G_0 + G_1)t^{10}$
$+ (F_0 G_0 - F_1 G_1 t^{10})(1 - t^{10})$.

20 adds for $F_0 +$
300 mults for thre
$F_0 G_0$, $F_1 G_1$, $(F_0 +$
243 adds for those
9 adds for $F_0 G_0 -$
with subs counted
and with delayed
19 adds for $\cdots(1$
19 adds to finish.

Total 300 mults, 3
Larger coefficients
still saves time.

Can apply idea rec
as poly degree gro

mials,

arry.

$4t^0) +$

$+81t^0)$

$+85t^0;$

$+ 5t^0.$

es.

gs)

idea

ns, etc.

Speedup: polynomial Karatsuba

How much work to multiply polys
$f = f_0 + f_1 t + \cdots + f_{19} t^{19}$,
$g = g_0 + g_1 t + \cdots + g_{19} t^{19}$?

Using the obvious method:
400 coeff mults, 361 coeff adds.

Faster: Write f as $F_0 + F_1 t^{10}$;
$F_0 = f_0 + f_1 t + \cdots + f_9 t^9$;
$F_1 = f_{10} + f_{11} t + \cdots + f_{19} t^9$.
Similarly write g as $G_0 + G_1 t^{10}$.

Then $fg = (F_0 + F_1)(G_0 + G_1) t^{10}$
$+ (F_0 G_0 - F_1 G_1 t^{10})(1 - t^{10})$.

20 adds for $F_0 + F_1$, $G_0 + G$
300 mults for three products
$F_0 G_0$, $F_1 G_1$, $(F_0 + F_1)(G_0 +$
243 adds for those products
9 adds for $F_0 G_0 - F_1 G_1 t^{10}$
with subs counted as adds
and with delayed negations.
19 adds for $\cdots (1 - t^{10})$.
19 adds to finish.

Total 300 mults, 310 adds.
Larger coefficients, slight ex
still saves time.

Can apply idea recursively
as poly degree grows.

Speedup: polynomial Karatsuba

How much work to multiply polys
$f = f_0 + f_1 t + \cdots + f_{19} t^{19}$,
$g = g_0 + g_1 t + \cdots + g_{19} t^{19}$?

Using the obvious method:
400 coeff mults, 361 coeff adds.

Faster: Write f as $F_0 + F_1 t^{10}$;
$F_0 = f_0 + f_1 t + \cdots + f_9 t^9$;
$F_1 = f_{10} + f_{11} t + \cdots + f_{19} t^9$.
Similarly write g as $G_0 + G_1 t^{10}$.

Then $fg = (F_0 + F_1)(G_0 + G_1) t^{10}$
$+ (F_0 G_0 - F_1 G_1 t^{10})(1 - t^{10})$.

20 adds for $F_0 + F_1$, $G_0 + G_1$.
300 mults for three products
$F_0 G_0$, $F_1 G_1$, $(F_0 + F_1)(G_0 + G_1)$.
243 adds for those products.
9 adds for $F_0 G_0 - F_1 G_1 t^{10}$
with subs counted as adds
and with delayed negations.
19 adds for $\cdots(1 - t^{10})$.
19 adds to finish.

Total 300 mults, 310 adds.
Larger coefficients, slight expense;
still saves time.

Can apply idea recursively
as poly degree grows.

: polynomial Karatsuba

ch work to multiply polys

$+ f_1 t + \cdots + f_{19} t^{19}$,

$- g_1 t + \cdots + g_{19} t^{19}$?

ne obvious method:

ff mults, 361 coeff adds.

Write f as $F_0 + F_1 t^{10}$;

$+ f_1 t + \cdots + f_9 t^9$;

$_0 + f_{11} t + \cdots + f_{19} t^9$.

write g as $G_0 + G_1 t^{10}$.

$= (F_0 + F_1)(G_0 + G_1) t^{10}$

$_0 - F_1 G_1 t^{10})(1 - t^{10})$.

20 adds for $F_0 + F_1$, $G_0 + G_1$.
300 mults for three products
$F_0 G_0$, $F_1 G_1$, $(F_0 + F_1)(G_0 + G_1)$.
243 adds for those products.
9 adds for $F_0 G_0 - F_1 G_1 t^{10}$
with subs counted as adds
and with delayed negations.
19 adds for $\cdots(1 - t^{10})$.
19 adds to finish.

Total 300 mults, 310 adds.
Larger coefficients, slight expense;
still saves time.

Can apply idea recursively
as poly degree grows.

Many ot

in polyn

"Toom,"

Increasi

polynom

$O(n \lg n$

to comp

Useful fo

that occ

In some

But Kar

for prim

on most

nial Karatsuba

o multiply polys
$\cdot + f_{19}t^{19}$,
$\cdot + g_{19}t^{19}$?

method:
61 coeff adds.

s $F_0 + F_1 t^{10}$;
$\cdots + f_9 t^9$;
$\cdots + f_{19} t^9$.
s $G_0 + G_1 t^{10}$.

$F_1)(G_0 + G_1)t^{10}$
$^{0})(1 - t^{10})$.

20 adds for $F_0 + F_1$, $G_0 + G_1$.
300 mults for three products
$F_0 G_0$, $F_1 G_1$, $(F_0 + F_1)(G_0 + G_1)$.
243 adds for those products.
9 adds for $F_0 G_0 - F_1 G_1 t^{10}$
with subs counted as adds
and with delayed negations.
19 adds for $\cdots(1 - t^{10})$.
19 adds to finish.

Total 300 mults, 310 adds.
Larger coefficients, slight expense;
still saves time.

Can apply idea recursively
as poly degree grows.

