NetworkAppliance®

Using HMMs to Model and Simulate Stateful Network File System Protocol Workloads

R.J. Honicky

Stateful Network File System Protocols (e.g NFSv4, CIFS)

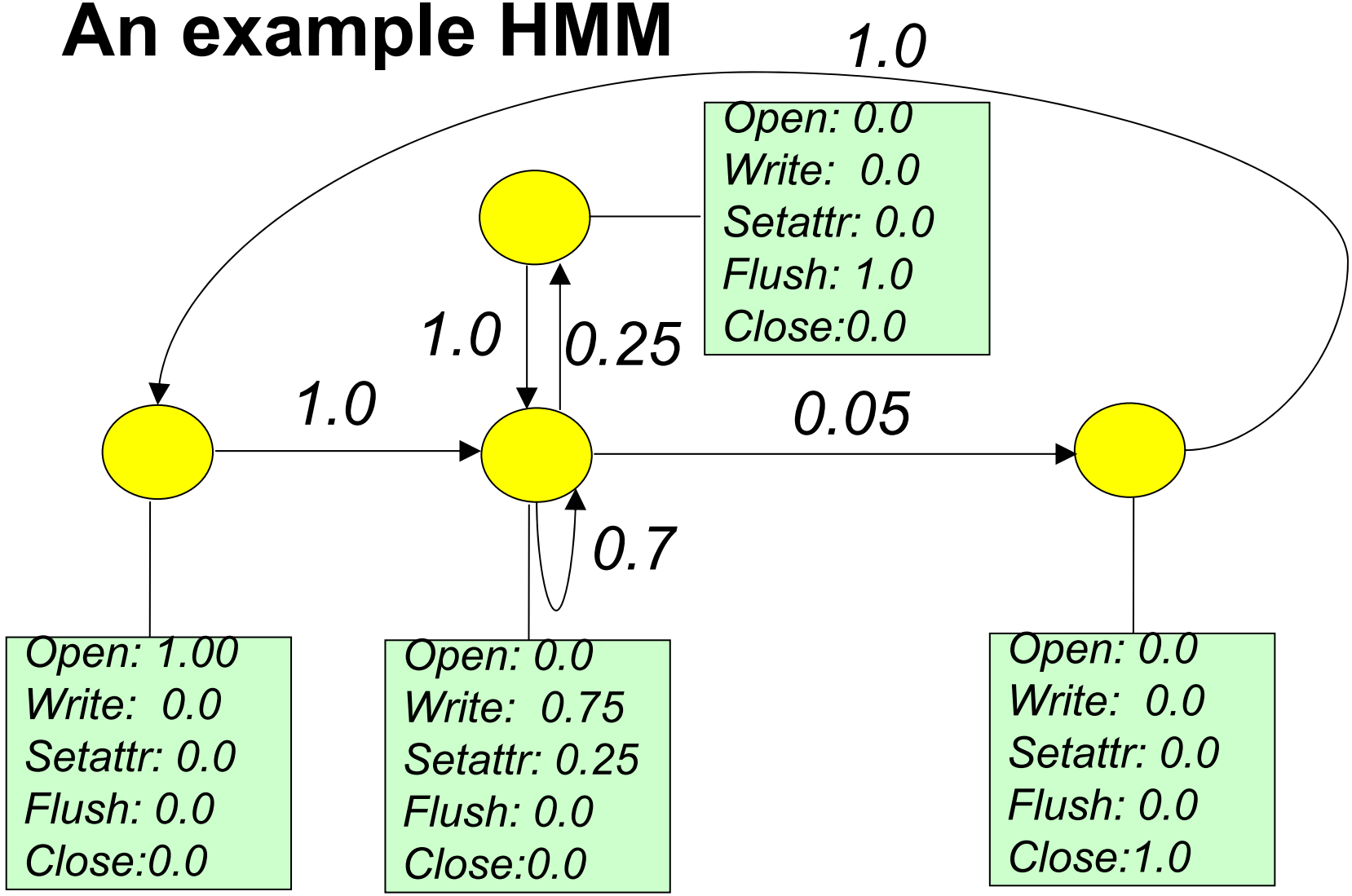
- **Some operations must precede others**
  - E.g. Open → Read → Close
- **Stateful behavior effects both correctness and performance**
  - Correctness: Read → Open is not allowed
  - Performance: Write → Flush != Flush → Write
    - Correct, but drastically different performance profile
- **Realistic workload generation requires a model which captures state**

NetworkAppliance®

Hidden Markov Models

- **HMMs are used to model a "Hidden" process**
  - learn the underlying state transition model with the "maximum likelyhood"
- **HMMs are similar to Probabilistic Finite State Machines**
  - Each node in the graph generates a symbol based on a distribution
- **Used for speech recognition, bio-informatics etc.**
  - Lots of theory and techniques
- **We use the vanilla algorithm to "learn" the best HMM for a trace**
  - Baum Welch algorithm (a special case of EM)
  - Can also learn the best HMM to model several traces at once
- **EM based algorithms often get stuck in local maxima/minima**
  - Use an evolutionary algorithm to recombine successful models and escape local maxima

An example HMM



Model Validation

- **How do we know that we are generating a realistic workload?**
- **First Order:**
  - Look at the mix of ops
    - Does it match the mix in the trace(s)
    - Yes!
- **Second Order:**
  - Count the frequency of different op pairs

    open read *read close*
  - Calculate "distance" between distribution of op pairs in generated workload and trace
  - Still figuring out the best distance metric
- **Looks good: proof by inspection ☺**

NetworkAppliance®