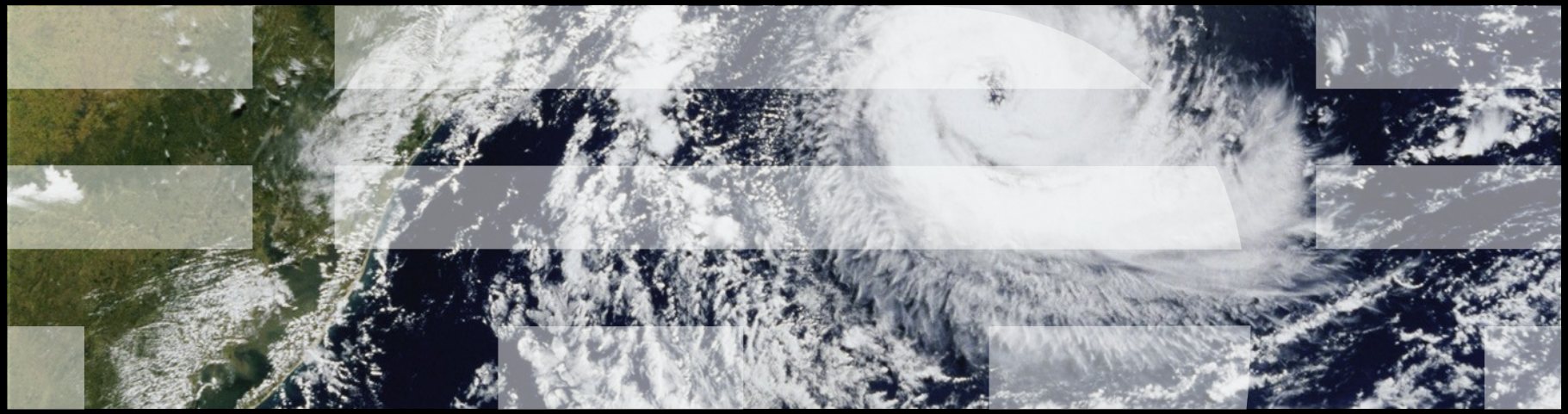
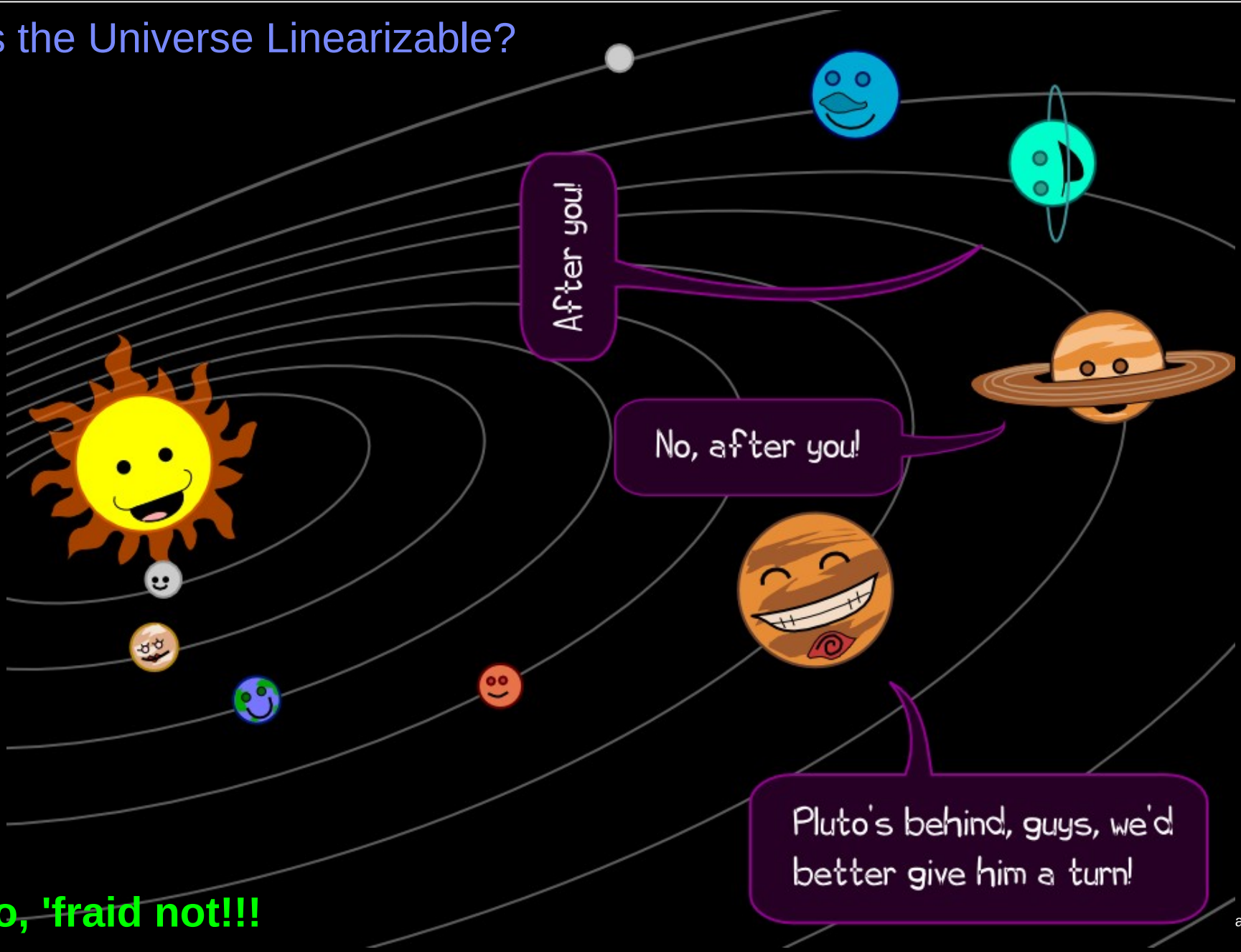
Paul E. McKenney – IBM Distinguished Engineer, Linux Technology Center

May 5, 2015

Linearizability: Who really needs it?



Is the Universe Linearizable?



No, 'fraid not!!!

Costs, Benefits, and Roles of Linearizability

- Costs:
  - "Linearizable counting networks" by Herlihy, Shavit, and Waarts:
    - "Finally, we prove that these trade-offs are inescapable: an ideal linearizable counting algorithm is impossible. Since ideal non-linearizable counting algorithms exist, these results establish a substantial complexity gap between linearizable and non-linearizable counting."
  - Doesn't necessarily help client code much
    - Haas et. al, "How FIFO is Your Concurrent FIFO Queue?"
  - I recently received a an RCU iterator patch to remove a ***single load from the stack***
- Benefits:
  - Simplifies analysis and verification (but doesn't necessarily help the client code much)
  - In the concurrent theoretician's toolbox, it is analogous to the hammer
- Role:
  - Legacy proof systems
  - Reasoning about "bulk concurrent code"
  - Both unnecessary and insufficient for critical hotpath code: Need something else for:
    - Extreme real-time response (30 microseconds in Linux kernel in 2006)
    - Extreme performance and scalability (OS kernels, server applications)
    - Code for statistics gathering and some classes of diagnostics
- Yes, can transform non-linearizable specifications to linearizable w/complex specification
  - You can also describe planetary movements using epicycles

Legal Statement

- This work represents the view of the author and does not necessarily represent the view of IBM.
- IBM and IBM (logo) are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries.
- Linux is a registered trademark of Linus Torvalds.
- Other company, product, and service names may be trademarks or service marks of others.