Docker build from Dockerfile with more memory

Asked 5 years, 6 months ago Modified 1 year, 10 months ago Viewed 34k times

29

How to `docker build` from Dockerfile with more memory?

This is a different question from this [Allow more memory when docker build a Dockerfile]

When installing the software natively, there is enough memory to successfully build and install the [`marian` tool]

But when building the Docker image using the `Dockerfile` https://github.com/marian-nmt/marian/blob/master/scripts/docker/Dockerfile.cpu , it fails with multiple memory exhausted errors

```
virtual memory exhausted: Cannot allocate memory
```

[out]:

```
Step : RUN cmake $MARIANPATH && make -j
 ---> Running in 4867d166d17a
-- The C compiler identification is GNU 5.4.0
-- The CXX compiler identification is GNU 5.4.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
CUDA_TOOLKIT_ROOT_DIR not found or specified
-- Cannot find CUDA libraries. Compiling without them.
-- Could NOT find CUDA (missing:  CUDA_TOOLKIT_ROOT_DIR CUDA_NVCC_EXECUTABLE
CUDA_INCLUDE_DIRS CUDA_CUDART_LIBRARY)
-- Looking for pthread.h
-- Looking for pthread.h - found
-- Looking for pthread_create
-- Looking for pthread_create - not found
-- Looking for pthread_create in pthreads
-- Looking for pthread_create in pthreads - not found
-- Looking for pthread_create in pthread
-- Looking for pthread_create in pthread - found
-- Found Threads: TRUE
-- Boost version: 1.58.0
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up

```
--   timer
--   iostreams
--   python
     thread
```

And eventually it leads to:

```
src/amun/CMakeFiles/libcommon.dir/build.make:254: recipe for target
'src/amun/CMakeFiles/libcommon.dir/common/loader.cpp.o' failed
virtual memory exhausted: Cannot allocate memory
make[2]: *** [src/amun/CMakeFiles/cpumode.dir/cpu/decoder/encoder_decoder.cpp.o] Error 1
src/amun/CMakeFiles/cpumode.dir/build.make:110: recipe for target
'src/amun/CMakeFiles/cpumode.dir/cpu/decoder/encoder_decoder.cpp.o' failed
[ 79%] Built target libcnpy
virtual memory exhausted: Cannot allocate memory
src/amun/CMakeFiles/libcommon.dir/build.make:326: recipe for target
'src/amun/CMakeFiles/libcommon.dir/common/printer.cpp.o' failed
make[2]: *** [src/amun/CMakeFiles/libcommon.dir/common/printer.cpp.o] Error 1
CMakeFiles/Makefile2:340: recipe for target 'src/amun/3rd_party/yaml-
cpp/CMakeFiles/libyaml-cpp-amun.dir/all' failed
make[1]: *** [src/amun/3rd_party/yaml-cpp/CMakeFiles/libyaml-cpp-amun.dir/all] Error 2
CMakeFiles/Makefile2:182: recipe for target 'src/amun/CMakeFiles/libcommon.dir/all'
failed
make[1]: *** [src/amun/CMakeFiles/libcommon.dir/all] Error 2
make[1]: *** [src/amun/CMakeFiles/cpumode.dir/all] Error 2
CMakeFiles/Makefile2:110: recipe for target 'src/amun/CMakeFiles/cpumode.dir/all' failed
make: *** [all] Error 2
```

Looking at the `docker build --help`, there are options to control memory usage:

```
$ docker build --help

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile

Options:
      --build-arg list             Set build-time variables (default [])
      --cache-from stringSlice     Images to consider as cache sources
      --cgroup-parent string       Optional parent cgroup for the container
      --compress                   Compress the build context using gzip
      --cpu-period int             Limit the CPU CFS (Completely Fair Scheduler) period
      --cpu-quota int              Limit the CPU CFS (Completely Fair Scheduler) quota
  -c, --cpu-shares int             CPU shares (relative weight)
      --cpuset-cpus string         CPUs in which to allow execution (0-3, 0,1)
      --cpuset-mems string         MEMs in which to allow execution (0-3, 0,1)
      --disable-content-trust      Skip image verification (default true)
  -f, --file string                Name of the Dockerfile (Default is
'PATH/Dockerfile')
      --force-rm                   Always remove intermediate containers
      --help                       Print usage
      --isolation string           Container isolation technology
      --label list                 Set metadata for an image (default [])
  -m, --memory string              Memory limit
      --memory-swap string         Swap limit equal to memory plus swap: '-1' to enable
unlimited swap
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up ✕

```
      --pull                          Always attempt to pull a newer version of the image
  -q, --quiet                         Suppress the build output and print image ID on
success
      --rm                            Remove intermediate containers after a successful
build (default true)
      --security-opt stringSlice      Security options
      --shm-size string               Size of /dev/shm, default value is 64MB
  -t, --tag list                      Name and optionally a tag in the 'name:tag' format
```

But I couldn't figure out the correct syntax of where exactly to put the `-m` option -_-|||

It isn't before the Dockerfile:

```
# Before Docker file.
$ docker build -m 4g Dockerfile.cpu -t ibot-cpu .

"docker build" requires exactly 1 argument(s).
See 'docker build --help'.

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile
```

It isn't after Dockerfile before -t

```
# Before -t
$ docker build Dockerfile.cpu -m 4g -t ibot-cpu .

"docker build" requires exactly 1 argument(s).
See 'docker build --help'.

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile
```

It isn't after -t before the local path

```
# Before local path
$ docker build Dockerfile.cpu -t ibot-cpu -m 4g .

"docker build" requires exactly 1 argument(s).
See 'docker build --help'.

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile
```

It isn't after the local path at the end too...

```
# At the end...
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up ✕

```
Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile
```

How to `docker build` from Dockerfile with more memory?

My docker version:

```
docker version
Client:
 Version:      17.03.1-ce
 API version:  1.27
 Go version:   go1.7.5
 Git commit:   c6d412e
 Built:        Tue Mar 28 00:40:02 2017
 OS/Arch:      darwin/amd64

Server:
 Version:      17.04.0-ce
 API version:  1.28 (minimum version 1.12)
 Go version:   go1.7.5
 Git commit:   4845c56
 Built:        Wed Apr  5 18:45:47 2017
 OS/Arch:      linux/amd64
 Experimental: false
```

docker memory docker-build neural-mt

Share Improve this question Follow

asked Jul 28, 2017 at 2:10

alvas
111k 106 434 713

1 Answer

Sorted by:

Highest score (default)

54

It is not something about order. The Dockerfile must be specified with `-f`

```
docker build -f Dockerfile.cpu -t ibot-cpu -m 4g .
```

However, take into account that by default docker does not limit the container memory. It can take the whole free memory.

As I can see that you are on OSX, which runs docker over a Linux VM. Configure the max memory

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up

For further information please see my other answer: How to assign more memory to docker container

Share Improve this answer Follow

edited Apr 23, 2021 at 16:01

answered Jul 31, 2017 at 11:12

Robert

32k 8 86 93

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up ✕