GLT API

Adrián Castelló 03/09/16

GLT (Generic Lightweight Threads)

- GLT != Specification
- MassiveThreads, Qthreads and Argobots
- Two parts
  - CORE: Functionality supported by all the libraries
  - Extended: All the functionality allowed by each library
- Two implementations
  - Static library
    - Better performance expected
    - Needs recompilation
  - Dynamic library
    - Does not need recompilation

CORE

Module	Number of functions
Init & Finalize	2
Work unit management*	18
Mutex	5
Barrier	3
Condition	5
Timer	6
Helper functions	2
Query functions+	27

* Tasklets and ULTs (If a library does not allow the use of tasklets, a ULT is used)

+ One for each module of the extended implementation

Extended modules

MassiveThreads	Qthreads	Argobots
WS API	Basic	Event
Profiling	Atomic	Future
Log	FEB	Promise
Seralize	Reduction (Sinc)	Extended mutex
Key (TLS)*	Loop	Key (TLS)*
Felock	Util	Self
Extended Work units	Data Structures	Pools
	Syscall	Scheduler
	Extended Runtime	Threads
	Memory	Extended Tasklets
		Extended ULTs

*Same functionality but it can not be implemented with Qthreads primitives

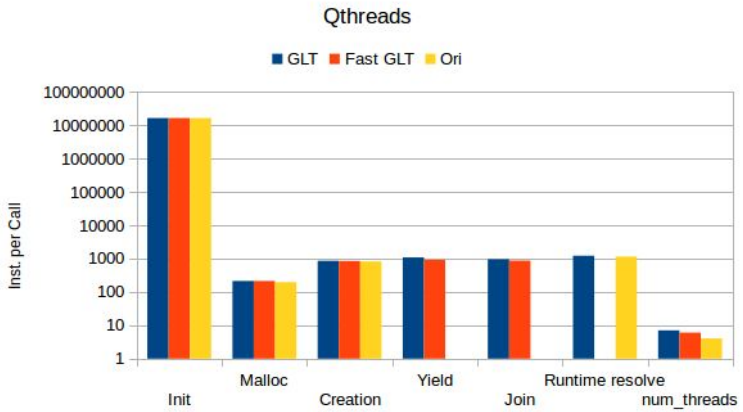
Programming Model I

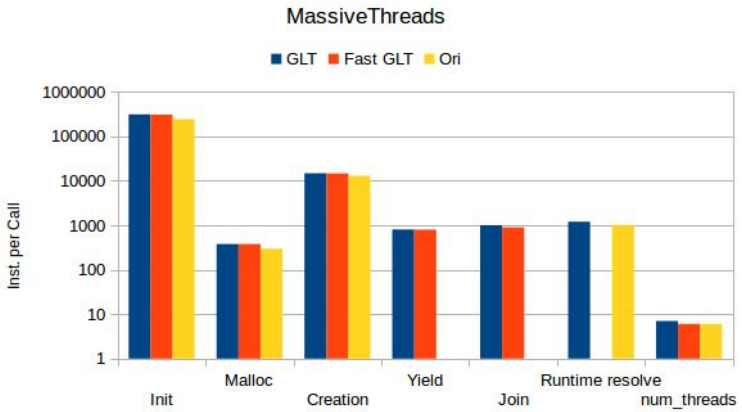
- By default one thread per logical CPU is created
- GLT_NUM_THREADS environment variable can change the number
- MassiveThreads
  - MYTH_WORKER_NUM can be used if the first one is not set
- Qthreads
  - QTHREADS_NUM_SHEPHERDS can be used if the first one is not set
  - GLT_NUM_WORKERS_PER_THREAD or QTHREAD_NUM_WORKERS_PER_SHEPHERD defines the number of workers managed by each thread (1 by default)
- Argobots
  - GLT_NUM_POOLS defines the total number of pools (1 for each threads by default)
  - If less pools than threads are created, they are shared and assigned using a round-robin mechanism
  - In the extended version, programmers can create as many threads and pools (and schedulers) as they need but the management will be controlled by them

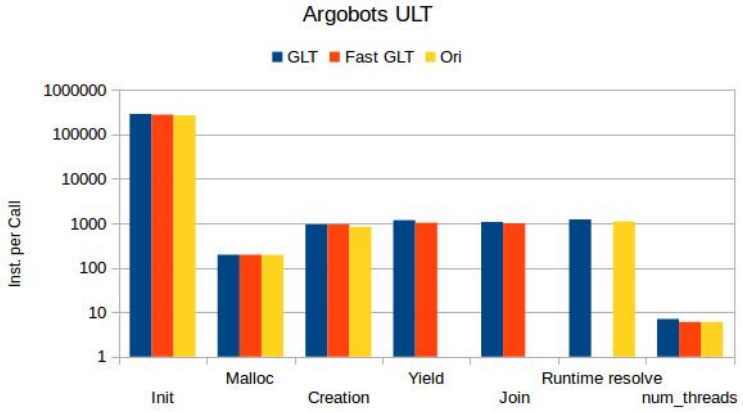
Programming Model II

- Thread AFFINITY is enabled by default
- Qthreads
  - If just one thread (Shepherd) and more than one worker are created, the thread is bounded to a node and the workers to a cores.
  - If more than one thread is created, it is bounded to a core

GLT API Overhead







Runtime resolve does not appear in Fast GLT due to the static linkage.
The values in Yield and Join operations come from the runtime load function.

Ori: code -> library
GLT: code -> GLT library -> library
Fast GLT: code ->library