Many other algebr
in polynomial mult
"Toom," "FFT," e

Increasingly impor
polynomial degree
$O(n \lg n \lg \lg n)$ co
to compute n-coe

Useful for sizes of
that occur in cryp
In some cases, yes
But Karatsuba is t
for prime-field EC
on most current C

suba

polys

?

dds.

^{10};

t^9.

t^{10}.

$G_1)t^{10}$

).

20 adds for $F_0 + F_1$, $G_0 + G_1$.
300 mults for three products
F_0G_0, F_1G_1, $(F_0 + F_1)(G_0 + G_1)$.
243 adds for those products.
9 adds for $F_0G_0 - F_1G_1t^{10}$
with subs counted as adds
and with delayed negations.
19 adds for $\cdots(1 - t^{10})$.
19 adds to finish.

Total 300 mults, 310 adds.
Larger coefficients, slight expense;
still saves time.

Can apply idea recursively
as poly degree grows.

Many other algebraic speedu
in polynomial multiplication
"Toom," "FFT," etc.

Increasingly important as
polynomial degree grows.
$O(n \lg n \lg\lg n)$ coeff operat
to compute n-coeff product

Useful for sizes of n
that occur in cryptography?
In some cases, yes!
But Karatsuba is the limit
for prime-field ECC/ECDLP
on most current CPUs.

20 adds for $F_0 + F_1$, $G_0 + G_1$.
300 mults for three products
F_0G_0, F_1G_1, $(F_0 + F_1)(G_0 + G_1)$.
243 adds for those products.
9 adds for $F_0G_0 - F_1G_1t^{10}$
with subs counted as adds
and with delayed negations.
19 adds for $\cdots(1 - t^{10})$.
19 adds to finish.

Total 300 mults, 310 adds.
Larger coefficients, slight expense;
still saves time.

Can apply idea recursively
as poly degree grows.

Many other algebraic speedups
in polynomial multiplication:
"Toom," "FFT," etc.

Increasingly important as
polynomial degree grows.
$O(n \lg n \lg \lg n)$ coeff operations
to compute n-coeff product.

Useful for sizes of n
that occur in cryptography?
In some cases, yes!
But Karatsuba is the limit
for prime-field ECC/ECDLP
on most current CPUs.

for $F_0 + F_1$, $G_0 + G_1$.
ts for three products
$_1G_1$, $(F_0 + F_1)(G_0 + G_1)$.
s for those products.
or $F_0G_0 - F_1G_1t^{10}$
s counted as adds
n delayed negations.
for $\cdots(1 - t^{10})$.
to finish.

0 mults, 310 adds.
oefficients, slight expense;
es time.

ly idea recursively
degree grows.

Many other algebraic speedups in polynomial multiplication: "Toom," "FFT," etc.

Increasingly important as polynomial degree grows. $O(n \lg n \lg \lg n)$ coeff operations to compute n-coeff product.

Useful for sizes of n that occur in cryptography? In some cases, yes! But Karatsuba is the limit for prime-field ECC/ECDLP on most current CPUs.

Modular

How to

Can use
f mod p
Can mul
precomp
easily ad

Slight sp
"Montgo

F_1, $G_0 + G_1$.
e products
$+ F_1)(G_0 + G_1)$.
e products.
$F_1 G_1 t^{10}$
as adds
negations.
$- t^{10})$.

310 adds.
, slight expense;

cursively
ows.

Many other algebraic speedups in polynomial multiplication: "Toom," "FFT," etc.

Increasingly important as polynomial degree grows. $O(n \lg n \lg \lg n)$ coeff operations to compute n-coeff product.

Useful for sizes of n that occur in cryptography? In some cases, yes! But Karatsuba is the limit for prime-field ECC/ECDLP on most current CPUs.

<u>Modular reduction</u>

How to compute f

Can use definition
$f \bmod p = f - p$ |
Can multiply f by
precomputed $1/p$
easily adjust to ob

Slight speedup: "2
"Montgomery redu

G_1.
s
$+ G_1$).
.
pense;

Many other algebraic speedups in polynomial multiplication: "Toom," "FFT," etc.

Increasingly important as polynomial degree grows. $O(n \lg n \lg \lg n)$ coeff operations to compute n-coeff product.

Useful for sizes of n that occur in cryptography? In some cases, yes! But Karatsuba is the limit for prime-field ECC/ECDLP on most current CPUs.

<u>Modular reduction</u>

How to compute $f \bmod p$?

Can use definition: $f \bmod p = f - p \lfloor f/p \rfloor$. Can multiply f by a precomputed $1/p$ approxima easily adjust to obtain $\lfloor f/p$

Slight speedup: "2-adic inve "Montgomery reduction."

Many other algebraic speedups in polynomial multiplication: "Toom," "FFT," etc.

Increasingly important as polynomial degree grows. $O(n \lg n \lg \lg n)$ coeff operations to compute n-coeff product.

Useful for sizes of n that occur in cryptography? In some cases, yes! But Karatsuba is the limit for prime-field ECC/ECDLP on most current CPUs.

Modular reduction

How to compute $f \bmod p$?

Can use definition: $f \bmod p = f - p \lfloor f/p \rfloor$. Can multiply f by a precomputed $1/p$ approximation; easily adjust to obtain $\lfloor f/p \rfloor$.

Slight speedup: "2-adic inverse"; "Montgomery reduction."

ther algebraic speedups
omial multiplication:
" "FFT," etc.

ngly important as
ial degree grows.
$\lg\lg n)$ coeff operations
ute n-coeff product.

or sizes of n
ur in cryptography?
cases, yes!
atsuba is the limit
e-field ECC/ECDLP
current CPUs.

