FACTORISATION OF RSA-704 WITH CADO-NFS

SHI BAI AND EMMANUEL THOMÉ AND PAUL ZIMMERMANN

Abstract. We give details of the factorization of RSA-704 with CADO-NFS. This is a record computation with publicly available software tools. The aim of this experiment was to stress CADO-NFS — which was originally designed for 512-bit factorizations — for larger inputs, and to identify possible rooms of improvement.

We report on the factorization of RSA-704 (212 decimal digits), which is the 2nd largest integer factorization with the General Number Field Sieve (GNFS), after the factorization of RSA-768 (232 digits) in December 2009 [3].

1. Polynomial Selection

Polynomial selection started end of April 2011. We used the following degree-6 polynomial:

$$\begin{aligned} f(x) &= 10614120x^6 \\ &+ 62813641710611x^5 \\ &+ 1938361239259842311964x^4 \\ &+ 93195711389054587511566471 5x^3 \\ &- 1118722849771428273314512798060648 3x^2 \\ &+ 2757913442475834957612632119277126344 50x \\ &+ 6316187855194115501570745234613072291012101 75 \end{aligned}$$

with 4 real roots, skewness 2159616.00, Murphy's $\alpha = -9.46$, and Murphy E-value $9.55 \cdot 10^{-16}$, together with the linear polynomial

$$g(x) = 17013143468292003100073935 99x - 10040119372014939875708192394943108.$$

This polynomial pair was found using the CADO-NFS implementation of Kleinjung's 2008 algorithm [2], and can be reproduced as follows:

```
$ ./polyselect2l -nq 800 -lq 7 -degree 6 -incr 60 -maxnorm 70 -admin 10614120
                 -admax 10614120 -N 740...359 -seed 1331320045 2000000
...
Y1: 17013143468292003100073935 99
Y0: -10040119372014939875708192394943108
c6: 10614120
c5: 62813641710611
c4: 1938361239259842311964
c3: 93195711389054587511566471 5
c2: -1118722849771428273314512798060648 3
c1: 2757913442475834957612632119277126344 50
c0: 6316187855194115501570745234613072291012101 75
```

Date: July 1st, 2012.

The command-line above says that we are searching the leading coefficient of the linear polynomial $g(x)$ of the form $g_1 = p_1 p_2 \cdots p_7 q_1 q_2$ with $P < q_1, q_2 < 2P$, $P = 2000000$, and $p_1, ..., p_7$ are seven small primes.

We also looked for degree-5 polynomials, but we prefered to use a degree-6 polynomial since CADO-NFS had been less used — if any at all — with such a degree. The total time spent in polynomial selection was about 12 core years (about 2.2 for degree 5, and 9.9 for degree 6), using about 100 cores at ANU.

2. Sieving

Sieving started on June 20, 2011, using clusters both at ANU and Inria Nancy-Grand Est. We used as factor base bound $250,000,000$ on the rational side, $500,000,000$ on the algebraic side, 2^{33} as large prime bound on both sides, with up to two large primes on the rational side and three on the algebraic side. Due to a limitation of the CADO-NFS lattice sieve, we had to use a sieving region of 2^{29} only ($I = 15$) which was sub-optimal ($I = 16$ like for the factorization of RSA-768 would have been better). This limitation has been removed now in CADO-NFS, but we prefered to finish the experiments with $I = 15$ (in addition, $I = 16$ uses a lot of memory, which would not enable us to use all cores of the clusters we had access to).

Due to the sub-optimal sieving parameters, we had to sieve over a large range of special-q, namely from 500M to 10000M. This in turn caused lots of duplicate relations (see below). On February 26, 2012 we had collected a total of 1,313,935,004 relations. We estimate to 500 CPU years the total sieving time (most of the computers used for sieving were Intel Xeons L5420 running at 2.5Ghz).

3. Filtering

The 1,313,935,004 raw relations gave 832,908,644 unique relations (i.e., 36.6% of duplicates). After removing singletons, we had a matrix with about 650M rows, and an initial excess of about 98M. After the "clique removal" algorithm, it remained about 320M rows and columns with an excess of 160, and an average weight of 22.19 per row.

In the merge phase, we merged ideals of weight up to 50. The CADO-NFS code does not consider the heavy ideals in the merge phase (here 132 heavy ideals were "buried"). The final matrix had about 89M rows and columns, with 17720843456 non-zero coefficients (about 200 per row), and 16568894691 in the sparse part (about 187 per row), while the dense part consists of the 32 heaviest ideals. (The size of the sparse matrix is 67Gb in binary form.)

