
University of
Kent

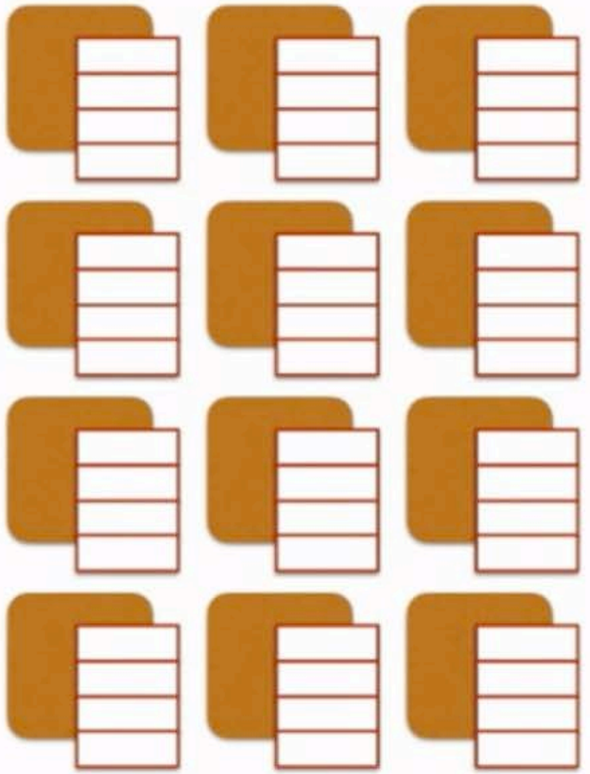
University of Kent

Multicore Erlang

University of Kent

Multicore Erlang

Each core has a separate run-queue.



University of Kent

Multicore Erlang

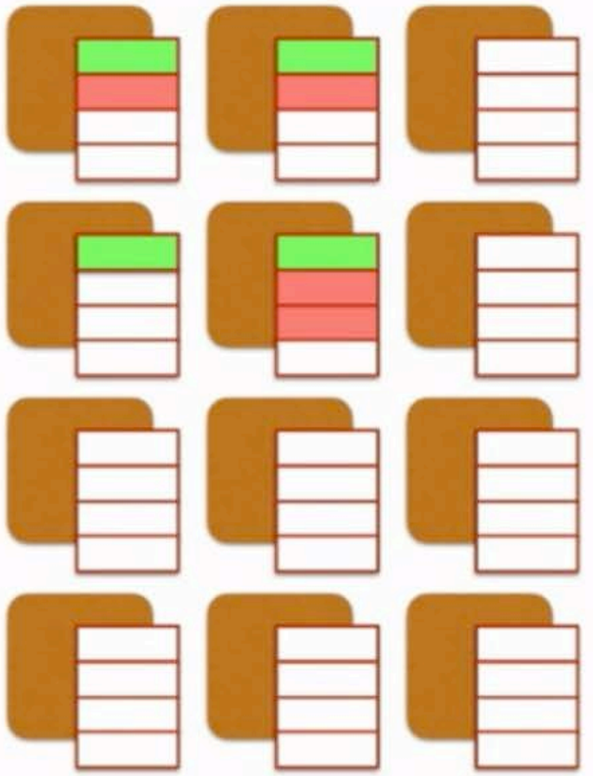
Each core has a separate run-queue.

Scheduling is per-core.

Processes spawned on the core that spawns them.

Processes move between cores by means of work stealing.

Garbage collection is per-process.



University of Kent

Scaling up with multicore

Multiple processes can be active simultaneously, one on each core.

Independent, sequential computations can be done in parallel.

Stateless services can be duplicated.

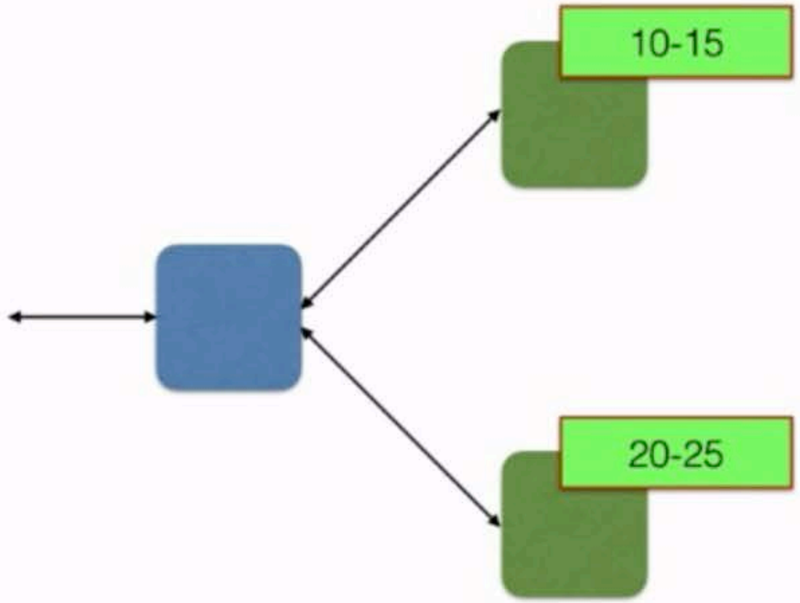
Some stateful services can be split or sharded.

University of Kent

Scaling up the frequency server

Front-end routes requests to servers 1 (10-15) and 2 (20-25) … and replies to the client.

Front-end can track which servers have available frequencies … and how many each.




University of
Kent