t5memory backend

t5memory was ported to work with Ubuntu 20.04 LTS. It uses Proxygen library(HTTP lib, supported by Facebook), which uses other dependencies, like

- would be pulled and built automatically during proxygen build: **fizz, fmt, folly, googletest,, wangle, zstd**
- should be installed in the system for proxygen build**: glog, gflags , boost, OpenSSL**
- libs, that are not related to proxygen: **xercesc, ICU**

For configuration, **t5memory** uses **gflags** and command line arguments. Configuration file(**~/.t5memory/t5memory.conf**) is disabled by default, but you can enable it from the command line(but probably there is no reason to use it) since **gflags** functionality is much more profound.

To get list of flags just call **t5memory --help** ,

To use any flag just add two dashes before, like **--v=2**

Here is the list of flags:

Most useful flags

	name	description	type	default value	limits
1	allowedram	Sets amount RAM(in MB) allowed for service to use	int 32	2048	[1... 10000]
2	t5_ip	Which ip to use in t5memory(default is any). Should be in format '1.1.1.1'. By default t5memory is listening all available ips	stri ng	""	
3	port	What port to listen on	int 32	4080	[0... 32768]
4	servicename	Sets service name to use in URL	stri ng	t5memory	
5	servicethreads	Sets amount of worker threads for service(for proxygen)	int 32	1	[1...100]
6	t5loglevel	Sets t5memory log level threshold from DEVELOP(0) to TRANSACTION(6)	int 32	2(T5INFO)	[0...6]
7	triplesthreshold	Sets threshold to pre-fuzzy filtering based on hashes of neighbor tokens	int 32	33	[0... 100]
8	timeout	Sets the timeout for service request handling in ms.	int 32	180000	[0... 360000]
9	~~useconfigfile~~	~~Set to use values from the config file that should be located under ~/.t5memory/t5memory.conf~~	~~bo ol~~	~~false~~	
10	v	Show all VLOG(m) messages for m <= this. Overridable by --vmodule. This should be used to enable DEBUG (v=1) and DEVELOP(v=2) logs. If set to v=1, all logs, starting from DEBUG(1) level, would be added to the buffer of logs(instead of only request data), which would be printed in case of an error. If set to v=2, all logs would ignore the buffer and would be printed, starting from level=0(develop logs)	int 32	0	[0...2]
11	alsologtostderr	log messages go to stderr in addition to logfiles	bo ol	false	
12	logtostderr	log messages go to stderr instead of logfiles	bo ol	false	
13	log_dir	If specified, logfiles are written into this directory instead of the default logging directory.	stri ng	~/.t5memory /LOG/	
14	stderrthreshold	log messages at or above this level are copied to stderr in addition to logfiles. This flag obsoletes --alsologtostderr.	int 32	2	
15	max_log_size	approx. maximum log file size (in MB). A value of 0 will be silently overridden to 1.	int 32	1800	

16	minloglevel	Messages logged at a lower level than this don't actually get logged anywhere	int 32	0	
17	stop_logging_if_f ull_disk	Stop attempting to log to disk if the disk is full	bo ol	false	
18	drop_log_memor y	Drop in-memory buffers of log contents. Logs can grow very quickly and they are rarely read before they need to be evicted from memory. Instead, drop them from memory as soon as they are flushed to disk.	bo ol	true	

Other flags

Flags from /build/gflags-0sowem/gflags-2.2.2/src/gflags.cc:
  -flagfile (load flags from file) type: string default: ""
  -fromenv (set flags from the environment [use 'export FLAGS_flag1=value'])
    type: string default: ""
  -tryfromenv (set flags from the environment if present) type: string
    default: ""
  -undefok (comma-separated list of flag names that it is okay to specify on
    the command line even if the program does not define a flag with that
    name. IMPORTANT: flags in this list that have arguments MUST use the
    flag=value format) type: string default: ""

Flags from /build/gflags-0sowem/gflags-2.2.2/src/gflags_completions.cc:
  -tab_completion_columns (Number of columns to use in output for tab
    completion) type: int32 default: 80
  -tab_completion_word (If non-empty, HandleCommandLineCompletions() will
    hijack the process and attempt to do bash-style command line flag
    completion on this value.) type: string default: ""

