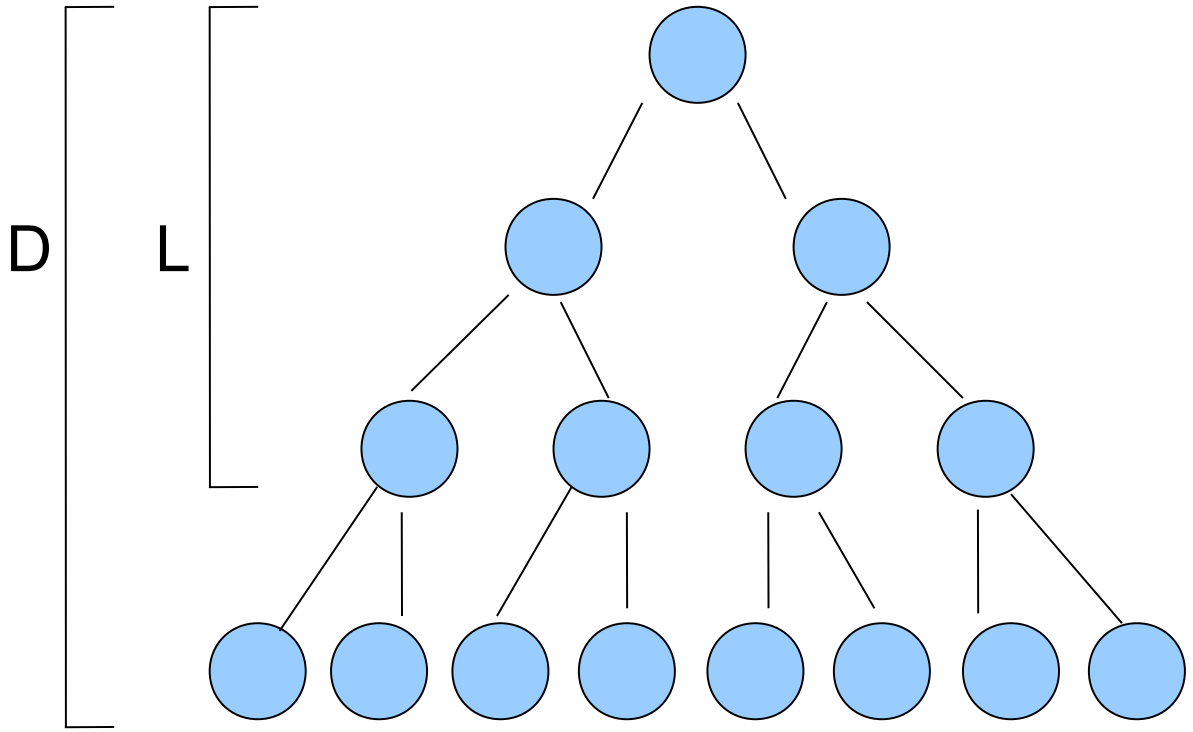
Modeling Processes Influenced By Hidden Processes

Or: How I Learned to Stop Tree Parsing and Love the Composite Machine

Greg Wimsatt
Phy 250: Computational Mechanics
June 1, 2013

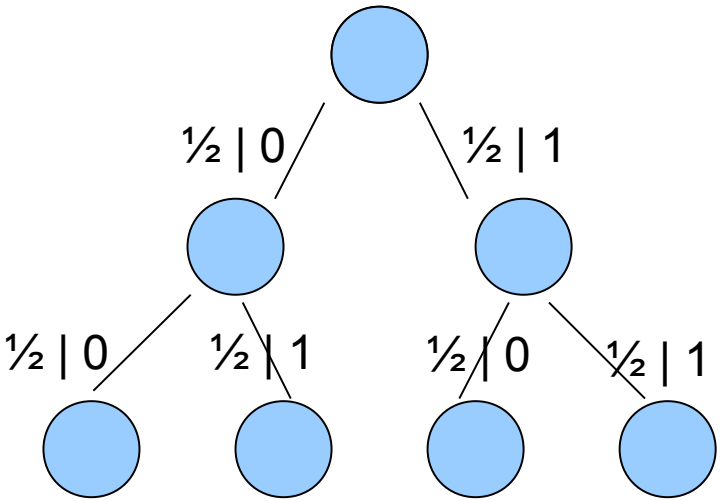
Parse Tree Reconstruction

- Epsilon machines are the best we can do with the past information
  - Each state predicts a future given the past faithfully by definition
- Parse Tree assumes a depth, D, and morph length, L
  - Assumes futures are only indeterminate up to future length L
  - Can find consistent morph construction that doesn't capture the full dynamics



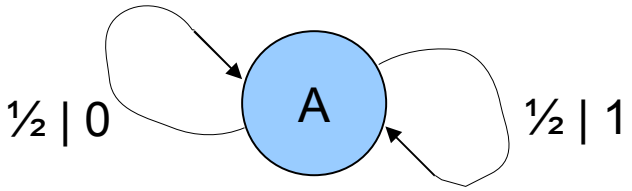
Example Parse Tree Mishap

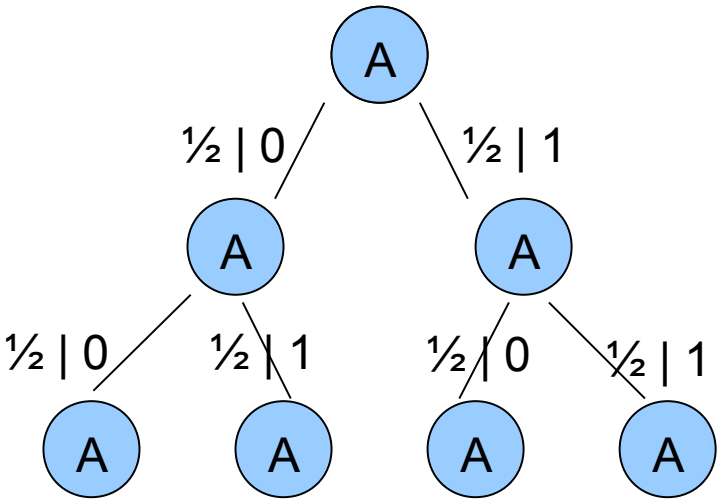
- Suppose we look at the following depth 2 tree
- What process do you think this is?