Modular reduction

How to compute $f \bmod p$?

Can use definition:
$f \bmod p = f - p\lfloor f/p \rfloor$.
Can multiply f by a
precomputed $1/p$ approximation;
easily adjust to obtain $\lfloor f/p \rfloor$.

Slight speedup: "2-adic inverse";
"Montgomery reduction."

e.g. 314

Precomp
$\lfloor 100000$
$= 36787$

Comput
314159
$= 11557$

Comput
3141592
$= 57823$
Oops, to
578230
306402

raic speedups
tiplication:
etc.

tant as
grows.
oeff operations
ff product.

n
tography?
!
the limit
C/ECDLP
CPUs.

Modular reduction

How to compute $f \bmod p$?

Can use definition:
$f \bmod p = f - p \lfloor f/p \rfloor$.
Can multiply f by a
precomputed $1/p$ approximation;
easily adjust to obtain $\lfloor f/p \rfloor$.

Slight speedup: "2-adic inverse";
"Montgomery reduction."

e.g. 31415926535

Precompute
$\lfloor 1000000000000/2$
$= 3678796$.

Compute
$314159 \cdot 3678796$
$= 1155726872564$

Compute
$31415926535 8 - 1$
$= 578230$.
Oops, too big:
$578230 - 271828$
$306402 - 271828$

ups

:

tions

r

<u>Modular reduction</u>

How to compute $f \bmod p$?

Can use definition:
$f \bmod p = f - p\lfloor f/p \rfloor$.
Can multiply f by a precomputed $1/p$ approximation; easily adjust to obtain $\lfloor f/p \rfloor$.

Slight speedup: "2-adic inverse"; "Montgomery reduction."

e.g. 314159265358 mod 271

Precompute
$\lfloor 1000000000000/271828 \rfloor$
$= 3678796$.

Compute
$314159 \cdot 3678796$
$= 1155726872564$.

Compute
$314159265358 - 1155726 \cdot 2$
$= 578230$.
Oops, too big:
$578230 - 271828 = 306402$.
$306402 - 271828 = 34574$.

Modular reduction

How to compute f mod p?

Can use definition:
$f \bmod p = f - p\lfloor f/p \rfloor$.
Can multiply f by a precomputed $1/p$ approximation; easily adjust to obtain $\lfloor f/p \rfloor$.

Slight speedup: "2-adic inverse"; "Montgomery reduction."

e.g. 314159265358 mod 271828:

Precompute
$\lfloor 1000000000000/271828 \rfloor$
$= 3678796$.

Compute
$314159 \cdot 3678796$
$= 1155726872564$.

Compute
$314159265358 - 1155726 \cdot 271828$
$= 578230$.
Oops, too big:
$578230 - 271828 = 306402$.
$306402 - 271828 = 34574$.

r reduction

compute f mod p?

definition:

$\ = f - p\lfloor f/p \rfloor$.

ltiply f by a

puted $1/p$ approximation;

djust to obtain $\lfloor f/p \rfloor$.

peedup: “2-adic inverse”;

omery reduction.”

e.g. 314159265358 mod 271828:

Precompute

$\lfloor 1000000000000/271828 \rfloor$

$= 3678796$.

Compute

$314159 \cdot 3678796$

$= 1155726872564$.

Compute

$314159265358 - 1155726 \cdot 271828$

$= 578230$.

Oops, too big:

$578230 - 271828 = 306402$.

$306402 - 271828 = 34574$.

We can

p is chos

to make

Special

for $\mathbf{F}_p^*$, C

but not

gls1271:

with deg

Curve25

NIST P-

secp112r

Divides

n

f mod p?

:

$\lfloor f/p \rfloor$.

a

approximation;

tain $\lfloor f/p \rfloor$.

2-adic inverse";

uction."

e.g. 314159265358 mod 271828:

Precompute
$\lfloor 1000000000000/271828 \rfloor$
$= 3678796$.

Compute
$314159 \cdot 3678796$
$= 1155726872564$.

Compute
$314159265358 - 1155726 \cdot 271828$
$= 578230$.
Oops, too big:
$578230 - 271828 = 306402$.
$306402 - 271828 = 34574$.

We can do better:

p is chosen with a

to make f mod p

Special primes hur

for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$),

but not for elliptic

gls1271: $p = 2^{127}$

with degree-2 exte

Curve25519: $p =$

NIST P-224: $p =$

secp112r1: $p = (2$

Divides special for

tion;

⌋.

erse";

e.g. 314159265358 mod 271828:

Precompute
$\lfloor 1000000000000/271828 \rfloor$
$= 3678796$.

Compute
$314159 \cdot 3678796$
$= 1155726872564$.

Compute
$314159265358 - 1155726 \cdot 271828$
$= 578230$.
Oops, too big:
$578230 - 271828 = 306402$.
$306402 - 271828 = 34574$.

We can do better: normally
p is chosen with a special fo
to make f mod p much fast

Special primes hurt security
for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$), etc.,
but not for elliptic curves!

gls1271: $p = 2^{127} - 1$,
with degree-2 extension.

Curve25519: $p = 2^{255} - 19.$

NIST P-224: $p = 2^{224} - 2^{96}$

secp112r1: $p = (2^{128} - 3)/7$
Divides special form.

e.g. 314159265358 mod 271828:

Precompute
$\lfloor 1000000000000/271828 \rfloor$
$= 3678796$.

Compute
$314159 \cdot 3678796$
$= 1155726872564$.

Compute
$314159265358 - 1155726 \cdot 271828$
$= 578230$.
Oops, too big:
$578230 - 271828 = 306402$.
$306402 - 271828 = 34574$.