After the merge phase, we tried the cycle optimization method from Denny and Müller [1], which is implemented in msieve too. We tried this method in two ways: firstly to reduce the number of *relations* in the matrix, secondly to reduce the number of *ideals*. In both cases we started with 319879050 relation-sets, and an initial weight of 1029799871 relations (11.63 on average per relation-set). If we use this method to reduce the number of *relations* in the merged matrix, we save only about 0.03% of the total number of relations, which in turn reduced by about 0.02% the total weight of the matrix. If we use this method to reduce the number of *ideals* (instead of relations) in the merged matrix, we gain only a little more in the total weight of the matrix. We conclude that the cycle optimization method from Denny and Müller is not interesting with modern merge algorithm using Markowitz pivoting.

4. Linear Algebra

The linear algebra was started on March 19, 2012. The block Wiedemann algorithm has been used, using only one sequence of 64-bit vectors (in other words, blocking parameters $m = n = 64$ have been used for the algorithm). The first phase (`krylov`) ended on May 11. The second phase (`lingen`) started on May 13, although the full run, taking 10 days, completed only on June 12, due to successive power outages. The last phase (`mksol`) started on June 13 and ended on June 29, 2012, producing a kernel of dimension 61.

All `krylov` and `mksol` jobs within this project were carried on the french Grid'5000 computer grid. The article [4] indicates how the block Wiedemann algorithm may be split into small jobs in order to be used on a grid. Overall, 1800h of wall-clock time have been spent on these `krylov` and `mksol` tasks, on clusters of varying sizes (clusters have been used in an opportunistic manner, depending on resource availability). Noteworthy is the fact that all of these jobs have been setup as "best-effort" jobs, allowing any competing job submission to immediately kill our job, without any notice. This setup has obviously led to jobs spending time in their start-up phase (importing data to the cluster nodes), yet not being able to do much useful work later on. The overhead cost of this sometimes useless I/O time on the previously indicated timing is roughly 800h. This large overhead could have been lowered considerably if we had been able to replicate our storage points on several nodes, as we did previously for the RSA-768 computation.

5. Characters and Square Root

The `characters` program gave 10 dependencies. We had 64 dependencies from the linear algebra, minus 32 since we skipped the 32 heaviest columns, thus 22 dependencies were "killed" by the characters, which is larger than the usual value of 10-15; we have to investigate why.

6. Conclusion

The factorization of RSA-704 is $n = p \cdot q$ where

$$\begin{aligned} p &= 90912135295978188784406583026004374858926083103283587204285121689 \\ &\quad 60411528640933367824950788367956756806141 \\ q &= 81438592591100452657278091262844293358778990021676278832009141724 \\ &\quad 29324360133004116702003240828777970252499 \end{aligned}$$

have both 106 digits.

The prime factors of $p \pm 1$ and $q \pm 1$ are:

$$\begin{aligned} p-1 &= 2^2 \cdot 5 \cdot 17 \cdot 7759 \cdot 248701 \cdot 3311937667 \cdot 1669783862489 \cdot \\ &\quad 188045064464200049383844 9 \cdot p49 \\ p+1 &= 2 \cdot 3^2 \cdot 43 \cdot 71 \cdot 157 \cdot 2630713 \cdot 1850017111 \cdot 1040072485315298476627 \cdot \\ &\quad 202390973793150189326984 5781 \cdot p36 \\ q-1 &= 2 \cdot 19 \cdot 149 \cdot 233 \cdot 426163 \cdot 34302641 \cdot 415283201 \cdot p79 \\ q+1 &= 2^2 \cdot 3^2 \cdot 5^4 \cdot 8753 \cdot 27539 \cdot 962945197 \cdot p85 \end{aligned}$$

Acknowledgements. The computation was done with CADO-NFS, a free software whose main developpers are Pierrick Gaudry, Alexander Kruppa, François Morain and the three authors (see `http://cado-nfs.gforge.inria.fr/`). Lionel Muller helped distributing jobs in the sieving phase. The `orac` cluster in the Mathematical Sciences Institute of the Australian National University was used for polynomial selection and sieving. The TALC cluster of Inria Nancy-Grand Est was used for the sieving phase, and Grid 5000 was used for the linear algebra phase. Grid'5000 is an experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see `https://www.grid5000.fr`).

References

[1] Denny, T. F., and Müller, V. On the reduction of composed relations from the number field sieve. Extended Abstract, 1995. 10 pages.

[2] Kleinjung, T. Polynomial selection. slides presented at the CADO workshop on integer factorization, 2008.

[3] Kleinjung, T., Aoki, K., Franke, J., Lenstra, A. K., Thomé, E., Bos, J. W., Gaudry, P., Kruppa, A., Montgomery, P. L., Osvik, D. A., te Riele, H., Timofeev, A., and Zimmermann, P. Factorization of a 768-bit rsa modulus. In *CRYPTO 2010 Advances in Cryptology - CRYPTO 2010* (Santa Barbara, USA, 2010), T. Rabin, Ed., vol. 6223 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 333–350.

[4] Kleinjung, T., Nussbaum, L., and Thomé, E. Using a grid platform for solving large sparse linear systems over GF(2). In *11th ACM/IEEE International Conference on Grid Computing (Grid 2010)* (2010), IEEE, pp. 161–168.