Example Parse Tree Mishap

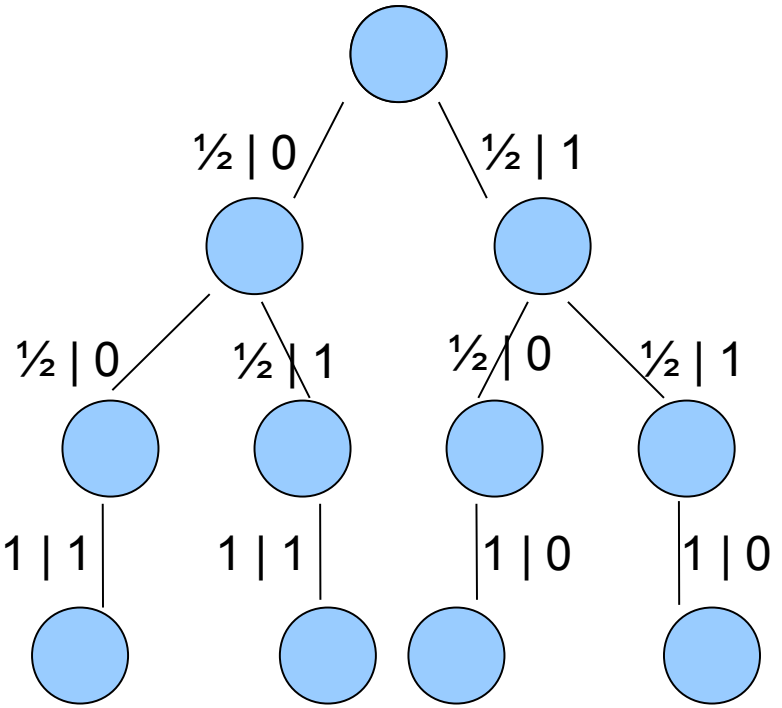
- Using L = 1, we'd reconstruct a fair coin





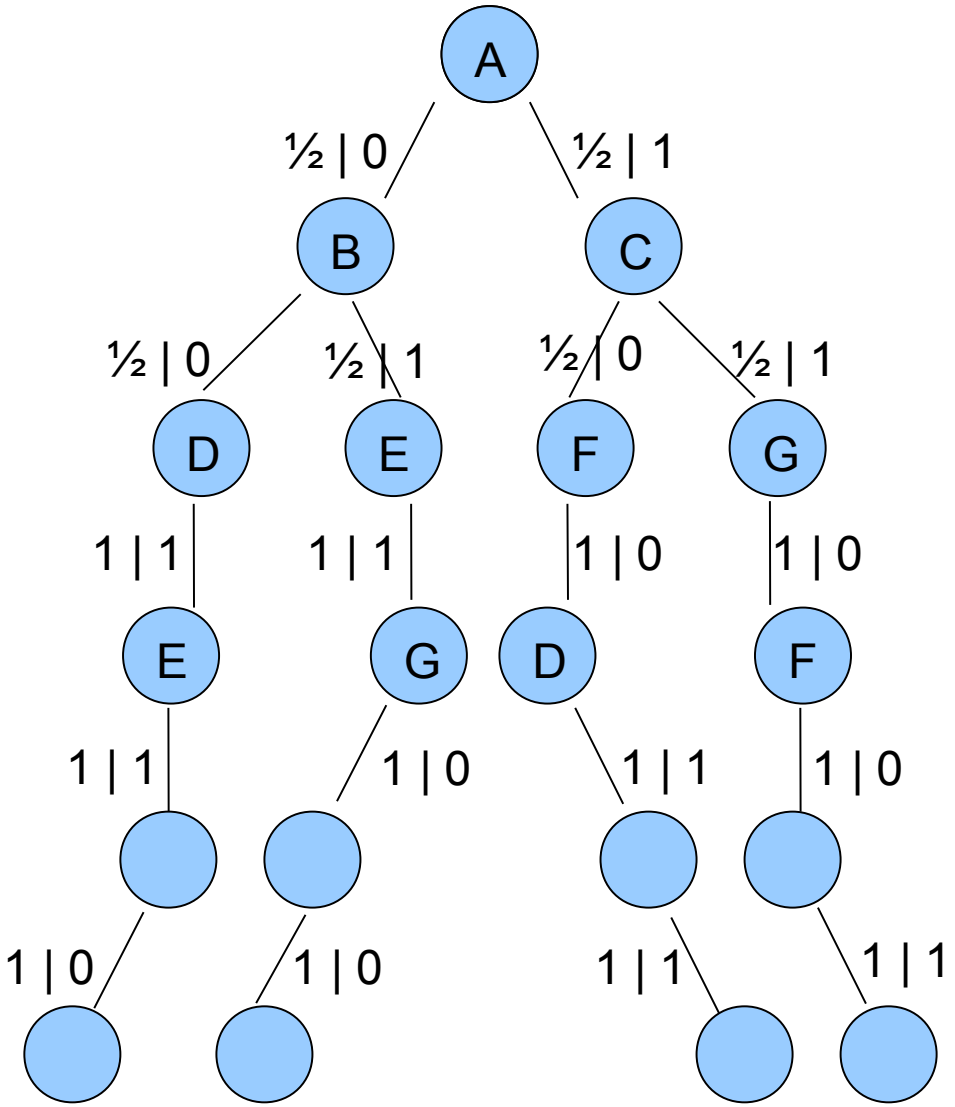
Example Parse Tree Mishap

- But suppose we took an extra character per word: D = 3
- And saw this?



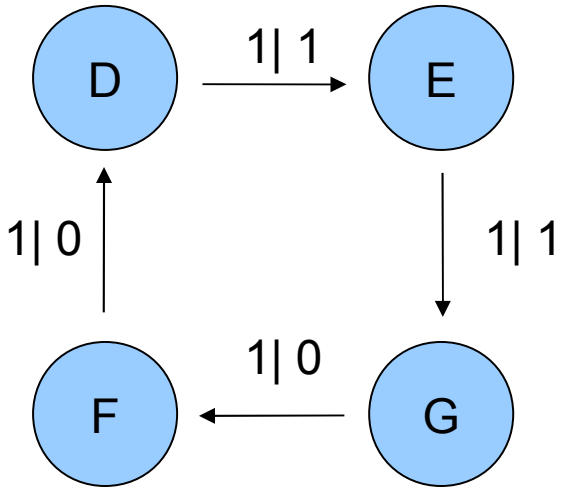
Example Parse Tree Mishap

- We'd need to take at least D = 5 to use L = 2 and remake our morphs without dangling states