We can do better: normally
p is chosen with a special form
to make f mod p much faster.

Special primes hurt security
for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$), etc.,
but not for elliptic curves!

gls1271: $p = 2^{127} - 1$,
with degree-2 extension.

Curve25519: $p = 2^{255} - 19$.

NIST P-224: $p = 2^{224} - 2^{96} + 1$.

secp112r1: $p = (2^{128} - 3)/76439$.
Divides special form.

159265358 mod 271828:

pute

0000000/271828⌋

796.

e

3678796

726872564.

e

265358 − 1155726 · 271828

30.

oo big:

− 271828 = 306402.

− 271828 = 34574.

We can do better: normally p is chosen with a special form to make f mod p much faster.

Special primes hurt security for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$), etc., but not for elliptic curves!

gls1271: $p = 2^{127} - 1$, with degree-2 extension.

Curve25519: $p = 2^{255} - 19$.

NIST P-224: $p = 2^{224} - 2^{96} + 1$.

secp112r1: $p = (2^{128} - 3)/76439$. *Divides* special form.

Small ex

Then 10

e.g. 314

314159

314159(

−94247

−67711

Easily ac

to the ra

by addin

e.g. −67

8 mod 271828:

271828⌋

.

155726 · 271828

= 306402.

= 34574.

We can do better: normally p is chosen with a special form to make f mod p much faster.

Special primes hurt security for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$), etc., but not for elliptic curves!

gls1271: $p = 2^{127} - 1$, with degree-2 extension.

Curve25519: $p = 2^{255} - 19$.

NIST P-224: $p = 2^{224} - 2^{96} + 1$.

secp112r1: $p = (2^{128} - 3)/76439$. *Divides* special form.

Small example: p

Then $1000000a +$

e.g. 31415926535

$314159 \cdot 1000000$

$314159(-3) + 265$

$-942477 + 26535$

-677119.

Easily adjust $b - 3$

to the range $\{0, 1,$

by adding/subtrac

e.g. $-677119 \equiv 3$

.828:

271828

.

We can do better: normally p is chosen with a special form to make f mod p much faster.

Special primes hurt security for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$), etc., but not for elliptic curves!

gls1271: $p = 2^{127} - 1$, with degree-2 extension.

Curve25519: $p = 2^{255} - 19$.

NIST P-224: $p = 2^{224} - 2^{96} + 1$.

secp112r1: $p = (2^{128} - 3)/76439$. *Divides* special form.

Small example: $p = 100000$

Then $1000000a + b \equiv b - 3$

e.g. $314159265358 =$
$314159 \cdot 1000000 + 265358$
$314159(-3) + 265358 =$
$-942477 + 265358 =$
-677119.

Easily adjust $b - 3a$ to the range $\{0, 1, \ldots, p - 1$ by adding/subtracting a few e.g. $-677119 \equiv 322884$.

We can do better: normally p is chosen with a special form to make f mod p much faster.

Special primes hurt security for $\mathbf{F}_p^*$, Clock($\mathbf{F}_p$), etc., but not for elliptic curves!

gls1271: $p = 2^{127} - 1$, with degree-2 extension.

Curve25519: $p = 2^{255} - 19$.

NIST P-224: $p = 2^{224} - 2^{96} + 1$.

secp112r1: $p = (2^{128} - 3)/76439$. *Divides* special form.

Small example: $p = 1000003$. Then $1000000a + b \equiv b - 3a$.

e.g. $314159265358 =$
$314159 \cdot 1000000 + 265358 \equiv$
$314159(-3) + 265358 =$
$-942477 + 265358 =$
-677119.

Easily adjust $b - 3a$ to the range $\{0, 1, \ldots, p - 1\}$ by adding/subtracting a few p's: e.g. $-677119 \equiv 322884$.

do better: normally
sen with a special form
f mod p much faster.

primes hurt security
Clock($\mathbf{F}_p$), etc.,
for elliptic curves!

$p = 2^{127} - 1$,
gree-2 extension.

519: $p = 2^{255} - 19$.

224: $p = 2^{224} - 2^{96} + 1$.

r1: $p = (2^{128} - 3)/76439$.
special form.

Small example: $p = 1000003$.
Then $1000000a + b \equiv b - 3a$.

e.g. $314159265358 =$
$314159 \cdot 1000000 + 265358 \equiv$
$314159(-3) + 265358 =$
$-942477 + 265358 =$
-677119.

Easily adjust $b - 3a$
to the range $\{0, 1, \ldots, p - 1\}$
by adding/subtracting a few p's:
e.g. $-677119 \equiv 322884$.

Hmmm,

Conditio
(Also da
branch t
Can elim
but adju

Speedup
for inter
"Lazy re
Adjust o

$b - 3a$ is
to contin

normally
special form
much faster.

rt security
etc.,
curves!

$- 1$,
ension.

$2^{255} - 19$.

$2^{224} - 2^{96} + 1$.

$^{128} - 3)/76439$.
rm.

Small example: $p = 1000003$.
Then $1000000a + b \equiv b - 3a$.

e.g. $314159265358 =$
$314159 \cdot 1000000 + 265358 \equiv$
$314159(-3) + 265358 =$
$-942477 + 265358 =$
-677119.

Easily adjust $b - 3a$
to the range $\{0, 1, \ldots, p - 1\}$
by adding/subtracting a few p's:
e.g. $-677119 \equiv 322884$.

Hmmm, is adjustn

Conditional branch
(Also dangerous fo
branch timing leak
Can eliminate the
but adjustment isn

Speedup: Skip the
for intermediate re
"Lazy reduction."
Adjust only for ou

$b - 3a$ is small enc
to continue compu

rm

er.

