Q: This paper is not considering capital costs for the replication

A: We assume that the hardware resources are fixed (Eq. 4 and Eq. 5), so the CAPEX are also fixed for all fault tolerance alternatives. The difference is how the HW resources are used by each fault tolerance approach.

Q: A core level MTBF is not really appropriate with multi-core processors.

A: Similar to [Casanova-INRIA-2012], we use the term core to represent the computing resource allocation unit (a core, a multi-core processor, etc.). We extend the range of MTBF to 1000 years, at which Lazy Shadowing outperforms Process Replication by a larger extent (as high as 76.6%), and is slightly worse than checkpointing/restart (within 7%). We will update all the figures and analysis.

Q: Many high-end systems today don't at all support swapping or demand paging.

First, checkpoint/restart also requires additional memory capacity. Second, we acknowledge the fact that applications and compute kernels in existing HPC environments were simplified significantly. It became clear, however, that strategies designed to work around the capabilities of the hardware cannot scale to exascale computing (See recent DoE reports on exascale computing). Consequently, the research focus has been on new paradigms with dynamic, asynchronous mechanisms, such as demand paging and cache tuning. Support of efficient demand paging is particularly critical as the data of future exascale applications (e.g., visualization) may not fit entirely in memory.

Q: What is the impact of replica synchronization, especially with collocation, on failure-free and faulty performance and energy consumption?

A: We only perform shadow leaping when a fault occurs and all mains are idle (Fig. 3). The time for leaping overlaps with failure recovery, and the energy consumption is considered in Eq. 5.

Q: The shadow process would be also subject to failure.

A: The underlying assumption of our failure model is that the probability that the main and its associated shadow fail simultaneously is extremely low. Mapping it to the well-known birthday problem, this assumption has been validated in [Ferreira-SC-2011], and is further corroborated in [Casanova-INRIA-2012].

Q: With message logging, the tossup is between a node that is slower to execute but faster to recover because of the shadow process, or the node that executes at full speed but will have to replay entirely from the checkpoint.

A: The concern raised by the reviewer is precisely why we introduced the concept of "leaping shadows". The proposed approach to deal with this problem is a distributed and coordinated solution. Upon failure of the main process, its associated shadow executes at a higher rate to “catch up”. The remaining main processes, however, become idle as soon as they reach their synchronization barriers. The leaping shadow model takes advantage of this idle period to allow all shadows to leap forward to the same state of their associated main, thereby ensuring forward progress and saving significant computational work (and thus energy).

Q: How are failures detected?

A: Failure detection is inherent in all fault-tolerant models. In the case of Lazy Shadowing, failures can be detected using heartbeat protocol. For simplicity and fairness, the cost was not considered for all fault tolerance approaches covered in this paper.

Q: What is the empirical support for a MTBF of 25 years?

A: Unfortunately, there is no commonly agreed on figure for MTBF. One study even points out that application behavior can affect MTBF [Sarood-SC-2013]. Google assumes 30 years for super reliable machines [Dean-LADIS-2009], so we have studied the range of 5 to 25 years.

Q: If a process dies, how does the whole program learn that the shadow process is the new endpoint for all MPI communication?

A: Like in MPI, a run-time system is necessary to coordinate all processes for Lazy Shadowing. It is also responsible for replacing the old endpoint with the new endpoint for communication.

Q: Lack of implementation

A: We implemented an MPI-based prototype, which can execute HPC workloads with lazy replicas [Mills-PDP-2014]. It was demonstrated that lazy shadowing effectively reduces overall energy, and the savings are application dependent. However, since the goal of this paper, which was submitted to the algorithm track, is to introduce algorithmic extensions of the shadow replication paradigm by discussing the novel concepts of shadow collocation and leaping, we did not include a discussion about the implementation. We will include a section in the paper to summarize the basic details and findings of the implementation.