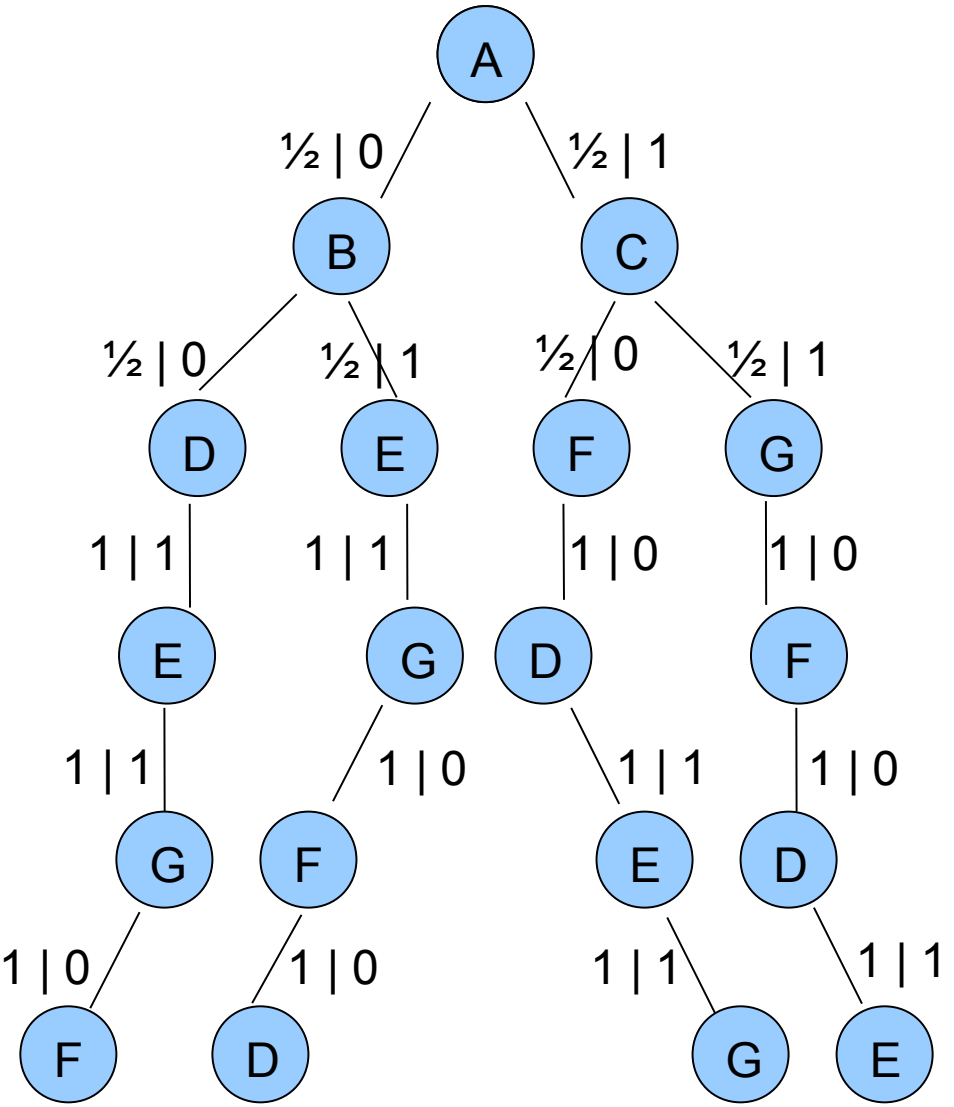


Example Parse Tree Mishap

- Looks like period-4 process



- Can we be sure this is full epsilon machine?
  - Be sure of parse trees in general?



Example Parse Tree Mishap

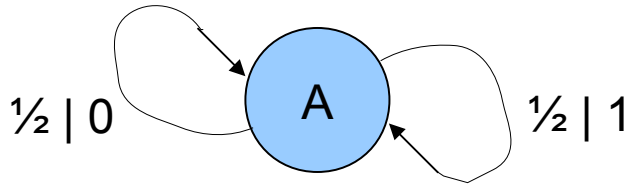
- Parse Trees only guaranteed to predict future length up to length L given past up to length D-L

- Work great if we expect low determining future length and past length

- Longer scale fluctuations?
  - Non-stationary processes?

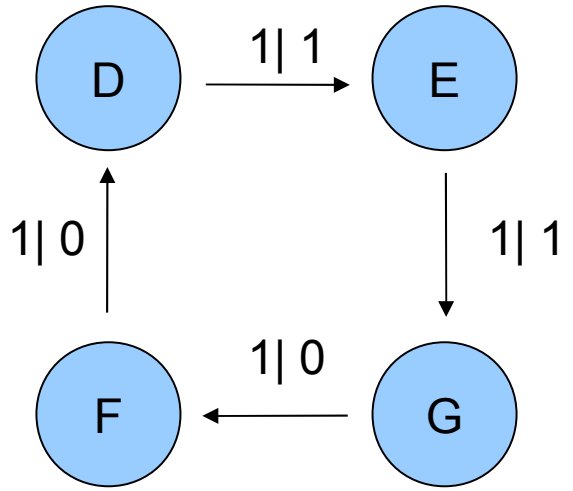
D = 2
L = 1

Best predictor of next symbol given past symbol



D = 5
L = 2

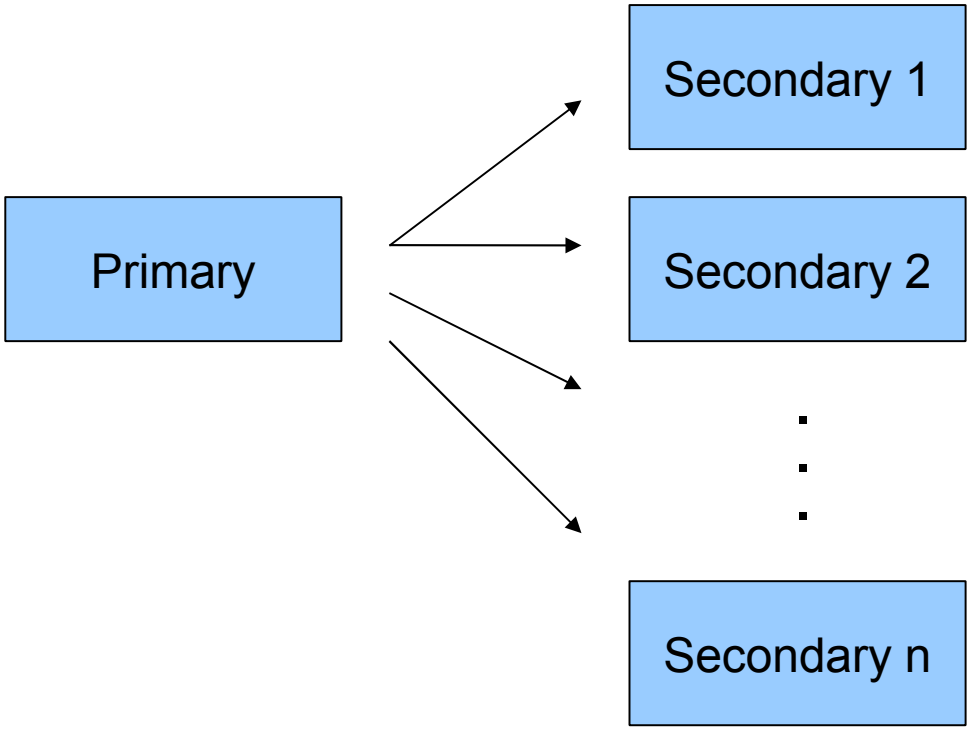
Best predictor of next 2 symbols given past up to length 3