${}^6 + 1$.

76439.

Small example: $p = 1000003$.
Then $1000000a + b \equiv b - 3a$.

e.g. $314159265358 =$
$314159 \cdot 1000000 + 265358 \equiv$
$314159(-3) + 265358 =$
$-942477 + 265358 =$
-677119.

Easily adjust $b - 3a$
to the range $\{0, 1, \ldots, p - 1\}$
by adding/subtracting a few p's:
e.g. $-677119 \equiv 322884$.

Hmmm, is adjustment so ea

Conditional branches are slo
(Also dangerous for defende
branch timing leaks secrets.)
Can eliminate the branches,
but adjustment isn't free.

Speedup: Skip the adjustme
for intermediate results.
"Lazy reduction."
Adjust only for output.

$b - 3a$ is small enough
to continue computations.

Small example: $p = 1000003$.
Then $1000000a + b \equiv b - 3a$.

e.g. $314159265358 =$
$314159 \cdot 1000000 + 265358 \equiv$
$314159(-3) + 265358 =$
$-942477 + 265358 =$
-677119.

Easily adjust $b - 3a$
to the range $\{0, 1, \ldots, p - 1\}$
by adding/subtracting a few p's:
e.g. $-677119 \equiv 322884$.

Hmmm, is adjustment so easy?

Conditional branches are slow.
(Also dangerous for defenders:
branch timing leaks secrets.)
Can eliminate the branches,
but adjustment isn't free.

Speedup: Skip the adjustment
for intermediate results.
"Lazy reduction."
Adjust only for output.

$b - 3a$ is small enough
to continue computations.

ample: $p = 1000003$.

$00000a + b \equiv b - 3a$.

$159265358 =$

$1000000 + 265358 \equiv$

$-3) + 265358 =$

$7 + 265358 =$

9.

djust $b - 3a$

ange $\{0, 1, \ldots, p-1\}$

ng/subtracting a few p's:

$7119 \equiv 322884$.

Hmmm, is adjustment so easy?

Conditional branches are slow.
(Also dangerous for defenders:
branch timing leaks secrets.)
Can eliminate the branches,
but adjustment isn't free.

Speedup: Skip the adjustment
for intermediate results.
"Lazy reduction."
Adjust only for output.

$b - 3a$ is small enough
to continue computations.

Can dela

multiplic

e.g. To s

in $\mathbf{Z}/10$

$3t^5 + 1t$

obtainin

$14t^7 + 4$

$82t^3 + 4$

Reduce:

$(-3c_i)t^i$

$64t^3 - 3$

Carry: 8

$1t^3 + 2t$

$= 1000003.$

$b \equiv b - 3a.$

$3 =$

$+ 265358 \equiv$

$5358 =$

$8 =$

$3a$

$, \ldots, p - 1\}$

ting a few p's:

22884.

Hmmm, is adjustment so easy?

Conditional branches are slow. (Also dangerous for defenders: branch timing leaks secrets.) Can eliminate the branches, but adjustment isn't free.

Speedup: Skip the adjustment for intermediate results. "Lazy reduction." Adjust only for output.

$b - 3a$ is small enough to continue computations.

Can delay carries u

multiplication by 3

e.g. To square 314

in $\mathbf{Z}/1000003$: Sq

$3t^5 + 1t^4 + 4t^3 +$

obtaining $9t^{10} + 6$

$14t^7 + 48t^6 + 72t^5$

$82t^3 + 43t^2 + 90t^1$

Reduce: replace (c

$(-3c_i)t^i$, obtainin

$64t^3 - 32t^2 + 48t^1$

Carry: $8t^6 - 4t^5 -$

$1t^3 + 2t^2 + 2t^1 -$

3.

a.

$\equiv$

$\}$

p's:

Hmmm, is adjustment so easy?

Conditional branches are slow. (Also dangerous for defenders: branch timing leaks secrets.) Can eliminate the branches, but adjustment isn't free.

Speedup: Skip the adjustment for intermediate results. "Lazy reduction." Adjust only for output.

$b - 3a$ is small enough to continue computations.

Can delay carries until after multiplication by 3.

e.g. To square 314159 in $\mathbf{Z}/1000003$: Square poly $3t^5 + 1t^4 + 4t^3 + 1t^2 + 5t^1$ obtaining $9t^{10} + 6t^9 + 25t^8$ $14t^7 + 48t^6 + 72t^5 + 59t^4 +$ $82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce: replace $(c_i)t^{6+i}$ by $(-3c_i)t^i$, obtaining $72t^5 + 3$ $64t^3 - 32t^2 + 48t^1 - 63t^0$.

Carry: $8t^6 - 4t^5 - 2t^4 +$ $1t^3 + 2t^2 + 2t^1 - 3t^0$.

Hmmm, is adjustment so easy?

Conditional branches are slow.
(Also dangerous for defenders: branch timing leaks secrets.)
Can eliminate the branches, but adjustment isn't free.

Speedup: Skip the adjustment for intermediate results.
"Lazy reduction."
Adjust only for output.

$b - 3a$ is small enough to continue computations.

Can delay carries until after multiplication by 3.

e.g. To square 314159 in $\mathbf{Z}/1000003$: Square poly $3t^5 + 1t^4 + 4t^3 + 1t^2 + 5t^1 + 9t^0$, obtaining $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce: replace $(c_i)t^{6+i}$ by $(-3c_i)t^i$, obtaining $72t^5 + 32t^4 + 64t^3 - 32t^2 + 48t^1 - 63t^0$.

