ParallelFX : parallelism made easy

Jérémie Laval (Garuma)

jeremie.laval@gmail.com

http://garuma.wordpress.com

What ?

• Ease the development of parallel (multi-threaded) applications that take advantage of multi-core processors.

• Written by Microsoft and running only on .NET.

• 3 main components :
  - A Task API similar in usage to classic Thread
  - Parallel loops : for, foreach...
  - Plinq (Parallel Linq) : allows Linq queries to run in parallel

The goal : create an open-source cross-platform implementation of ParallelFX running on Mono.

Why ?

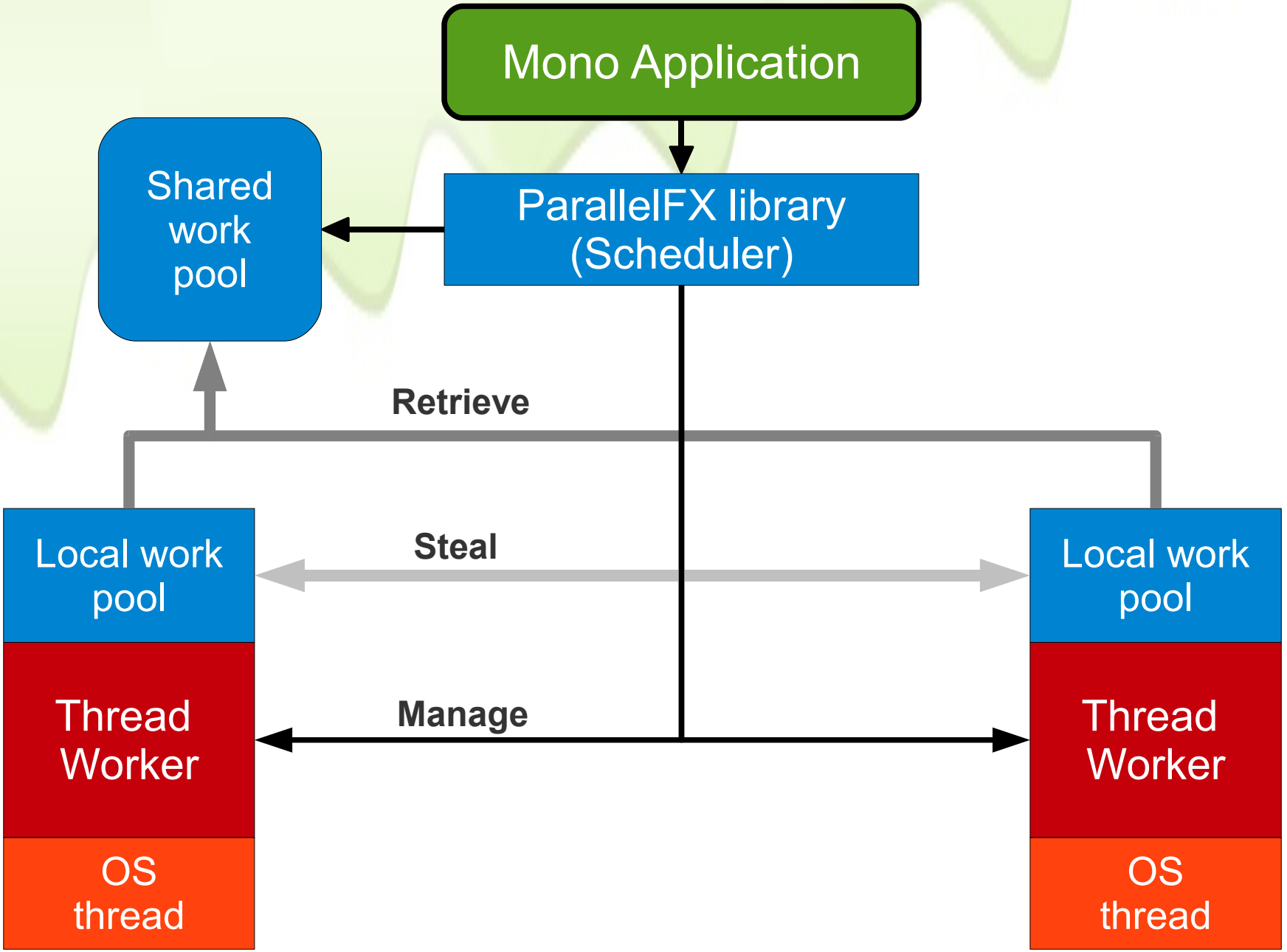
- Today trend is to improve the number of core in CPUs, not their individual speed.

- Theoretically it should give a ×n performance boost but it's not (n being the number of core).

- Reason : application are designed to be single-threaded.

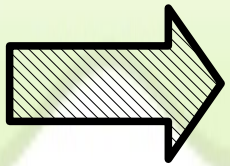
- Second reason : usually doing multi-threading "by hand" is hard and not efficient.