The Composite Machine

• Composite Machine: Primary + Secondary Processes

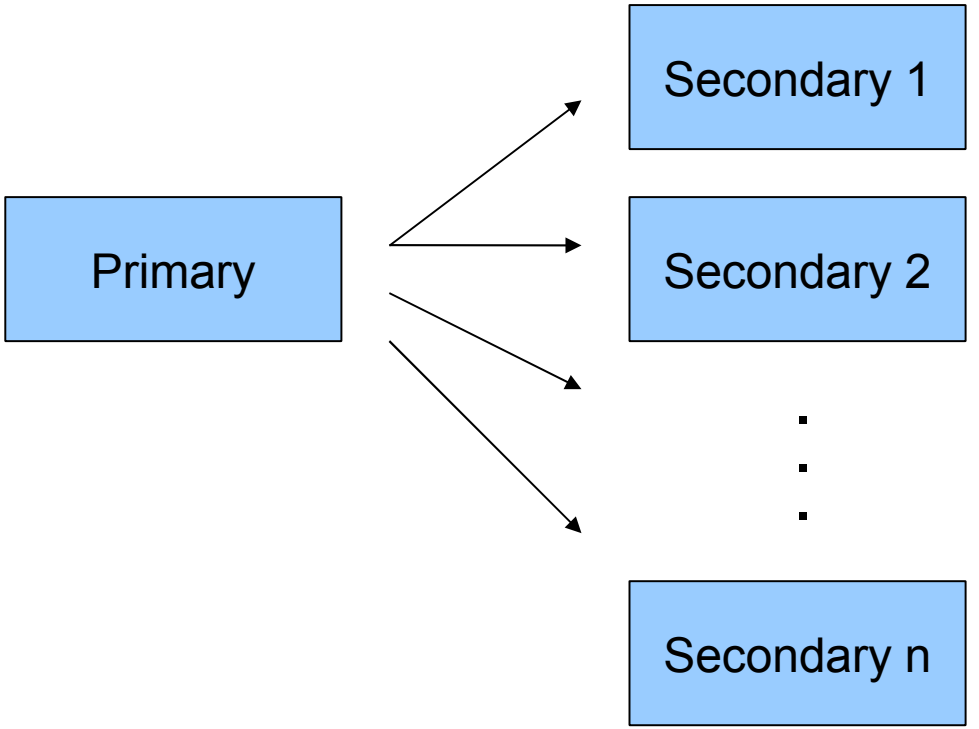
- Primary process output determines secondary process to be activated
- The active secondary process directly determines visible output string for observer
- Primary process progresses no faster than secondary processes
  - e.g. primary process progresses every five secondary output symbols



The Composite Machine

• Easy to generate data from these machines

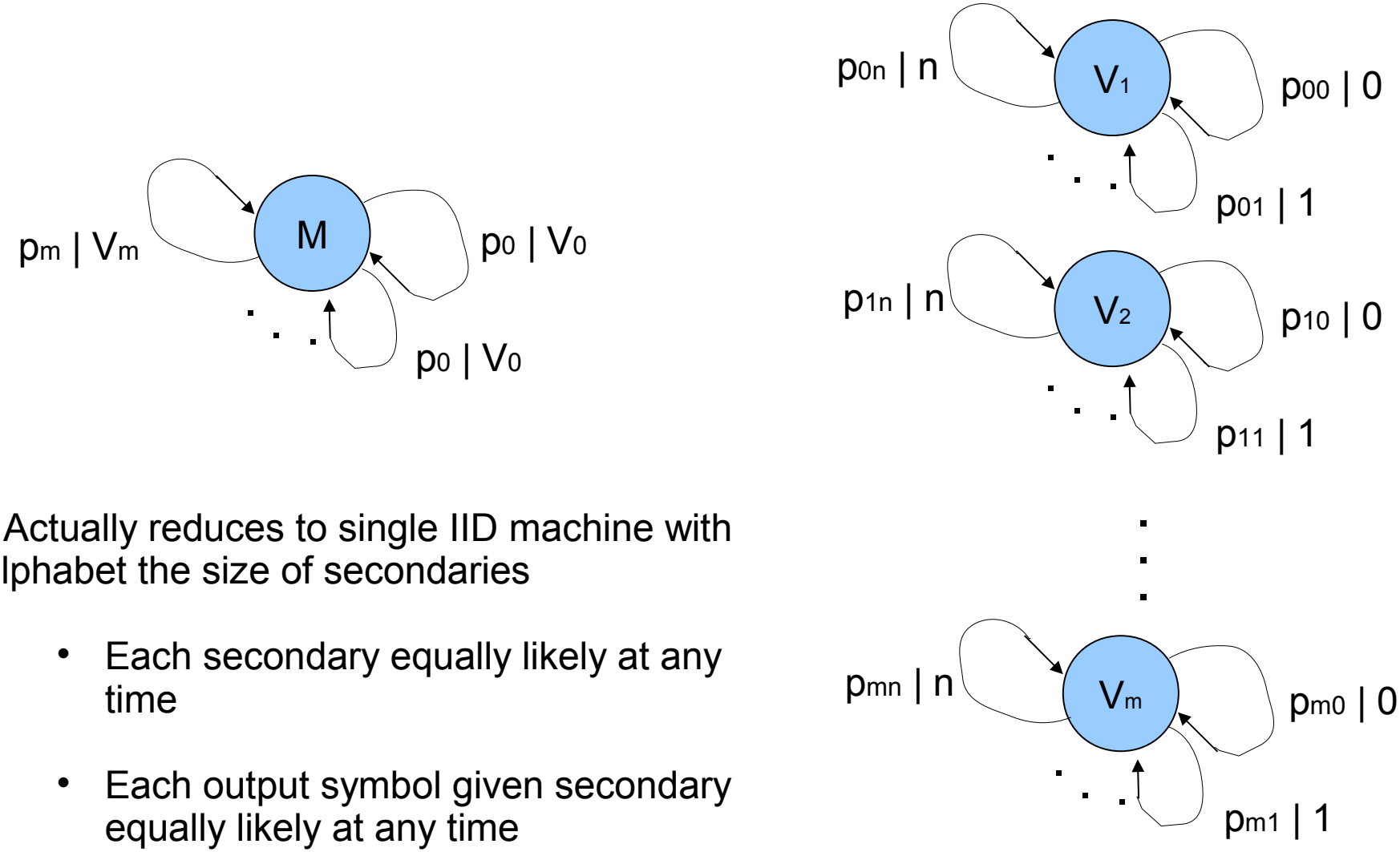
- Often poorly reconstructed with pure parse tree analysis
- Learning to deal with these machines can expand our epsilon machine reconstruction capability