Carry: $8t^6 - 4t^5 - 2t^4 + 1t^3 + 2t^2 + 2t^1 - 3t^0$.

is adjustment so easy?

onal branches are slow.
ngerous for defenders:
timing leaks secrets.)
inate the branches,
stment isn't free.

: Skip the adjustment
mediate results.
eduction."
only for output.

s small enough
nue computations.

Can delay carries until after multiplication by 3.

e.g. To square 314159 in $\mathbf{Z}/1000003$: Square poly $3t^5 + 1t^4 + 4t^3 + 1t^2 + 5t^1 + 9t^0$, obtaining $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce: replace $(c_i)t^{6+i}$ by $(-3c_i)t^i$, obtaining $72t^5 + 32t^4 + 64t^3 - 32t^2 + 48t^1 - 63t^0$.

Carry: $8t^6 - 4t^5 - 2t^4 + 1t^3 + 2t^2 + 2t^1 - 3t^0$.

To minir
mix redu
carrying

e.g. Star
$25t^8 + 1$
$82t^3 + 4$

Reduce
$t^5 \to t^6$:
$5t^5+2t^4$

Finish re
$64t^3 - 3$
$t^0 \to t^1$
$-4t^5 - 2$

ment so easy?

hes are slow.
or defenders:
ks secrets.)
branches,
n't free.

e adjustment
esults.

tput.

ough
utations.

Can delay carries until after multiplication by 3.

e.g. To square 314159 in $\mathbf{Z}/1000003$: Square poly $3t^5 + 1t^4 + 4t^3 + 1t^2 + 5t^1 + 9t^0$, obtaining $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce: replace $(c_i)t^{6+i}$ by $(-3c_i)t^i$, obtaining $72t^5 + 32t^4 + 64t^3 - 32t^2 + 48t^1 - 63t^0$.

Carry: $8t^6 - 4t^5 - 2t^4 + 1t^3 + 2t^2 + 2t^1 - 3t^0$.

To minimize poly
mix reduction and
carrying the top s

e.g. Start from squ
$25t^8 + 14t^7 + 48t^6$
$82t^3 + 43t^2 + 90t$

Reduce $t^{10} \to t^4$
$t^5 \to t^6$: $6t^9 + 25t$
$5t^5 + 2t^4 + 82t^3 + 4$

Finish reduction:
$64t^3 - 32t^2 + 48t$
$t^0 \to t^1 \to t^2 \to$
$-4t^5 - 2t^4 + 1t^3 +$

sy?

w.

rs:

)

ent

Can delay carries until after multiplication by 3.

e.g. To square 314159 in $\mathbf{Z}/1000003$: Square poly $3t^5 + 1t^4 + 4t^3 + 1t^2 + 5t^1 + 9t^0$, obtaining $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce: replace $(c_i)t^{6+i}$ by $(-3c_i)t^i$, obtaining $72t^5 + 32t^4 + 64t^3 - 32t^2 + 48t^1 - 63t^0$.

Carry: $8t^6 - 4t^5 - 2t^4 + 1t^3 + 2t^2 + 2t^1 - 3t^0$.

To minimize poly degree, mix reduction and carrying, carrying the top sooner.

e.g. Start from square $9t^{10}$ + $25t^8 + 14t^7 + 48t^6 + 72t^5 +$ $82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce $t^{10} \to t^4$ and carry $t^5 \to t^6$: $6t^9 + 25t^8 + 14t^7 +$ $5t^5 + 2t^4 + 82t^3 + 43t^2 + 90t^1$

Finish reduction: $-5t^5 + 2t$ $64t^3 - 32t^2 + 48t^1 - 87t^0$.
$t^0 \to t^1 \to t^2 \to t^3 \to t^4$
$-4t^5 - 2t^4 + 1t^3 + 2t^2 - 1t^1$

Can delay carries until after multiplication by 3.

e.g. To square 314159 in $\mathbf{Z}/1000003$: Square poly $3t^5 + 1t^4 + 4t^3 + 1t^2 + 5t^1 + 9t^0$, obtaining $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce: replace $(c_i)t^{6+i}$ by $(-3c_i)t^i$, obtaining $72t^5 + 32t^4 + 64t^3 - 32t^2 + 48t^1 - 63t^0$.

Carry: $8t^6 - 4t^5 - 2t^4 + 1t^3 + 2t^2 + 2t^1 - 3t^0$.

To minimize poly degree, mix reduction and carrying, carrying the top sooner.

e.g. Start from square $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce $t^{10} \to t^4$ and carry $t^4 \to t^5 \to t^6$: $6t^9 + 25t^8 + 14t^7 + 56t^6 - 5t^5 + 2t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Finish reduction: $-5t^5 + 2t^4 + 64t^3 - 32t^2 + 48t^1 - 87t^0$. Carry $t^0 \to t^1 \to t^2 \to t^3 \to t^4 \to t^5$: $-4t^5 - 2t^4 + 1t^3 + 2t^2 - 1t^1 + 3t^0$.

ay carries until after
cation by 3.

square 314159
00003: Square poly
$^4 + 4t^3 + 1t^2 + 5t^1 + 9t^0$,
g $9t^{10} + 6t^9 + 25t^8 +$
$8t^6 + 72t^5 + 59t^4 +$
$3t^2 + 90t^1 + 81t^0$.

replace $(c_i)t^{6+i}$ by
, obtaining $72t^5 + 32t^4 +$
$2t^2 + 48t^1 - 63t^0$.

$t^6 - 4t^5 - 2t^4 +$
$^2 + 2t^1 - 3t^0$.