How ? (current design)



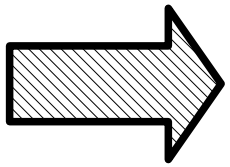
Peeking into the "magic"

Shared work pool



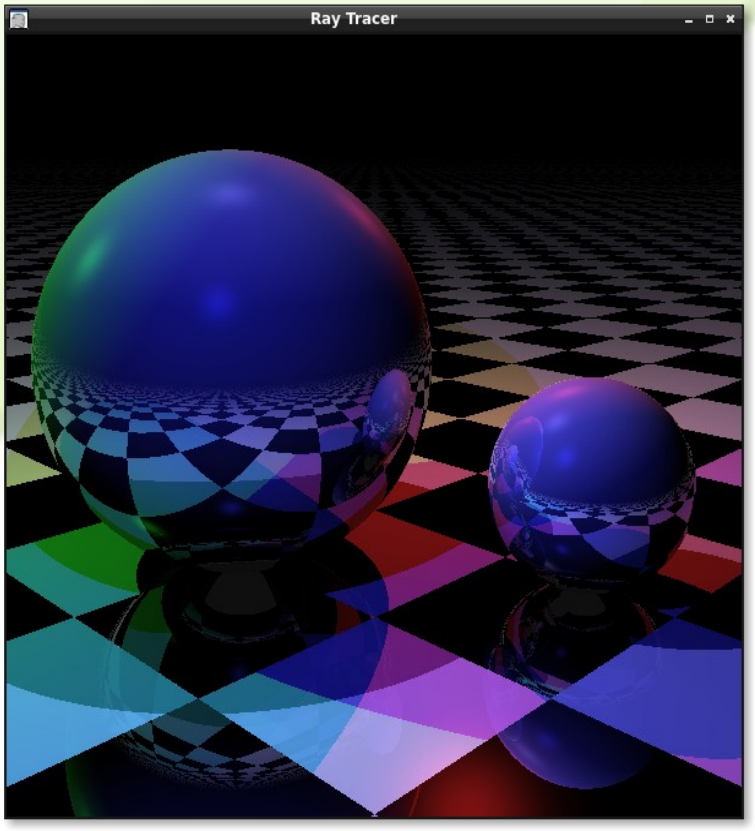
- Stack with a back-off layer.
- Uses CAS for atomic operations, completely lock-free.
- Back-off layer allows correct performances at high load.

Local work pool



- Deque-like (3 operations : pushBottom, popBottom, popTop).
- Worker uses pushBottom & popBottom (LIFO-style).
- popTop used by stealers (FIFO-style).
- Minimize CAS and uses no lock.

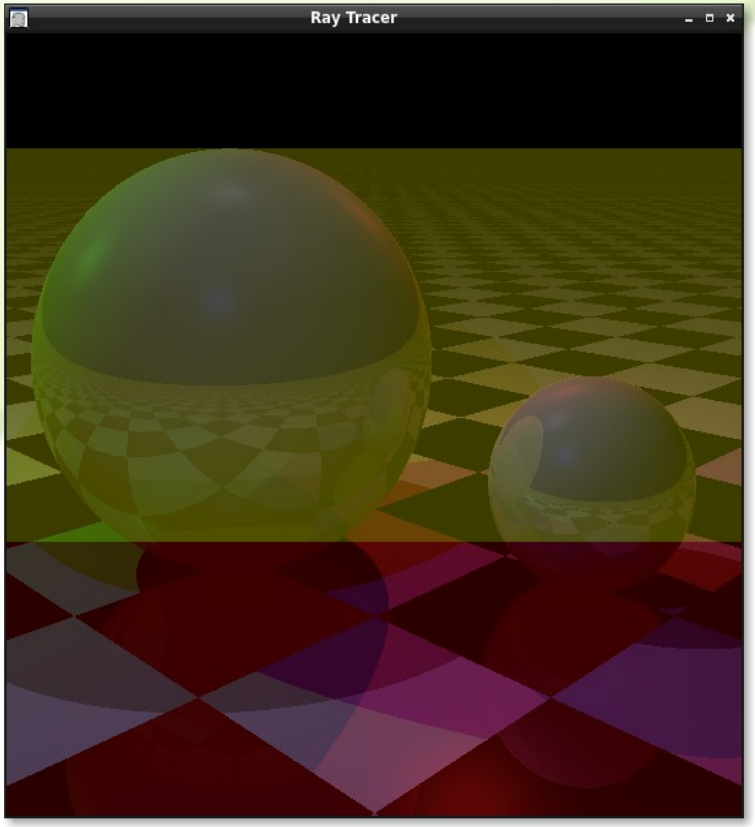
Example



- A classic ray-tracer implementation written with no multithreading in mind.

- Processing time on my computer : ~31s.

Example (next)



- Using current implementation of ParallelFX. Color mask represents the ThreadWorker which did the work.

- Processing time on my computer : ~18s, almost 42% speed-up.

Thanks for your attention

Questions ?