Flags from /build/gflags-0sowem/gflags-2.2.2/src/gflags_reporting.cc:
  -help (show help on all flags [tip: all flags can have two dashes])
    type: bool default: false currently: true
  -helpfull (show help on all flags -- same as -help) type: bool
    default: false
  -helpmatch (show help on modules whose name contains the specified substr)
    type: string default: ""
  -helpon (show help on the modules named by this flag value) type: string
    default: ""
  -helppackage (show help on all modules in the main package) type: bool
    default: false
  -helpshort (show help on only the main module for this program) type: bool
    default: false
  -helpxml (produce an xml version of help) type: bool default: false
  -version (show version and build info and exit) type: bool default: false

 Flags from /home/libs/proxygen/_build/deps/folly/folly/detail/MemoryIdler.cpp:
  -folly_memory_idler_purge_arenas (if enabled, folly memory-idler purges
    jemalloc arenas on thread idle) type: bool default: true

 Flags from /home/libs/proxygen/_build/deps/folly/folly/executors/IOThreadPoolExecutor.cpp:
  -dynamic_iothreadpoolexecutor (IOThreadPoolExecutor will dynamically create
    threads) type: bool default: true

 Flags from /home/libs/proxygen/_build/deps/folly/folly/executors/ThreadPoolExecutor.cpp:
  -threadtimeout_ms (Idle time before ThreadPoolExecutor threads are joined)
    type: int64 default: 60000

 Flags from /home/libs/proxygen/_build/deps/folly/folly/experimental/observer/detail/ObserverManager.cpp:
  -observer_manager_pool_size (How many internal threads ObserverManager
    should use) type: int32 default: 4

 Flags from /home/libs/proxygen/_build/deps/folly/folly/synchronization/Hazptr.cpp:
  -folly_hazptr_use_executor (Use an executor for hazptr asynchronous
    reclamation) type: bool default: true

Flags from /home/libs/proxygen/_build/deps/wangle/wangle/ssl/SSLSessionCacheManager.cpp:
  -dcache_unit_test (All VIPs share one session cache) type: bool
    default: false

Flags from /home/libs/proxygen/lib/utils/ZlibStreamCompressor.cpp:
  -zlib_compressor_buffer_growth (The buffer growth size to use during IOBuf
    zlib deflation) type: int64 default: 2024

Flags from /home/or/workspace/translate5/translate5-tm-service-source/source/RestAPI/ProxygenHandler.cpp:
  -request_number (Include request sequence number in response) type: bool
    default: true

Flags from src/logging.cc:
  -alsologtoemail (log messages go to these email addresses in addition to
    logfiles) type: string default: ""
  -colorlogtostderr (color messages logged to stderr (if supported by
    terminal)) type: bool default: false currently: true
  -log_backtrace_at (Emit a backtrace when logging at file:linenum.)
    type: string default: ""
  -log_link (Put additional links to the log files in this directory)
    type: string default: ""
  -log_prefix (Prepend the log prefix to the start of each log line)
    type: bool default: true
  -logbuflevel (Buffer log messages logged at this level or lower (-1 means
    don't buffer; 0 means buffer INFO only; ...)) type: int32 default: 0
  -logbufsecs (Buffer log messages for at most this many seconds) type: int32
    default: 30
  -logemaillevel (Email log messages logged at this level or higher (0 means
    email all; 3 means email FATAL only; ...)) type: int32 default: 999
  -logfile_mode (Log file mode/permissions.) type: int32 default: 436
  -logmailer (Mailer used to send logging email) type: string
    default: "/bin/mail"

Flags from src/utilities.cc:
  -symbolize_stacktrace (Symbolize the stack trace in the tombstone)
    type: bool default: true

Flags from src/vlog_is_on.cc:

  -vmodule (per-module verbose level. Argument is a comma-separated list of
    <module name>=<log level>. <module name> is a glob pattern, matched
    against the filename base (that is, name ignoring .cc/.h./-inl.h). <log
    level> overrides any value given by --v.) type: string default: ""

Logs

For logging t5memory uses glog library. t5memory has it's own logging levels (DEVELOP=0, DEBUG=1, INFO=2, WARNIGN=3, ERROR=4, FATAL=5, TRANSACTION=6), that maps to 4 levels of glog
t5memory log glog log
DEVELOP = INFO && VLOG>=2

DEBUG = INFO && VLOG>=1

INFO = INFO

WARNING=WARNIGN

FATAL = DFATAL(regular FATAL would crash app immediately)

TRANSACTION = INFO

You can manipulate logs with flags.
By default t5memory wouldn't log to stderr, to change that use --logtostderr, or, if you want logs in both strerr and files, then use --alsologtostderr

You can set log behaviour with --v=1 or --v=2 flag.

