make test failure: d1mach - i out of bounds

Loris Bennett-2 Jul 12, 2010; **make test failure: d1mach - i out** Reply | Threaded | More

13 posts

Hi all,

Having successfully compiled Octave 3.2.4 with --enable-64, make check
fails with

```
../run-octave --norc --silent --no-history ./fntests.m .
 d1mach - i out of bounds         0
gmake[1]: *** [check] Segmentation fault (core dumped)
```

Does anyone have any pointers to the cause of the problem?

Thanks

Loris

--
Dr. Loris Bennett
ZEDAT Computer Centre
Freie Universität Berlin
Berlin, Germany

_____
Help-octave mailing list
[hidden email]
https://www-old.cae.wisc.edu/mailman/listinfo/help-octave

Dmitri A. Sergatsk Jul 12, 2010; **Re: make test failure: d1mach - i ou** Reply | Threaded | More

711 posts

On Mon, Jul 12, 2010 at 9:55 AM, Loris Bennett
<[hidden email]> wrote:

> Hi all,
>
> Having successfully compiled Octave 3.2.4 with --enable-64, make check
> fails with
>
> ../run-octave --norc --silent --no-history ./fntests.m .
> d1mach - i out of bounds 0
> gmake[1]: *** [check] Segmentation fault (core dumped)
>
... [show rest of quote]

a) If you want to get a meaningful answer you really need to
provide a whole lot more details.

b) --enable-64 is very experimental and you should _really_ know what you
are doing.

Regards,

Dmitri.
--

_____
Help-octave mailing list
[hidden email]
https://www-old.cae.wisc.edu/mailman/listinfo/help-octave

John W. Eaton Jul 12, 2010; **make test failure: d1mach - i out of** Reply | Threaded | More

Administrator
4235 posts

In reply to this post by Loris Bennett-2

On 12-Jul-2010, Loris Bennett wrote:

| Hi all,
|
| Having successfully compiled Octave 3.2.4 with --enable-64, make check
| fails with
|
| ../run-octave --norc --silent --no-history ./fntests.m .
| d1mach - i out of bounds 0
| gmake[1]: *** [check] Segmentation fault (core dumped)
|
| Does anyone have any pointers to the cause of the problem?

Did you ensure that any Fortran code (or functions called using the F77_FUNC interface) was compiled so that INTEGER values are 8 bytes, not 4?

If you are interested in using the experimental --enable-64 option, I recommend using the development sources of Octave from the Mercurial archive as there have been a number of changes since 3.2.

I'm attaching some notes that I have about building the development version and dependencies with --enable-64 that I intend to add to the Octave manual.

Given that Octave can handle arrays with 2^31-1 elements on 64-bit systems without having to use --enable-64, are you sure you need --enable-64? 2^31-1 elements is about 16GB per array for double precision arrays.

jwe

Note: the following only applies to systems that have 64-bit pointers. Configuring Octave with --enable-64 cannot magically make a 32-bit system have a 64-bit address space.

On 64-bit systems, Octave is limited to (approximately) the following array sizes:

```
double:    16GB
single:     8GB
{u,}int64: 16GB
{u,}int32:  8GB
{u,}int16:  4GB
{u,}int8:   2GB
```

In each case, the limit is really 2^31-1 elements because of the default type of the value used for indexing arrays (signed 32-bit integer, corresponding to the size of a Fortran INTEGER value).

Trying to create larger arrays will produce the following error:

octave:1> a = zeros (1024*1024*1024*3, 1, 'int8');
error: memory exhausted or requested size too large for range of Octave's index type -- trying to return to prompt

You will obtain this error even if your system has enough memory to create this array (4 GB in the above case).

To use arrays larger than 2 GB, Octave has to be configured with the option --enable-64. This option is experimental and you are (as always) encouraged to submit bug reports if you find a problem. With this option, Octave will use 64-bit integers internally for array dimensions and indexing. However, all numerical libraries used by Octave will also need to use 64-bit integers for array dimensions and indexing. In most cases, this means they will need to be compiled from source since most (all?) distributions which package these libraries compile them with the default Fortran integer size.

The following instructions were tested with the development version of Octave and GCC 4.3.4 on an x86_64 Debian system.