Simple Composite Systems

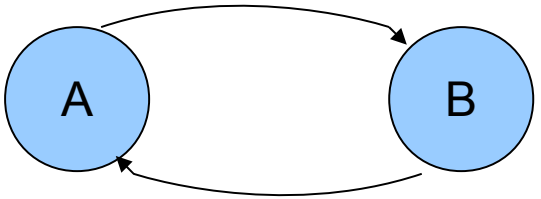
- Primary progresses with every secondary output symbol
    - IID Primary with IID Secondaries
    - Periodic Primary with IID Secondaries
        - 2 Secondaries
        - 3 Secondaries
- Primary progresses with every few secondary symbols or so
- Primary progresses much slower than secondary processes
    - Main focus of my project

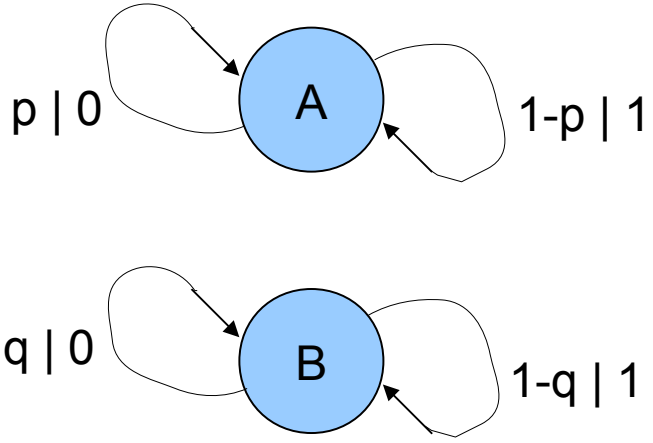
IID Primary: IID Secondary



- Actually reduces to single IID machine with alphabet the size of secondaries
  - Each secondary equally likely at any time
  - Each output symbol given secondary equally likely at any time

Periodic Primary: 2 IID Secondaries

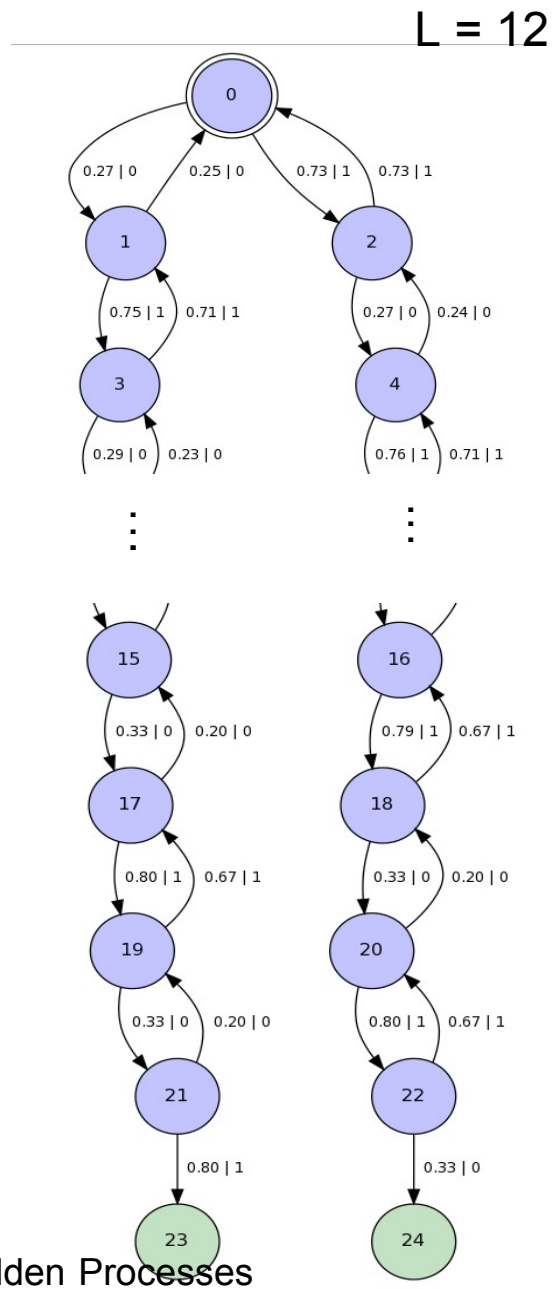




- Easy to understand and generate data for

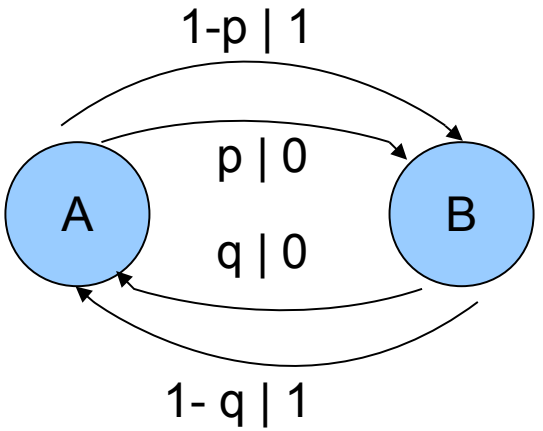
MSP of Periodic Primary: 2 IID Secondaries

- Consider p = 1/3, q = 1/5
- Not too hard to understand, but not as simple as original conception
  - Transient state chain converges onto actual secondary states, alternates appropriately
- Infinite MSP:
  - No morph length fully captures it

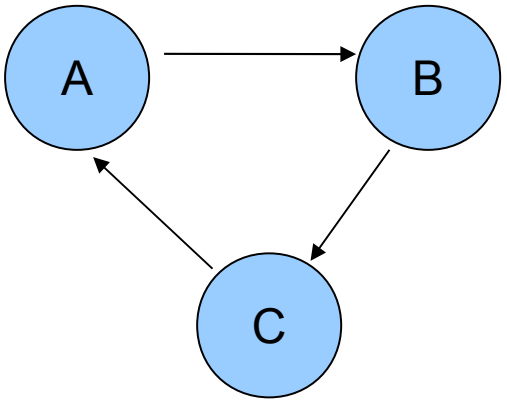


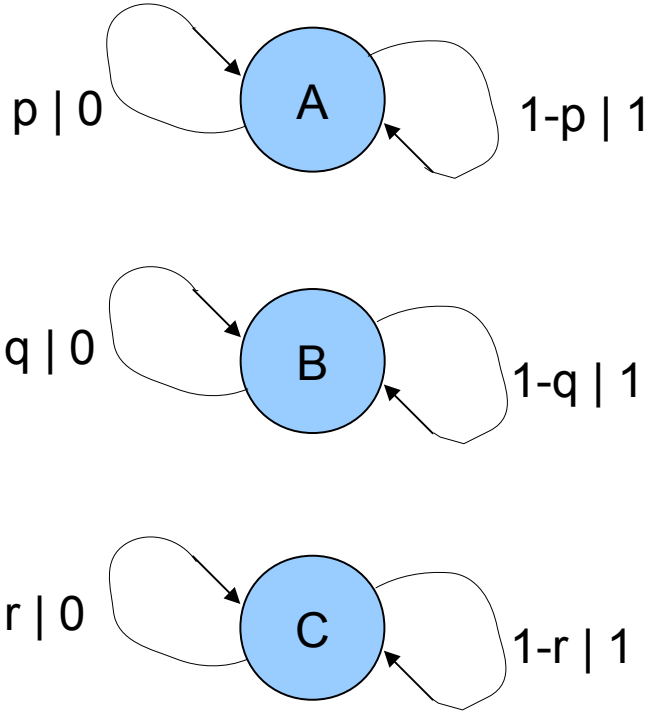
Periodic Primary: 2 IID Secondaries

- Could simply reconstruct by considering every other symbol
  - Submachine for even symbols, submachine for odd symbols
    - Parse Tree each submachine
  - Connect submachines appropriately