To minimize poly degree, mix reduction and carrying, carrying the top sooner.

e.g. Start from square $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce $t^{10} \to t^4$ and carry $t^4 \to t^5 \to t^6$: $6t^9 + 25t^8 + 14t^7 + 56t^6 - 5t^5 + 2t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Finish reduction: $-5t^5 + 2t^4 + 64t^3 - 32t^2 + 48t^1 - 87t^0$. Carry $t^0 \to t^1 \to t^2 \to t^3 \to t^4 \to t^5$: $-4t^5 - 2t^4 + 1t^3 + 2t^2 - 1t^1 + 3t^0$.

Speedup

$p = 2^{61}$

Five coe
$f_4t^4 + f$
Most co

Square ·
Coeff of

Reduce:
$\cdots + (2$
Coeff co
Very litt
addition
on 32-bi

until after

3.

4159

uare poly

$1t^2 + 5t^1 + 9t^0$,

$t^9 + 25t^8 +$

$^5 + 59t^4 +$

$^1 + 81t^0$.

$c_i)t^{6+i}$ by

g $72t^5 + 32t^4 +$

$^1 - 63t^0$.

$- 2t^4 +$

$3t^0$.

To minimize poly degree,
mix reduction and carrying,
carrying the top sooner.

e.g. Start from square $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce $t^{10} \to t^4$ and carry $t^4 \to t^5 \to t^6$: $6t^9 + 25t^8 + 14t^7 + 56t^6 - 5t^5 + 2t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Finish reduction: $-5t^5 + 2t^4 + 64t^3 - 32t^2 + 48t^1 - 87t^0$. Carry $t^0 \to t^1 \to t^2 \to t^3 \to t^4 \to t^5$: $-4t^5 - 2t^4 + 1t^3 + 2t^2 - 1t^1 + 3t^0$.

Speedup: non-inte

$p = 2^{61} - 1$.

Five coeffs in radi
$f_4t^4 + f_3t^3 + f_2t^2$
Most coeffs could

Square $\cdots + 2(f_4 f$
Coeff of t^5 could b

Reduce: $2^{65} = 2^4$
$\cdots + (2^5(f_4 f_1 +$
Coeff could be $>$
Very little room fo
additions, delayed
on 32-bit platform

$+ 9t^0$,

$+$

$-$

$32t^4 +$

To minimize poly degree, mix reduction and carrying, carrying the top sooner.

e.g. Start from square $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce $t^{10} \to t^4$ and carry $t^4 \to t^5 \to t^6$: $6t^9 + 25t^8 + 14t^7 + 56t^6 - 5t^5 + 2t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Finish reduction: $-5t^5 + 2t^4 + 64t^3 - 32t^2 + 48t^1 - 87t^0$. Carry $t^0 \to t^1 \to t^2 \to t^3 \to t^4 \to t^5$: $-4t^5 - 2t^4 + 1t^3 + 2t^2 - 1t^1 + 3t^0$.

Speedup: non-integer radix

$p = 2^{61} - 1$.

Five coeffs in radix 2^{13}? $f_4t^4 + f_3t^3 + f_2t^2 + f_1t^1 +$

Most coeffs could be 2^{12}.

Square $\cdots + 2(f_4f_1 + f_3f_2)t$

Coeff of t^5 could be $> 2^{25}$.

Reduce: $2^{65} = 2^4$ in $\mathbf{Z}/(2^{61}$

$\cdots + (2^5(f_4f_1 + f_3f_2) + f$

Coeff could be $> 2^{29}$.

Very little room for additions, delayed carries, et on 32-bit platforms.

To minimize poly degree, mix reduction and carrying, carrying the top sooner.

e.g. Start from square $9t^{10} + 6t^9 + 25t^8 + 14t^7 + 48t^6 + 72t^5 + 59t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Reduce $t^{10} \to t^4$ and carry $t^4 \to t^5 \to t^6$: $6t^9 + 25t^8 + 14t^7 + 56t^6 - 5t^5 + 2t^4 + 82t^3 + 43t^2 + 90t^1 + 81t^0$.

Finish reduction: $-5t^5 + 2t^4 + 64t^3 - 32t^2 + 48t^1 - 87t^0$. Carry $t^0 \to t^1 \to t^2 \to t^3 \to t^4 \to t^5$: $-4t^5 - 2t^4 + 1t^3 + 2t^2 - 1t^1 + 3t^0$.

<u>Speedup: non-integer radix</u>

$p = 2^{61} - 1$.

Five coeffs in radix 2^{13}? $f_4t^4 + f_3t^3 + f_2t^2 + f_1t^1 + f_0t^0$. Most coeffs could be 2^{12}.

Square $\cdots + 2(f_4f_1 + f_3f_2)t^5 + \cdots$. Coeff of t^5 could be $> 2^{25}$.

Reduce: $2^{65} = 2^4$ in $\mathbf{Z}/(2^{61} - 1)$; $\cdots + (2^5(f_4f_1 + f_3f_2) + f_0^2)t^0$. Coeff could be $> 2^{29}$. Very little room for additions, delayed carries, etc. on 32-bit platforms.

mize poly degree,
uction and carrying,
the top sooner.

rt from square $9t^{10}+6t^9+$
$4t^7+48t^6+72t^5+59t^4+$
$3t^2+90t^1+81t^0$.

$t^{10}\to t^4$ and carry $t^4\to$
$6t^9+25t^8+14t^7+56t^6-$
$+82t^3+43t^2+90t^1+81t^0$.

eduction: $-5t^5+2t^4+$
$2t^2+48t^1-87t^0$. Carry
$\to t^2\to t^3\to t^4\to t^5$:
$2t^4+1t^3+2t^2-1t^1+3t^0$.

