1- Automata Processing

- Used widely in different areas
- Von Neumann architectures are not efficient at FSM processing:
  - Irregular memory accesses
  - Limited Parallelism

Solution: Use Automata Processor (AP)

- Enables in-memory processing
- Exploits state parallelism of NFAs

2- Challenges & Opportunities

Applications are getting Bigger, but AP capacity is Limited

Challenge: Repeated Executions!

Opportunity: Underutilization of AP

Pattern mismatch → Many unused states are configured to AP

Potential Solution: Remove Cold states from the NFAs

3- Potential Benefits & Research Questions

Q1: How to predict Cold states?

- Oracular knowledge of input
- Solution: Use a small profiling input to predict the Hot/Cold states

Q2: How to partition NFAs?

- Arbitrary states partitioning
- Solution: Partition using Topological Order

Q3: How to handle mispredictions efficiently?

- Problem: Input stream execution on the predicted Cold set is too expensive
- Solution: SparseAP Execution Mode

4- Efficient Automata Processing on AP

5- Summary

- Observation: Repeated configurations and executions on AP which causes inefficiency
- Goal: Accelerate large-scale NFA processing on AP

  - Demonstrate that a large number of NFA states are Cold during execution but still configured to AP
  - Predict if a state is Cold or Hot @ compile time using a small profiling input
  - Propose topological-order based NFA partitioning into Predicted Cold and Predicted Hot states
  - Develop SparseAP to handle mispredictions efficiently using our proposed Enable and Jump operations

- Results:
  - 2.1x Speedup (up to 47x)