Periodic Primary: 3 IID Secondaries

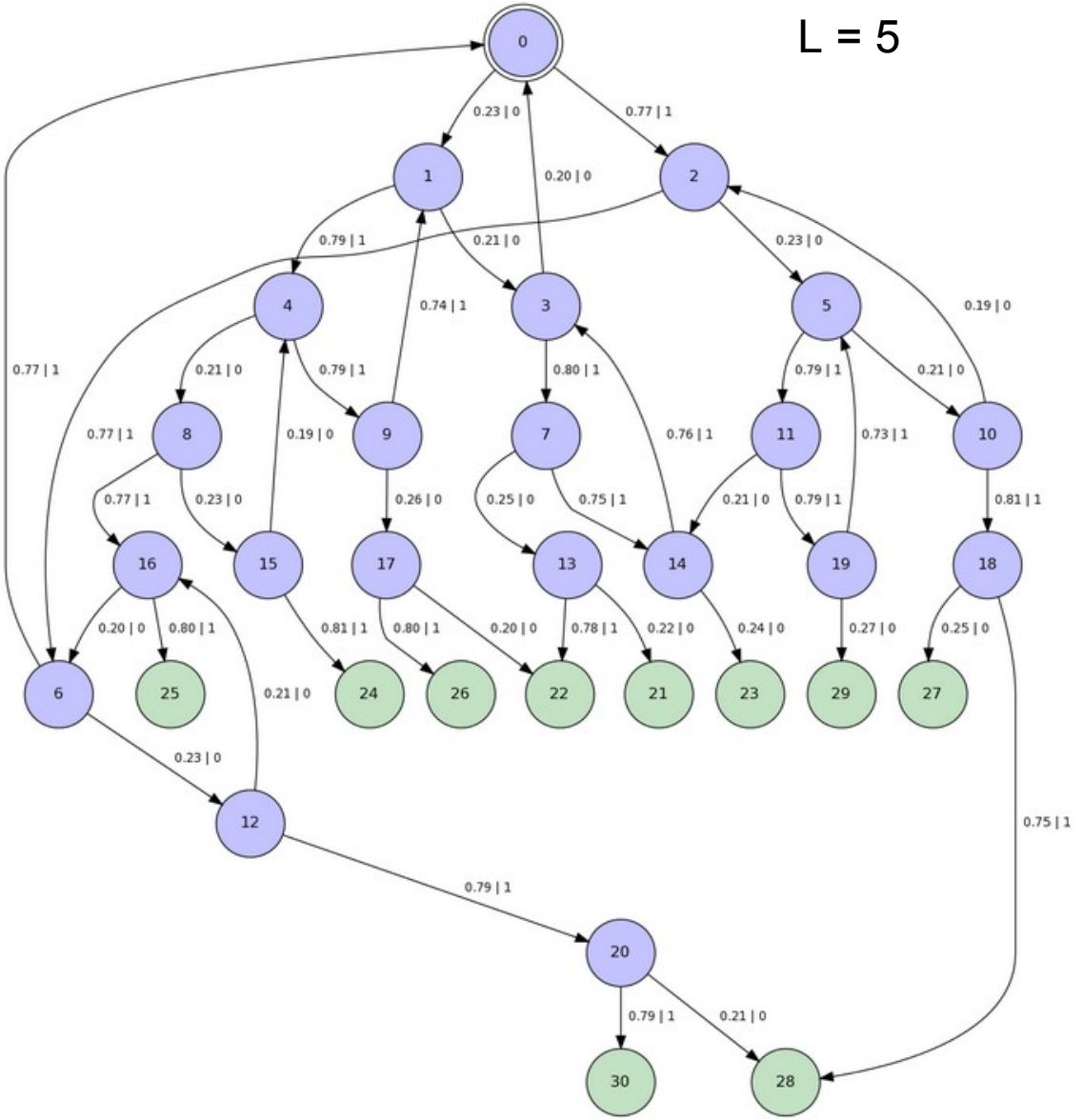




- Also easy to understand and generate data for

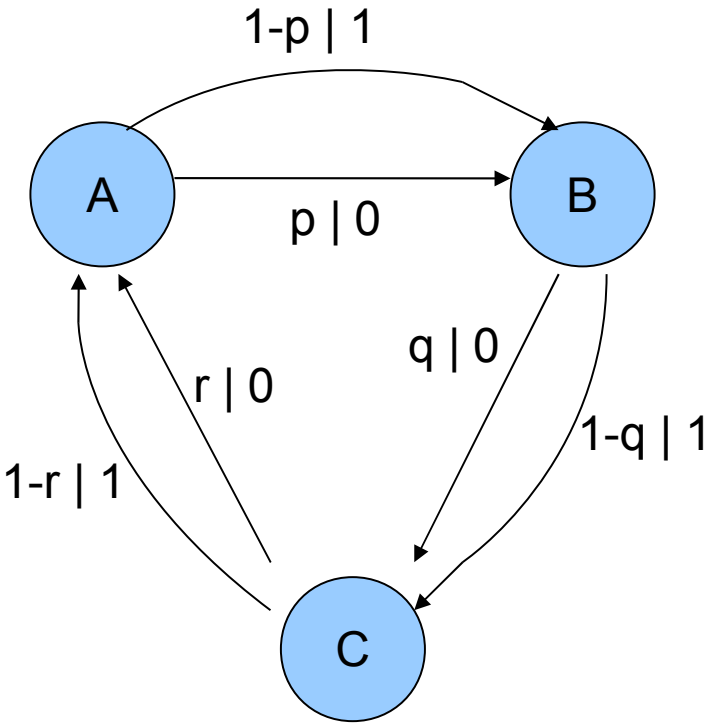
MSP of Periodic Primary: 3 IID Secondaries

- Consider p = 1/5, q = 1/7, r = 1/3
- Gets quite messy
- Computationally demanding with larger L
- But still has transient states that converge to secondary states with high L
  - Cycle properly too (ish)
- Not quite satisfying
  - More secondaries and higher secondary alphabet size gets even worse



