

Re: sparse polynomial arithmetic

Source: <http://sci.tech--archive.net/Archive/sci.math.symbolic/2008-04/msg00022.html>

- *From:* Roman Pearce <rpearcea@xxxxxxxxxx>
 - *Date:* Tue, 1 Apr 2008 18:47:45 -0700 (PDT)
-

On Apr 1, 4:45 pm, rjf <fate...@xxxxxxxxxx> wrote:

It seems to be fast for polynomials with a billion terms.
Could you run tests with polynomials with, say, 20 terms, done 100 million times?

In that case it would make sense to run a different algorithm. For example, you could multiply out all of the terms and sort them in memory, and add up coefficients in one pass. For larger polynomials you multiply blocks and merge them with a geobucket, using two dynamic arrays for all temporary storage as described here:

http://www.cecm.sfu.ca/~rpearcea/sdmp/sdmp_paper.pdf

We didn't optimize for this case, but it is interesting so here are the times. Maple cheats by remembering the result. Pari has an advantage because it uses a dense representation so there is no sorting or monomial operations. Intel Core2 Xeon 3.0 GHz, 64-bit, with 16GB RAM:

Latency benchmark (dense)

```
f := (x+1)^20;
```

```
g := f+1;
```

```
multiply p := f*g; 10^6 times
```

```
# stan.cecm.sfu.ca p := f*g
```

```
sdmp Apr 2008 monomial=1 word 30.940
```

```
Trip v0.99 double precision 14.024
```

```
Trip v0.99 rational numbers 41.561
```

```
Pari 2.3.3 (w/ GMP) 23.412 (dense)
```

```
Magma V2.14-7 32.400
```

```
Singular 3-0-4 60.940
```

```
Maple 11 6.778 (remembered)
```

```
# sdmp input
```

```
f := sdmp(expand((x+1)^20), plex(x));
```

```
g := sdmp(expand((x+1)^20+1), plex(x));
```

```
infolevel[sdmp] := 0;
```

```
TIMER := time();
```

```

to 10^6 do
p := sdmp:-Multiply(f,g);
end do:
time()-TIMER;

# trip input
_modenum = NUMDBL;
f = (x+1)^20$
g = f+1$
time_s;
for n = 1 to 10^6 { p = f*g$}$
time_t;
_modenum = NUMRATMP;

# pari input
f = (x+1)^20;
g = f+1;
gettime();
for(n=1,10^6,p = f*g);
print(gettime());

# Magma input
P<x> := PolynomialRing(RationalField(),1);
f := (x+1)^20;
g := f+1;
TIMER := Cputime();
for i := 1 to 10^6 do
p := f*g;
end for;
Cputime(TIMER);

# singular input
ring R=0,(x),lp;
poly f = (x+1)^20;
poly g = f+1;
poly p;
int TIMER = timer;
for (int i=0; i < 10^6; i=i+1) { p = f*g; }
timer-TIMER;

# Maple input
f := expand((x+1)^20):
g := f+1:
TIMER := time():
to 10^6 do
p := expand(f*g):
end do:
time()-TIMER;
.

```