The versions listed below are the versions used for testing. If newer versions of these packages are available, you should try to use them, though there may be some differences.
nnn
All libraries and header files will be installed in subdirectories of $prefix64 (you must choose the location of this directory).

BLAS and LAPACK (http://www.netlib.org/lapack)

Reference versions for both libraries are included in the reference LAPACK 3.2.1 distribution from netlib.org.

Copy the file make.inc.example and name it make.inc. The options -fdefault-integer-8 and -fPIC (on 64-bit CPU) have to be added to the variable OPTS and NOOPT.

Once you have compiled this library make sure that you use it for compiling Suite Sparse and Octave. In the following we assume that you installed the LAPACK library as $prefix64/lib/liblapack.a.

ARPACK (http://www.caam.rice.edu/software/ARPACK)

In ARmake.inc set the following variables:

```
home=path to directory ARPACK
FC=gfortran
FFLAGS=-fPIC -fdefault-integer-8
MAKE=/usr/bin/make
ARPACKLIB=$(home)/libarpack.a
DIRS=$(UTILdir) $(SRCdir)
```

Edit the file UTIL/second.f and change the line

```
EXTERNAL          ETIME
```

to

```
INTRINSIC         ETIME
```

After building .a library, you can make a shared version with

```
mkdir tmp
cd tmp
```

```
ar x ../libarpack.a
gcc -shared -o ../libarpack.so *.o -L$prefix64/lib -llapack -lblas
cd ..
rm -rf tmp
```

Copy the library libarpack.so to $prefix64/lib/libarpack.a.

QRUPDATE (http://sourceforge.net/projects/qrupdate)

In the Makeconf file:

Add -fdefault-integer-8 to FFLAGS.

Adjust the BLAS and LAPACK variables as needed if your 64-bit aware BLAS and LAPACK libraries are in a non-standard location.

Set PREFIX to the top-level directory of your install tree.

Run make solib to make a shared library.

Run make install to install the library.

SUITESPARSE (http://www.cise.ufl.edu/research/sparse/SuiteSparse)

In UFconfig/UFconfig.mk use the following options for CFLAGS and F77FLAGS:

```
CC = gcc
CFLAGS = -fPIC -O -DLP64 -DLONGBLAS='long int' -DLONG='long int'
F77 = gfortran
F77FLAGS = -fPIC -O -fdefault-integer-8
BLAS = -L$BLAS/lib -lblas -lgfortran"
LAPACK = -L$LAPACK/lib -llapack"
```

Disable the GPL-incompatible METIS library:

```
CHOLMOD_CONFIG = -DNPARTITION
SPQR_CONFIG = -DNPARTITION
METIS_PATH =
METIS =
```

Disable the DI versions of the CHOLMOD library files by setting

```
OBJ = $(DL)
```

in CHOLMOD/Lib/Makefile.

Disable the DI versions of the CHOLMOD tests by commenting out or deleting the following lines in CHOLMOD/Demo/Makefile:

```
./cholmod_demo < Matrix/bcsstk01.tri
./cholmod_demo < Matrix/lp_afiro.tri
./cholmod_demo < Matrix/can___24.mtx
./cholmod_demo < Matrix/c.tri
./cholmod_simple < Matrix/c.tri
./cholmod_simple < Matrix/can___24.mtx
./cholmod_simple < Matrix/bcsstk01.tri
```

Run make to build the libraries.

The SuiteSparse Makefile does not have an install target so you must install the files by hand:

```
   cp {AMD,BTF,CAMD,CCOLAMD,CHOLMOD,COLAMD,CXSparse,UMFPACK}/Lib/lib*a
$prefix64/lib
   mkdir $prefix64/include/suitesparse
   cp {AMD,BTF,CAMD,CCOLAMD,CHOLMOD,COLAMD,CXSparse,UMFPACK}/Include/*h
UFconfig/UFconfig.h $prefix64/include/suitesparse

  You can generate shared versions of these libraries by doing the
  following in the $prefix64/lib directory:

   top=$(pwd)
   for f in *.a; do
     mkdir tmp
     cd tmp
     ar vx ../$f
     gcc -shared -o ../${f%%.a}.so *.o
     cd $top
     rm -rf tmp
   done
```

QHULL
http://www.qhull.org

Suggestions on how to compile qhull will be most welcome.

