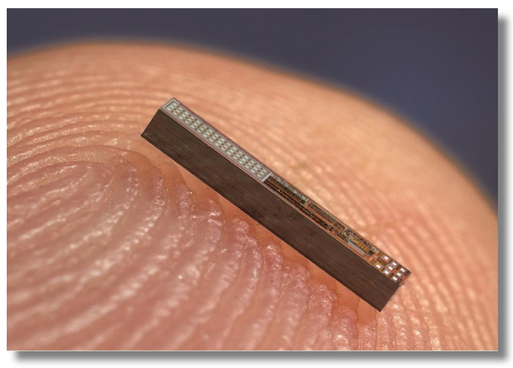
Automatic Verification of Intermittent Systems

Manjeet Dahiya and Sorav Bansal
{dahiya, sbansal}@cse.iitd.ac.in
Indian Institute of Technology Delhi, India

Background

Battery technology not at pace with transistor technology. We use energy harvesting, e.g., solar, RF signals to power small devices.



Implantable pressure sensing medical device
(1 mm diameter, 150 uW)

Untethered moth with RFID measurement sensor (< 1 cm^2, weighs 0.25 g)

Is harvested energy suitable?

- No, not adequate for continuous operation
- Unpredictable power failures

How do programs complete then?

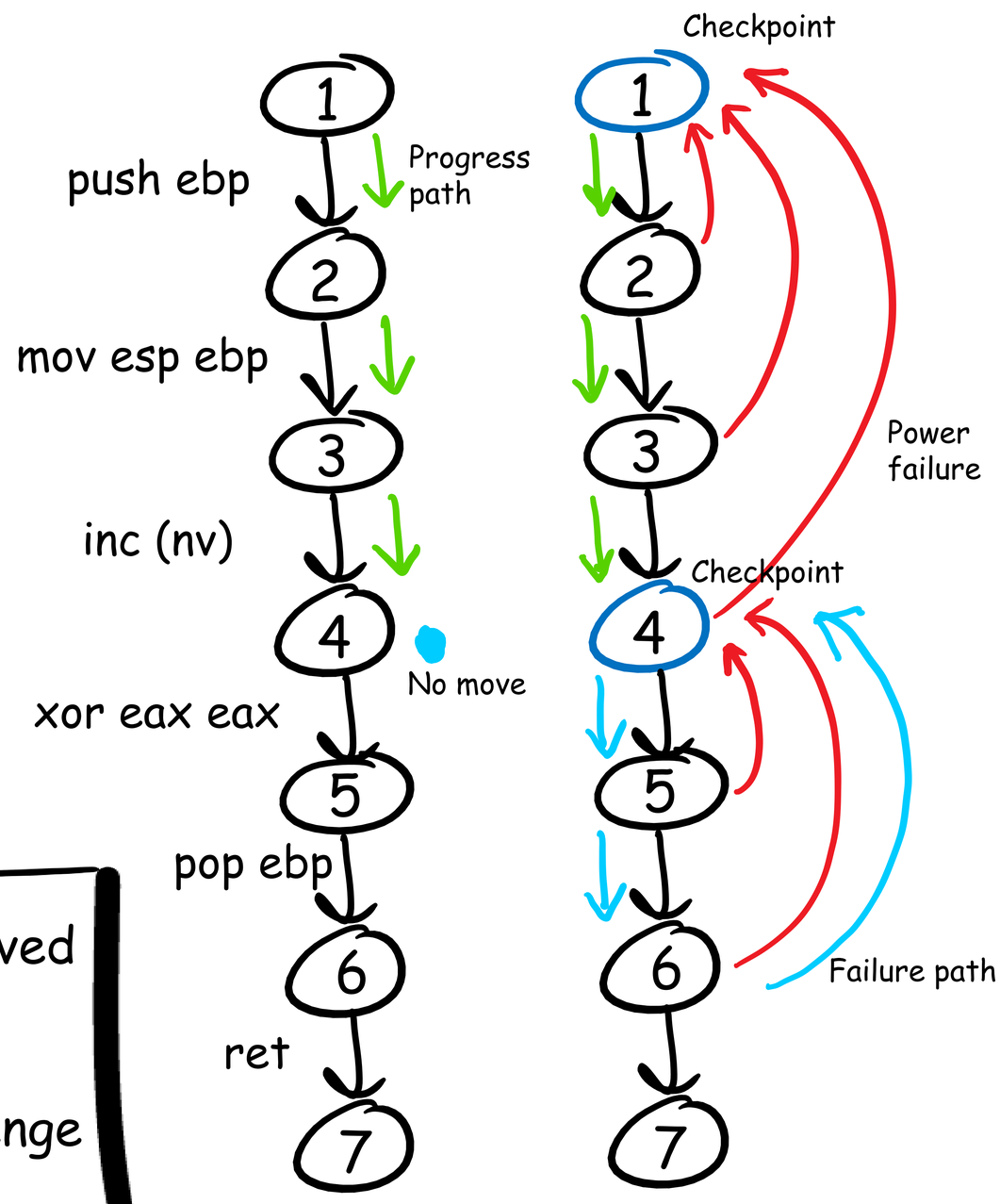
Intermittent Computation

- Checkpoints are inserted to save intermediate program state
- Power failures resume from the last saved checkpoint instead of the beginning

Motivation: Complexities and challenges with intermittent computation?

- Not complete program state is required to be saved
- Smaller checkpoints => faster runtime
- Volatile + Non-volatile memory => more complex
- Existing instrumentation tools report that challenge to work with compiler optimizations

Question we answer & Technique: Do the intermittent programs produce correct results?



C: continuous program **I: intermittent program**

State elements to save: **CP1:** esp, (esp), nv, eip
CP4: esp, (esp+4), eip

Is C == I?

- Undecidable problem, establish "==" by determining a weak bisimulation relation
- Correlation:
  - Power failure in I <-> No move in C
  - Progress path in I <-> Progress path in C
- Invariants: Houdini guess-and-check

Evaluation: verification & synthesis of checkpoints

- Avg. verification time 73s
- Implemented a greedy synthesis loop to minimize checkpoint size
- Checkpoint data size reduction: avg. 4x over DINO [PLDI2015]
- Synthesis time: 42s – 7h

MORE INFO ↘ Automatic Verification of Intermittent Systems. Manjeet Dahiya and Sorav Bansal. In VMCAI 2018.