By default t5memory logs every log with level=INFO during initialization and then switches to state when it logs only errors and requests data, that caused errors. It saves request data(URL and body if it has) and add it once if there would appear any error. Then it flushes itself

  and with next error from the same request there would be no duplicates of info about request.

If you would set --v=1, then it would save all other logs, starting from DEBUG, into the buffer, but would print it only if there would be an error.

If you would set --v=2, then it would omit buffer and print logs directly, starting from DEVELOP.
You can disable some logs with --minloglevel. Also, if that makes sense, old logging threshold could be implemented to set some bevahiour of --v=1 or --v=2, but to filter some DEBUG and DEVELOP logs.

Build process

We are building t5memory in docker containers. To build t5memory you should build container with proxygen first. This would require a lot of resources (you should have at least 16 GB RAM and image would take 20 GB)

To build proxygen container you should go to folder where Dockerfile_proxygen is located(from 'translate5-tm-service-source/docker', but you need only dockerfile to build container) and run this command:

docker build -t translate5/proxygen -f Dockerfile_proxygen .

and then wait.
Then you can build t5memory from folder where Dockerfile(from 'translate5-tm-service-source/docker/') with command

docker build -t translate5/t5memory .
This would pull t5memory repository from github and would build t5memory from last commit
Alternatively you can call the same command from the project's root directory, this would build t5memory from local files instead of repo, which could be used for development.

Run a single docker container with a specific version

To run t5memory as a single docker container

```
docker run -d --restart unless-stopped --name t5memory -p 127.0.0.1:4041:4040/tcp -v /home/translate5tmservice/.
t5memory-in-docker:/root/.t5memory/MEM --entrypoint  /root/t5memory  translate5/t5memory --port=4040 --
alsologtostderr=1 --v=0
```

The command and its parameters explained:

Command part	description	Multiple t5memory instances
docker run -d	Run the container detached, so in the background	
--restart unless-stopped	Ensure that the container is restarted (f.e. after reboot)	
--name t5memory	The label of the container	**must be changed** if multiple containers of t5memory should run.
-p 127.0.0.1:4041:4040/tcp	The port mapping from where the internal port (4040) is mapped to on the host (127.0.0.1:4041)	**must be changed** if multiple containers of t5memory should run.
-v /home /translate5tmservice/.t5mem ory-in-docker:/root/. t5memory/MEM	The data path on the host mapped into the container - **this affects the TM storage and persistency!**	**should be changed** if multiple containers of t5memory should run, there might be scenarios where multiple instances share the same files and therefore the same mapping
--entrypoint /root/t5memory	The command which should be called in the container on startup	
translate5/t5memory	The container image, with any version tag the latest image locally available is used. A version tag might by added here: translate5/t5memory:0.3.17	When running multiple versions with different versions, version tag should be change accordingly.
--port=4040 -- alsologtostderr=1 --v=0	The parameters passed to the command defined in --entrypoint, so direct t5memory parameters	

Update of the above running container

To update / change the image version of the above running container, do the following:

```
docker stop t5memory                   # stops the running container
docker rm t5memory                      # removes the container - danger! If the data is not mapped via -v
volumes parameters, the data is lost!
docker pull translate5/t5memory         # updates the image to the latest version

#  or to pull a specific version:
docker pull translate5/t5memory:0.3.17   # optionally pulls a specific version

# the same docker run as explained above, so by default:
docker run -d --restart unless-stopped --name t5memory -p 127.0.0.1:4041:4040/tcp -v /home/translate5tmservice/.
t5memory-in-docker:/root/.t5memory/MEM --entrypoint  /root/t5memory  translate5/t5memory --port=4040 --
alsologtostderr=1 --v=0
```