Week 19

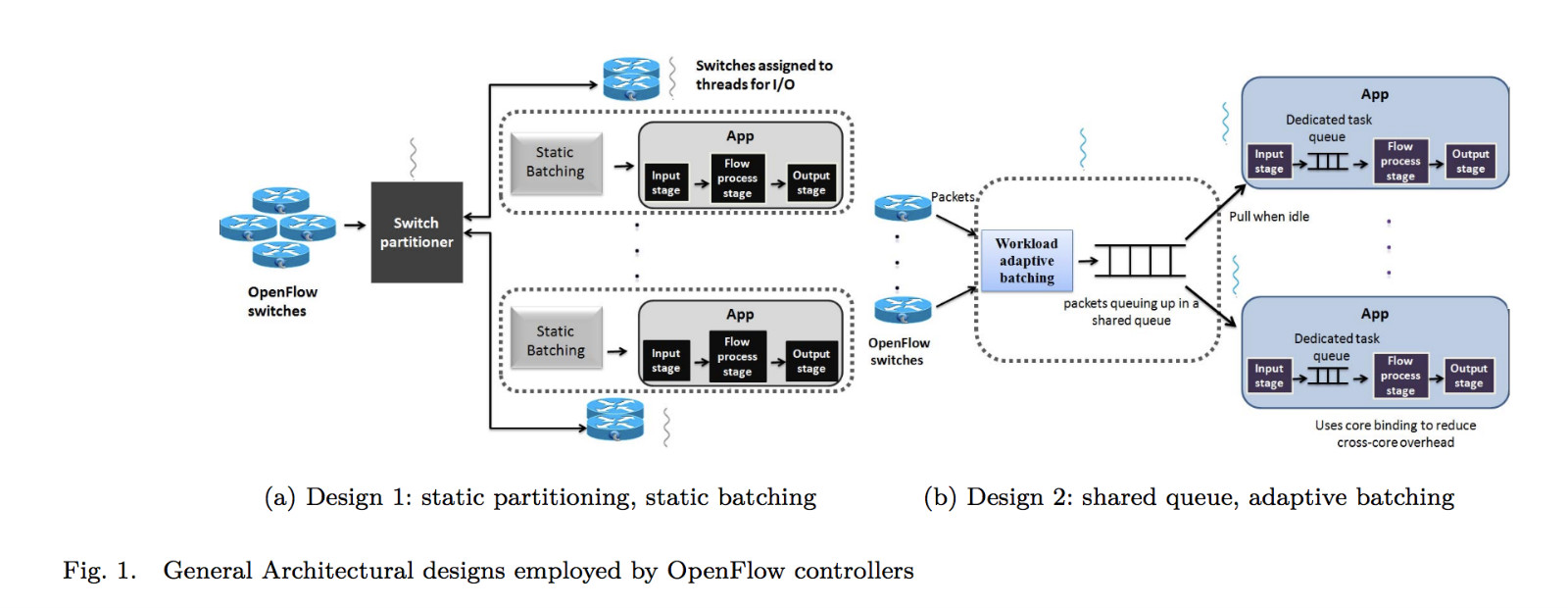
By comparing the internal architectures of the four open-source OpenFlow controllers being considered in this work. The key architectural features that differ between these controllers are:

Multicore Support: in terms of the number of main (if any) and worker threads started by the multithreaded controller running on a system with P cores;

Switch Partitioning: defined in terms of the distribution and allocation of connected OpenFlow switches to worker threads running in the controller;

Packet Batching: a technique used by all OpenFlow controllers where multiple bytes are read from or written to the underlying network using a socket buffer; this feature helps prevent the overhead of a socket (read/write) system call for a large number of pending requests, thereby improving the overall throughput of the controller;

Task Batching: a strategy used to allocate already received packets to the worker threads for processing, hence directly impacting the latency of the controller.



These architectural choices are shown pictorially in Figure 1. The first design [Figure 1(a)], uses static switch partitioning and static batching and is employed by Beacon, NOX-MT and Floodlight. A switch partitioner module that runs inside the main thread listens for incoming switch connections and partitions them amongst the worker threads. Once a switch has been assigned to a particular thread, it is responsible for all the subsequent I/O from that switch.

The second design, uses shared-queue, adaptive-batching design and is employed by Maestro. The main thread is responsible for listening to new switch connections and storing incoming packets in a shared queue by employing packet batching technique. Once packets have been stored in the shared-queue, worker threads can pull these packets and perform packet processing.