CADO-NFS

Crible Algébrique: Distribution, Optimisation - Number Field Sieve

Home
Prerequisites
Download
Development
Bugs / Support
User Reports

Introduction

CADO-NFS is a complete implementation in C/C++ of the Number Field Sieve (NFS) algorithm for factoring integers and computing discrete logarithms in finite fields. It consists in various programs corresponding to all the phases of the algorithm, and a general script that runs them, possibly in parallel over a network of computers. CADO-NFS is distributed under the Gnu Lesser General Public License (LGPL) version 2.1 (or any later version).

CADO-NFS is the result of a collaborative effort involving many persons, over various periods of time. The current list of active contributors can be extracted from the git repository or from the openhub.net page. A tentative list of CADO-NFS authors is (alphabetical order):

- Shi Bai
- Razvan Barbulescu
- Cyril Bouvier
- Richard Brent
- Christophe Clavier
- Jérémie Detrey
- Andreas Enge
- Alain Filbois
- Nuno Franco
- Pierrick Gaudry
- Laurent Grémy
- Aurore Guillevic
- Nadia Heninger
- Laurent Imbert
- Alexander Kruppa
- Jérome Milan
- François Morain
- Lionel Muller
- Thomas Prest
- Thomas Richard
- Emmanuel Thomé
- Marion Videau
- Paul Zimmermann

Citing CADO-NFS

The recommended way to cite CADO-NFS in a scientific publication is (bibtex entry):

- `The CADO-NFS Development Team.` *`CADO-NFS, An Implementation of the Number Field Sieve Algorithm`*`, Release 2.3.0, 2017, http://cado-nfs.gforge.inria.fr/`

where the release number and date should be changed to correspond to the version you actually used. If you used the development version, it is a good idea to give the git revision number. You can, at the very least, say "development version" and give the current date. You can base your citation entry on the following template (bibtex entry for the development version, to be completed):

- `The CADO-NFS Development Team. `*`CADO-NFS, An Implementation of the Number Field Sieve Algorithm`*`, development version, 20XX, http://cado-nfs.gforge.inria.fr/`

Features

Algorithms used in CADO-NFS 2.3.0 are the following:

- The polynomial selection uses the algorithm of Kleinjung (2008) then candidate polynomials are optimized using the algorithm of Bai, Bouvier, Kruppa and Zimmermann (2014)
- The filtering step follows Cavallar's thesis and Bouvier's work. Right now it is partly parallel.
- Relation search is done using lattice sieving, including multithread support to reduce memory.
- The linear algebra step is implemented using block Wiedemann algorithm. This implementation is parallel at multithread and MPI levels.
- The square root step uses the naive algorithm and is parallel. An alternate (experimental) implementation is available for very large computations, or pathological Galois groups.

Speed comparison of CADO-NFS versions 1.1 and 2.0, obtained on a dual 8-core Intel(R) Xeon(R) CPU E5-2650 at 2.00GHz, running Linux 3.2.0. Both CADO releases were compiled with gcc 4.7.2.
With CADO-NFS 1.1 using cadofactor.pl, the timing runs used 16 processes of 1 thread each for polynomial selection and 8 processes of 2 threads each for sieving; with CADO-NFS 2.0 using cadofactor.py, both phases used 8 processes of two threads each, with CADO-NFS 2.1 using `./factor.sh N -s 8 -t 2 tasks.linalg.bwc.threads=16`. With CADO-NFS 2.3, the same behavior is obtained with `./cado-nfs.py --client-threads 2 --server-threads 16 --slaves 8`. The table below lists CPU time [wall clock time in square brackets].

Input number	**CADO-NFS 1.1**	**CADO-NFS 2.0**	**CADO-NFS 2.1**	**CADO-NFS 2.2.0**	**CADO-NFS 2.3.0**
RSA-120	100 hours	45.6 hours	45.2 hours [3.6 hours]	32.2 hours [2.2 hours]	26 hours [1.9 hours]
RSA-130	288 hours	231 hours	219 hours [15.9 hours]	124 hours [8.2 hours]	107 hours [7.5 hours]
RSA-140	809 hours	614 hours	643 hours [45.7 hours]	469 hours [30.9 hours]	352 hours [23 hours]
RSA-155	268 days	141 days	126 days [12.2 days]	90 days [5.8 days]	83 days [5.3 days]

For 85- to 100-digits, Ben Buhrow compared various QS and NFS tools, see here.

You can browse into the development tree history to find the NEWS file for past releases:

- NEWS for 2.3 branch

- NEWS for 2.2 branch
- NEWS for 2.1 branch
- NEWS for 2.0 branch
- NEWS for 1.1 branch

Last modification: Wed 10 Jun 2020 03:48:46 PM CEST