Speedup: non-integer radix

$p=2^{61}-1$.

Five coeffs in radix 2^{13}?
$f_4t^4+f_3t^3+f_2t^2+f_1t^1+f_0t^0$.
Most coeffs could be 2^{12}.

Square $\cdots+2(f_4f_1+f_3f_2)t^5+\cdots$.
Coeff of t^5 could be $>2^{25}$.

Reduce: $2^{65}=2^4$ in $\mathbf{Z}/(2^{61}-1)$;
$\cdots+(2^5(f_4f_1+f_3f_2)+f_0^2)t^0$.
Coeff could be $>2^{29}$.
Very little room for
additions, delayed carries, etc.
on 32-bit platforms.

Scaled:
f_4 is mu
f_3 is mu
f_2 is mu
f_1 is mu
f_0 is mu
$\cdots+(2$

Better:
f_4 is mu
f_3 is mu
f_2 is mu
f_1 is mu
f_0 is mu
Saves a

degree,
carrying,
ooner.

uare $9t^{10} + 6t^9 +$
$+ 72t^5 + 59t^4 +$
$^1 + 81t^0$.

and carry $t^4 \to$
$^8 + 14t^7 + 56t^6 -$
$3t^2 + 90t^1 + 81t^0$.

$-5t^5 + 2t^4 +$
$^1 - 87t^0$. Carry
$t^3 \to t^4 \to t^5$:
$2t^2 - 1t^1 + 3t^0$.

Speedup: non-integer radix

$p = 2^{61} - 1$.

Five coeffs in radix 2^{13}?
$f_4 t^4 + f_3 t^3 + f_2 t^2 + f_1 t^1 + f_0 t^0$.
Most coeffs could be 2^{12}.

Square $\cdots + 2(f_4 f_1 + f_3 f_2) t^5 + \cdots$.
Coeff of t^5 could be $> 2^{25}$.

Reduce: $2^{65} = 2^4$ in $\mathbf{Z}/(2^{61} - 1)$;
$\cdots + (2^5(f_4 f_1 + f_3 f_2) + f_0^2) t^0$.
Coeff could be $> 2^{29}$.
Very little room for
additions, delayed carries, etc.
on 32-bit platforms.

Scaled: Evaluate a
f_4 is multiple of 2
f_3 is multiple of 2
f_2 is multiple of 2
f_1 is multiple of 2
f_0 is multiple of 2
$\cdots + (2^{-60}(f_4 f_1 -$

Better: Non-intege
f_4 is multiple of 2
f_3 is multiple of 2
f_2 is multiple of 2
f_1 is multiple of 2
f_0 is multiple of 2
Saves a few bits ir

$+6t^9+$

$59t^4+$

$t^4 \rightarrow$

$56t^6-$

$+81t^0.$

$^4+$

Carry

$\rightarrow t^5$:

$+3t^0.$

Speedup: non-integer radix

$p = 2^{61} - 1$.

Five coeffs in radix 2^{13}?

$f_4t^4 + f_3t^3 + f_2t^2 + f_1t^1 + f_0t^0$.

Most coeffs could be 2^{12}.

Square $\cdots + 2(f_4f_1 + f_3f_2)t^5 + \cdots$.

Coeff of t^5 could be $> 2^{25}$.

Reduce: $2^{65} = 2^4$ in $\mathbf{Z}/(2^{61}-1)$;

$\cdots + (2^5(f_4f_1 + f_3f_2) + f_0^2)t^0$.

Coeff could be $> 2^{29}$.

Very little room for additions, delayed carries, etc. on 32-bit platforms.

Scaled: Evaluate at $t = 1$.

f_4 is multiple of 2^{52};

f_3 is multiple of 2^{39};

f_2 is multiple of 2^{26};

f_1 is multiple of 2^{13};

f_0 is multiple of 2^0. Reduce

$\cdots + (2^{-60}(f_4f_1 + f_3f_2) +$

Better: Non-integer radix 2^1

f_4 is multiple of 2^{49};

f_3 is multiple of 2^{37};

f_2 is multiple of 2^{25};

f_1 is multiple of 2^{13};

f_0 is multiple of 2^0.

Saves a few bits in coeffs.

Speedup: non-integer radix

$p = 2^{61} - 1$.

Five coeffs in radix 2^{13}?
$f_4t^4 + f_3t^3 + f_2t^2 + f_1t^1 + f_0t^0$.
Most coeffs could be 2^{12}.

Square $\cdots + 2(f_4f_1 + f_3f_2)t^5 + \cdots$.
Coeff of t^5 could be $> 2^{25}$.

Reduce: $2^{65} = 2^4$ in $\mathbf{Z}/(2^{61} - 1)$;
$\cdots + (2^5(f_4f_1 + f_3f_2) + f_0^2)t^0$.
Coeff could be $> 2^{29}$.
Very little room for
additions, delayed carries, etc.
on 32-bit platforms.

Scaled: Evaluate at $t = 1$.
f_4 is multiple of 2^{52};
f_3 is multiple of 2^{39};
f_2 is multiple of 2^{26};
f_1 is multiple of 2^{13};
f_0 is multiple of 2^0. Reduce:
$\cdots + (2^{-60}(f_4f_1 + f_3f_2) + f_0^2)t^0$.

Better: Non-integer radix $2^{12.2}$.
f_4 is multiple of 2^{49};
f_3 is multiple of 2^{37};
f_2 is multiple of 2^{25};
f_1 is multiple of 2^{13};
f_0 is multiple of 2^0.
Saves a few bits in coeffs.