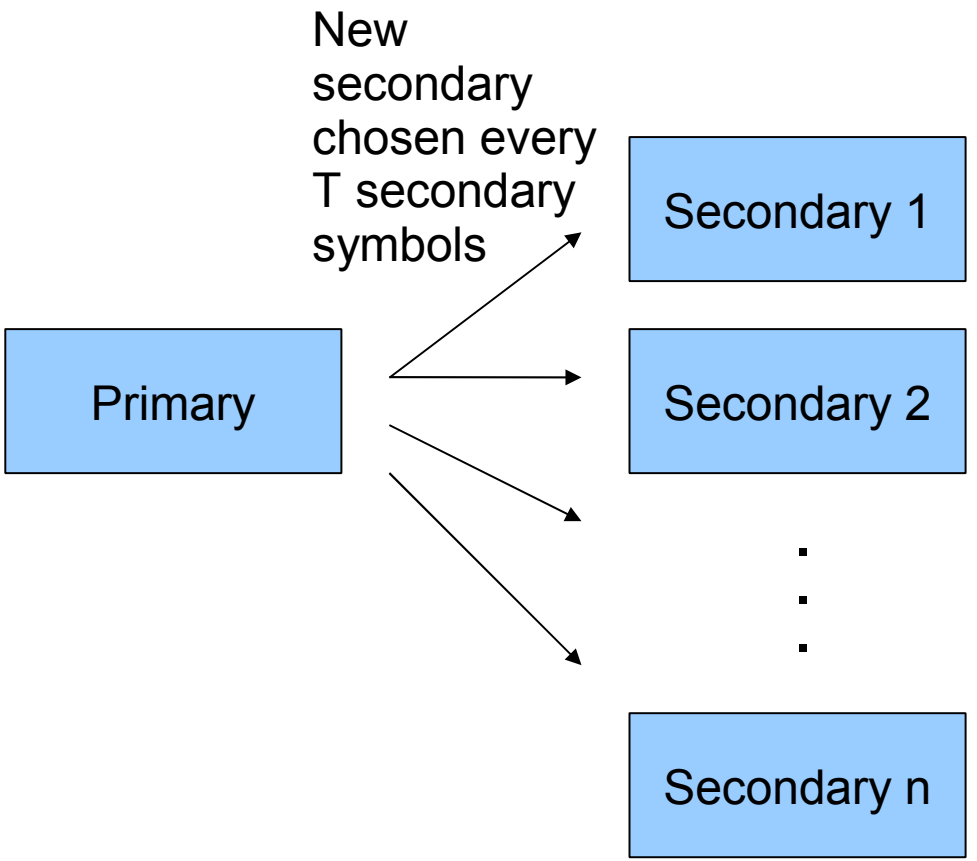
Periodic Primary: 3 IID Secondaries

- Again, could simply reconstruct by considering symbols three apart
  - Submachine for 0 mod 3, 1 mod 3, and 2 mod 3 timed symbols
    - Parse Tree each submachine
  - Connect submachines appropriately



Slower Primary: IID Secondaries

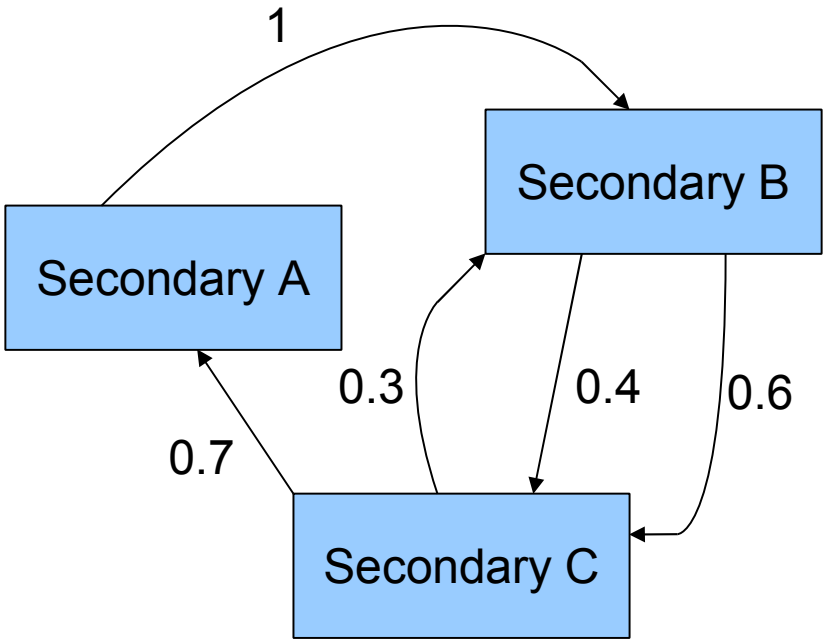
- Primary progresses after some number of secondary output symbols
- Could maybe handle if secondaries were IID
- Otherwise limited time in each secondary process hinders ability to reconstruct
- Not obvious how to properly reconstruct whole system



Very Slow Primary with Secondaries

• If primary progresses far slower than rate of secondary output symbols

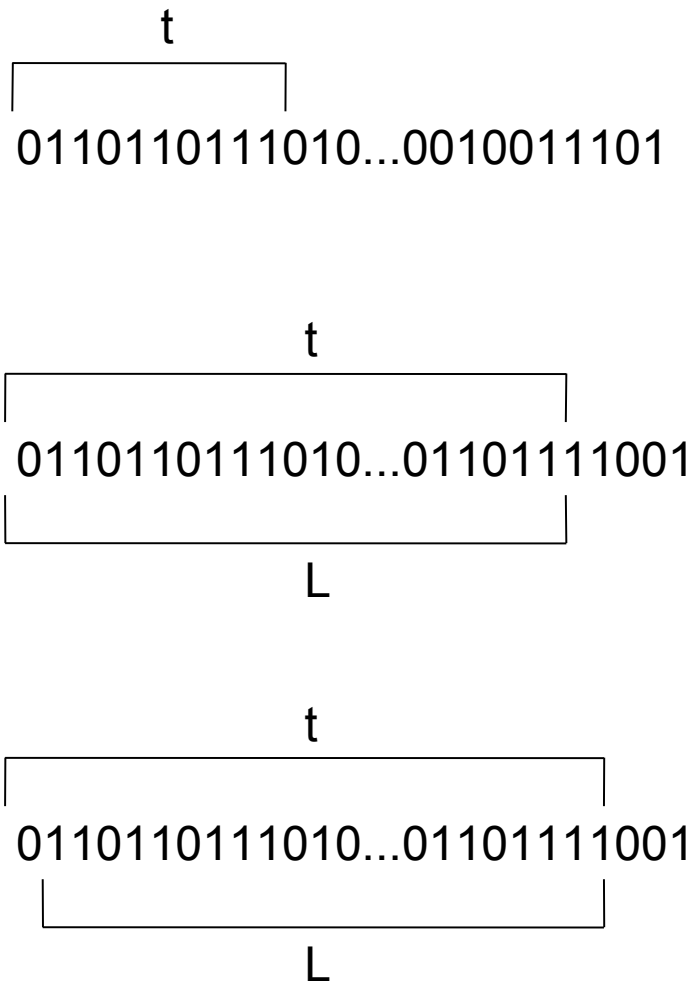
- Lots of output symbols before active secondary changes
- Can accurately reconstruct current secondary process