Octave

Octave's 64-bit index support is activated with the configure option
--enable-64.

```
  ./configure \
    CPPFLAGS="-I$SUITESPARSE/include -I$QHULL/include " \
    LIBS="-L$SUITESPARSE/lib -lmetis -L$QHULL/lib  -L$ARPACK -larpack -L$LAPACK/lib -
L$BLAS/lib -lblas -lgfortran -lm -lpthread" \
    FFLAGS=-fdefault-integer-8 \
    F77=gfortran --enable-64
```

Remaining Dependencies:

qhull
glpk

atlas instead of reference blas and lapack

Probably nothing special needs to be done for the following:

```
  pcre or regex
  zlib
  hdf5
  fftw3
  cURL
  GraphicsMagick++
  OpenGL
  freetype
  fontconfig
  fltk
```

_____

Help-octave mailing list
[hidden email]
https://www-old.cae.wisc.edu/mailman/listinfo/help-octave

Loris Bennett-2 Jul 13, 2010; **Re: make test failure: d1mach - i o** Reply | Threaded | More

13 posts

In reply to this post by Dmitri A. Sergatskov

"Dmitri A. Sergatskov" <[hidden email]> writes:

> On Mon, Jul 12, 2010 at 9:55 AM, Loris Bennett
> <[hidden email]> wrote:
>> Hi all,
>>
>> Having successfully compiled Octave 3.2.4 with --enable-64, make check
>> fails with
>>
>> ../run-octave --norc --silent --no-history ./fntests.m .
>> d1mach - i out of bounds 0
... [show rest of quote]

The error that a user was getting with a 32-bit version was:

```
error: memory exhausted or requested size too large for range of
Octave's index type
```

As there was no indication that the memory really was exhausted, I
assumed it must be the latter cause, which lead me to the wiki page

http://wiki.octave.org/wiki.pl?EnableLargeArrays

Do I just need a 64-bit version without --enable-64?

However, as I am going to recompile Octave from the development sources,
it is probably not worth worring about the segentation fault above
unless it reoccurs.

Cheers

Loris

--
Dr. Loris Bennett
ZEDAT Computer Centre
Freie Universität Berlin
Berlin, Germany

_____
Help-octave mailing list
[hidden email]
https://www-old.cae.wisc.edu/mailman/listinfo/help-octave

Loris Bennett-2 Jul 14, 2010; **Re: make test failure: d1mach - i ou** Reply | Threaded | More

13 posts

In reply to this post by John W. Eaton

"John W. Eaton" <[hidden email]> writes:

> On 12-Jul-2010, Loris Bennett wrote:
>
> | Hi all,
> |
> | Having successfully compiled Octave 3.2.4 with --enable-64, make check
> | fails with

> |
> | ../run-octave --norc --silent --no-history ./fntests.m .
> | d1mach - i out of bounds 0
... [show rest of quote]

I suspect that this is the cause of the problem and that I would indeed need to recompile lapack with 8 byte ints.

> If you are interested in using the experimental --enable-64 option, I
> recommend using the development sources of Octave from the Mercurial
> archive as there have been a number of changes since 3.2.
>
> I'm attaching some notes that I have about building the development
> version and dependencies with --enable-64 that I intend to add to the
> Octave manual.
>
> Given that Octave can handle arrays with 2^31-1 elements on 64-bit
... [show rest of quote]

The user who was having the original problem with "memory exhausted / size too large for index type" has reported that it occured with a matrix with only 40^4 rows und 4 columns, so it seems like the limit on the array index is not the problem.

The problem appears to be that AIX by default restricts the number of 256MB data segments a process can use for data to just 1. If the process needs more, this can be achieved by setting an environent variable:

LDR_CNTRL=MAXDATA=0xn0000000

where n is the number of segments required.

I shall check whether setting this really does solve the user's problem.

Cheers

Loris

> jwe
>
> Note: the following only applies to systems that have 64-bit
> pointers. Configuring Octave with --enable-64 cannot magically make
> a 32-bit system have a 64-bit address space.
>
> On 64-bit systems, Octave is limited to (approximately) the following
> array sizes:
>
... [show rest of quote]

--
Dr. Loris Bennett
ZEDAT Computer Centre
Freie Universität Berlin
Berlin, Germany

« Return to Octave - General | 371 views