Critique 2

The paper discusses how to implement scheduler in Xen in multi-core scenario. The implementation itself is pretty straightforward, which consists of two different schedulers: global and partitioned scheduling policy. In regards to the overhead, I actually don't consider the overhead would be a huge problem as the tasks each VCPU needs to handle is relatively small compared to OS, so even when the scheduling policy might take theoretically long time to finish running, the limited number of tasks would make this overhead fairly negligible.

However, the biggest roadblock to achieve an efficient scheduler is the resource dispatching and frequently context switching. While Xen implements the earlies deadline first policy, the problem of context switching can be set aside for now. Yet, the resource dispatching is still going to be a problem. The fact that the partitioned scheduler can possibly steal resource from another VCPU partially relieve the problem, yet when we take the total network bandwidth and socket buffer into account, we can find that while tasked are indeed scheduled properly, the underlying resources (bandwidth and all kinds of buffers) still might suffer from priority inversion. For example, there are two tasks A & B. A is a fire alarm notification application and B is a real time temperature sensor. Apparently A has a much higher priority than B and our scheduler would indeed schedule A more often than B. But we need to notice that B might pumping crazily amount of packets to the underlying socket, triggering the congestion control. So when it is A's turn to send packet, the packet itself might be jammed as there is not enough budget for the socket to send the packet. Moreover, even though the socket is not overflowed, a limited bandwidth is going to be fatal as packet might be dropped or delayed due to the misbehavior of a low priority application. Furthermore, the resource dispatching of I/O resources is another thing that we should worry about. If a low priority application holds the lock of an I/O resource and later on, when a high priority application tries to grab the I/O resource, it might fail to do so and cause unnecessary delay.

So apart from implementing a good CPU resource scheduler, one also needs to take various other resources into account to achieve a true efficient and provisioned scheduler.