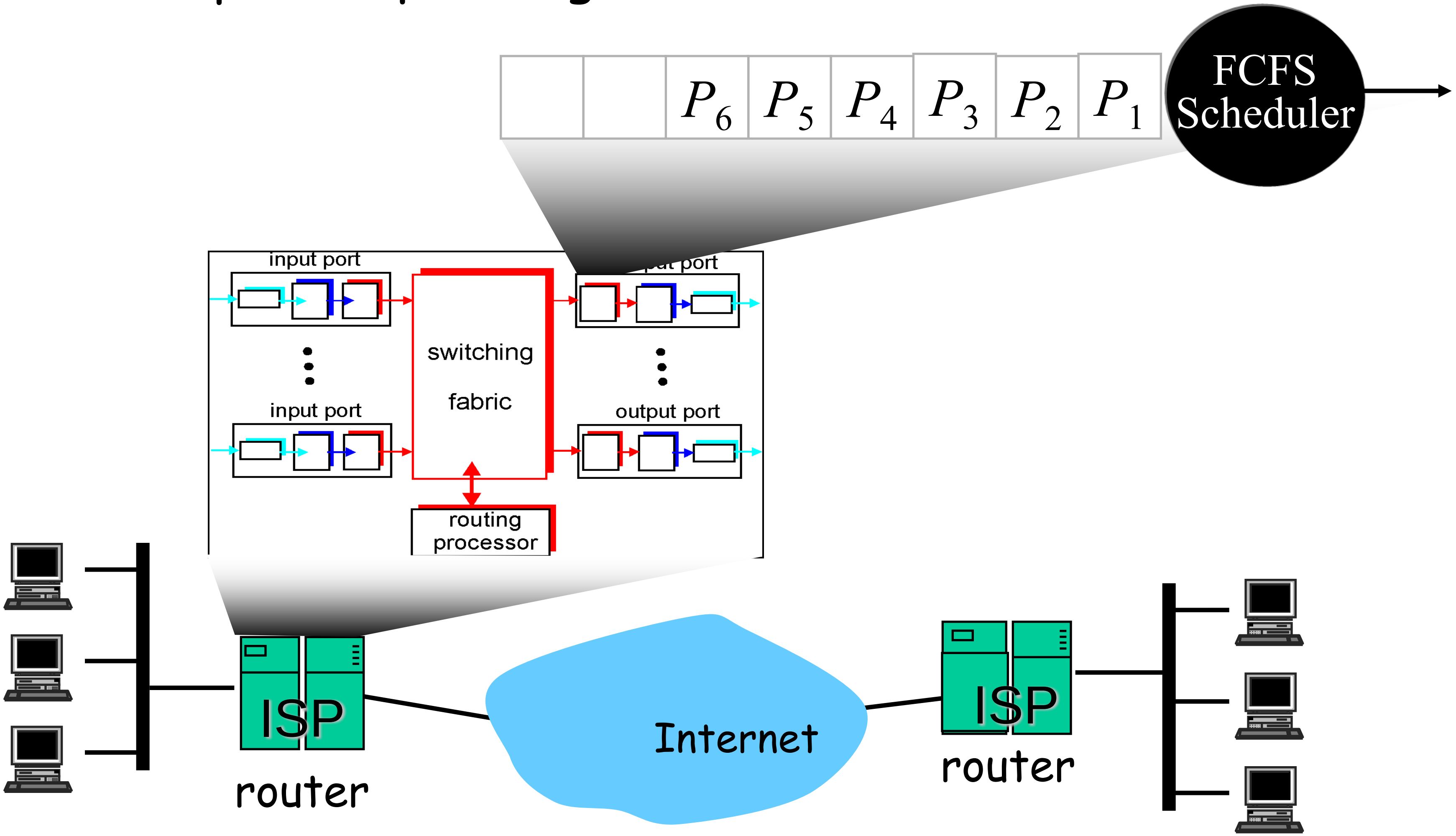
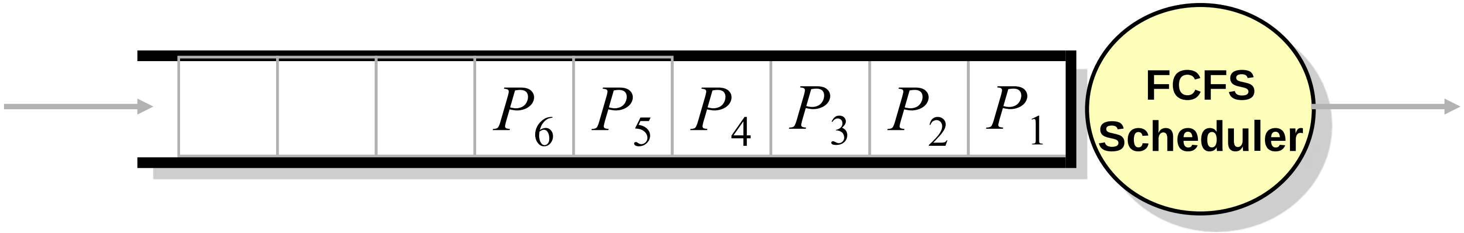
Randomization in Router Queue Management