Example Markov Primary Process: Secondaries output 1000 symbols before primary switches them

Reconstructing Very Slow Primary with IID Secondaries

- Consider a slow primary with IID secondaries
- Start by generating probability distribution of process from starting symbols
- Once L symbols are input, build probability distribution of past L symbols
- If probability distributions match (they will at this iteration) continue
- Add next symbol to base probability distribution and replace second distribution with distribution over last L symbols
- Continue like this until the two distributions don't agree within a decided error



Reconstructing Very Slow Primary with IID Secondaries

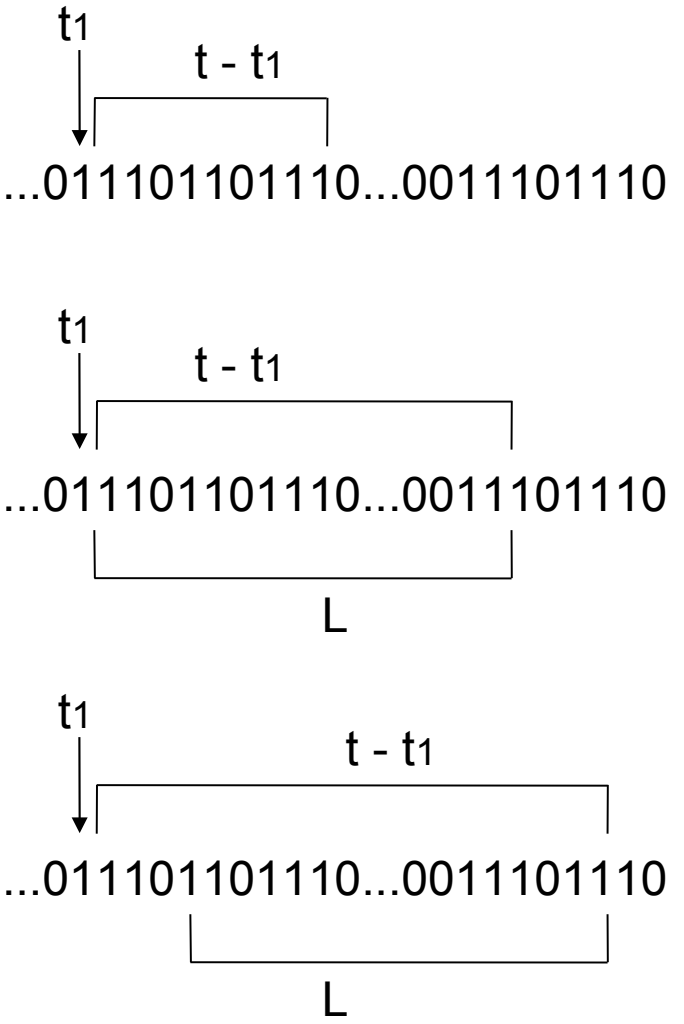
- When two distributions disagree, set aside last t symbol distribution as first slave process and note the time, t_1

- Continue anew starting at current symbol, t_1

- When distribution starting from t_1 disagrees with past L symbol distribution, note time as t_2

- Save new base distribution (starting at t_1) as new secondary process if it disagrees with the first secondary process

- If they agree, combine first secondary process with new base distribution and save as first secondary



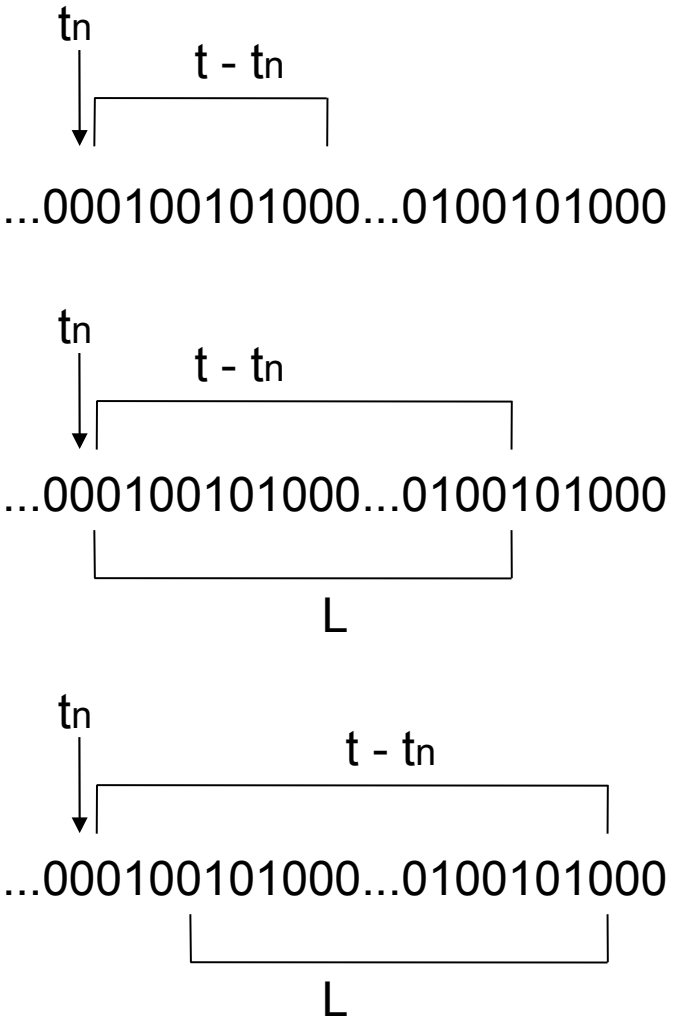
Reconstructing Very Slow Primary with IID Secondaries

- Continue like this, generating new base distributions and past L symbol distributions and comparing until they disagree

- Make sure to check against all saved secondary distributions before saving a new one and combining them if appropriate

- Should wind up with a decent approximation of the secondary processes with large enough input stream

- Can build parse tree out of transitions between secondary processes to get primary process



Wrap-up

- Simple tree-parsing doesn't always capture the full process even if consistent L-morphs are found

- Composite Systems provide a challenge for the simple parse-tree method

    - Conquering them can expand our epsilon-machine re-construction repertoire

- Main focus of my project: instantiate method of reconstructing primary and secondary processes from very slow primary with IID secondaries composite process

    - Potentially even consider more complicated secondary processes