Randomization in Router Queue Management

- normally, packets dropped only when queue overflows
  - "Drop-tail" queueing

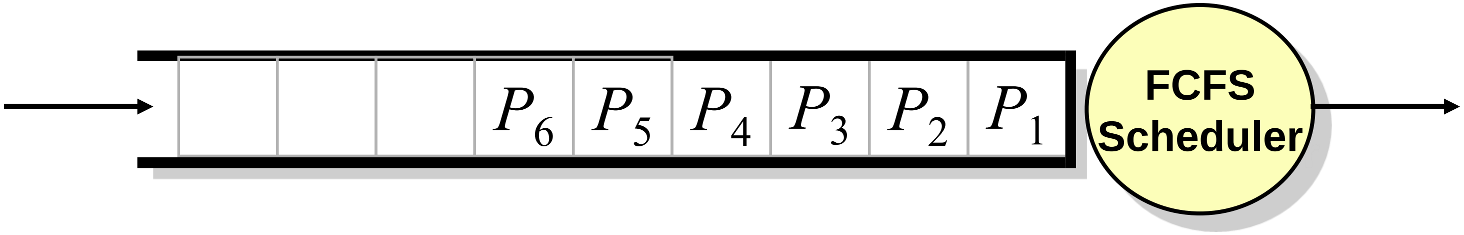


The case against drop-tail queue management



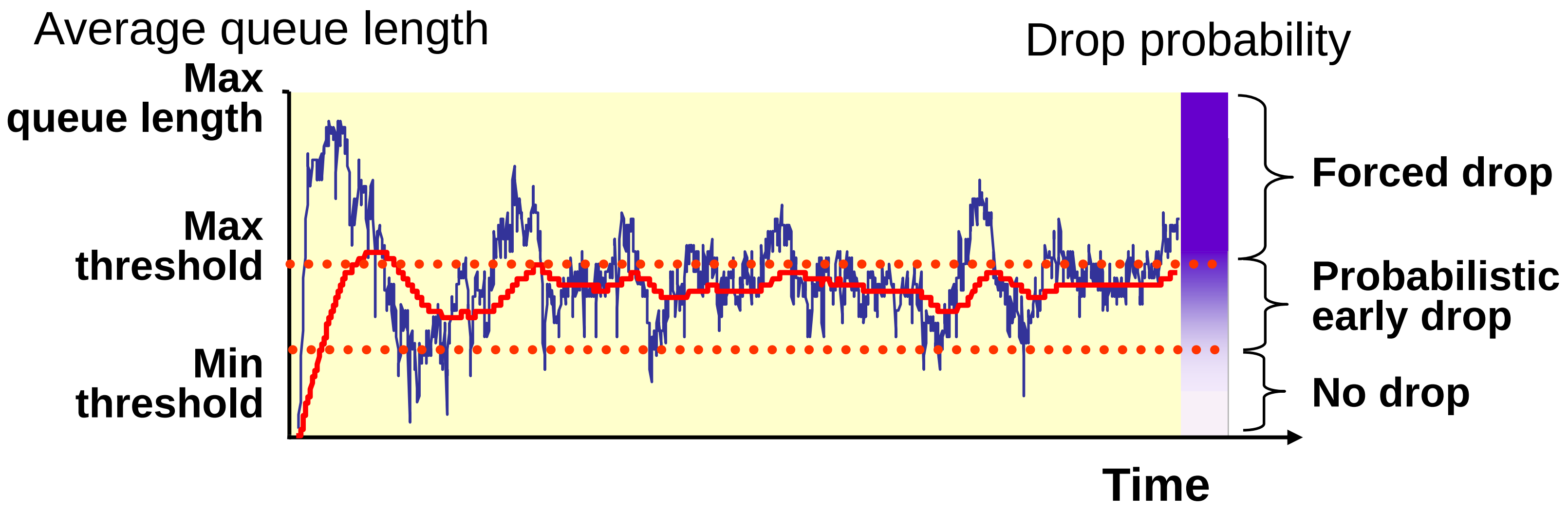
- large queues in routers are “a bad thing”
  - End-to-end latency dominated by length of queues at switches in network
- allowing queues to overflow is “a bad thing”
  - connections transmitting at high rates can starve connections transmitting at low rates
  - connections can *synchronize* their response to congestion

Idea: early random packet drop



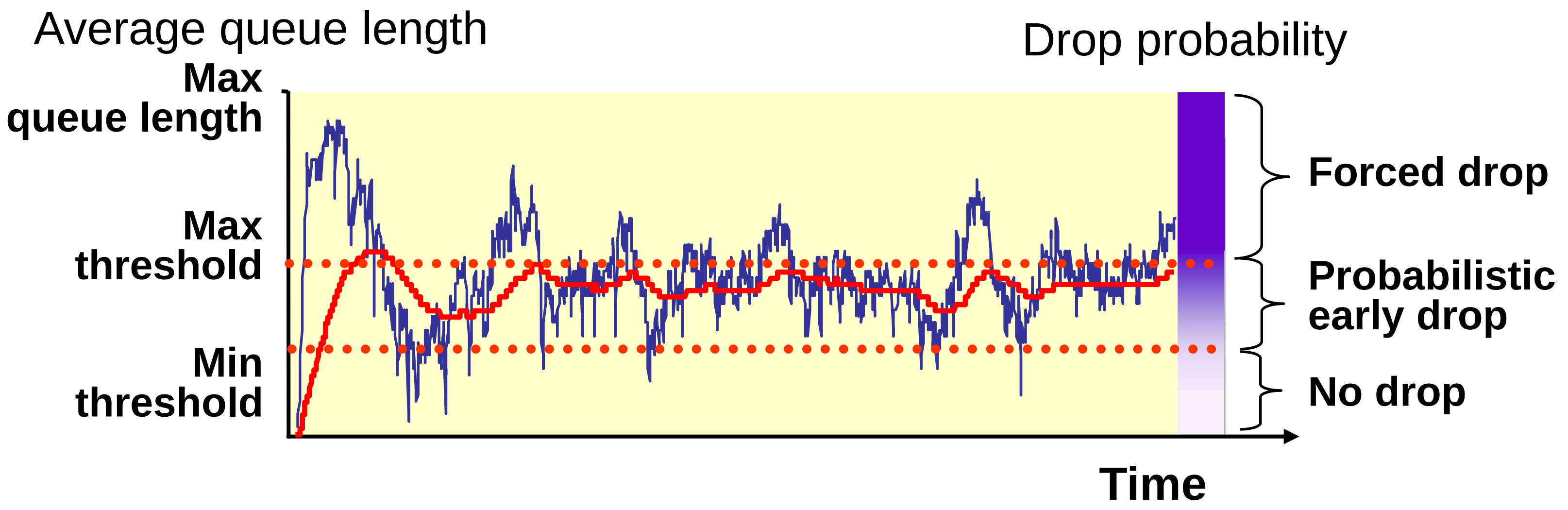
- When queue length exceeds threshold, packets dropped with fixed *probability*
  - probabilistic packet drop: flows see same loss *rate*
  - problem: bursty traffic (burst arrives when queue is near full) can be overpenalized

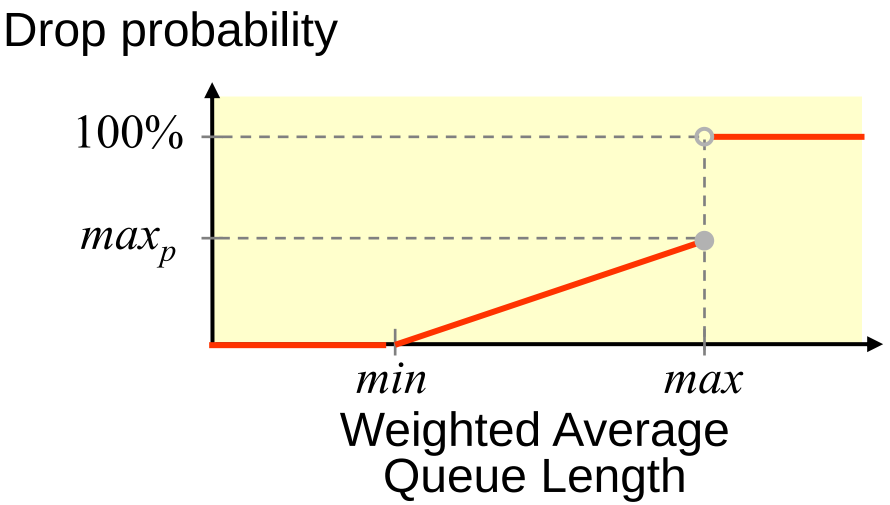
Random early detection (RED) packet drop



- use exponential *average* of queue length to determine when to drop
  - avoid overly penalizing short-term bursts
  - React to longer term trends
- tie drop prob. to weighted avg. queue length
  - avoids over-reaction to mild overload conditions

Random early detection (RED) packet drop





Random early detection (RED) packet drop

- large number (5) of parameters: difficult to tune (at least for http traffic)
- gains over drop-tail FCFS not that significant
- still not widely deployed …

RED: why probabilistic drop?

- provide gentle transition from no-drop to all-drop
  - provide “gentle” early warning
- provide same loss rate to all sessions:
  - with tail-drop, low-sending-rate sessions can be completely starved
- avoid synchronized loss bursts among sources
  - avoid cycles of large-loss followed by no-transmission