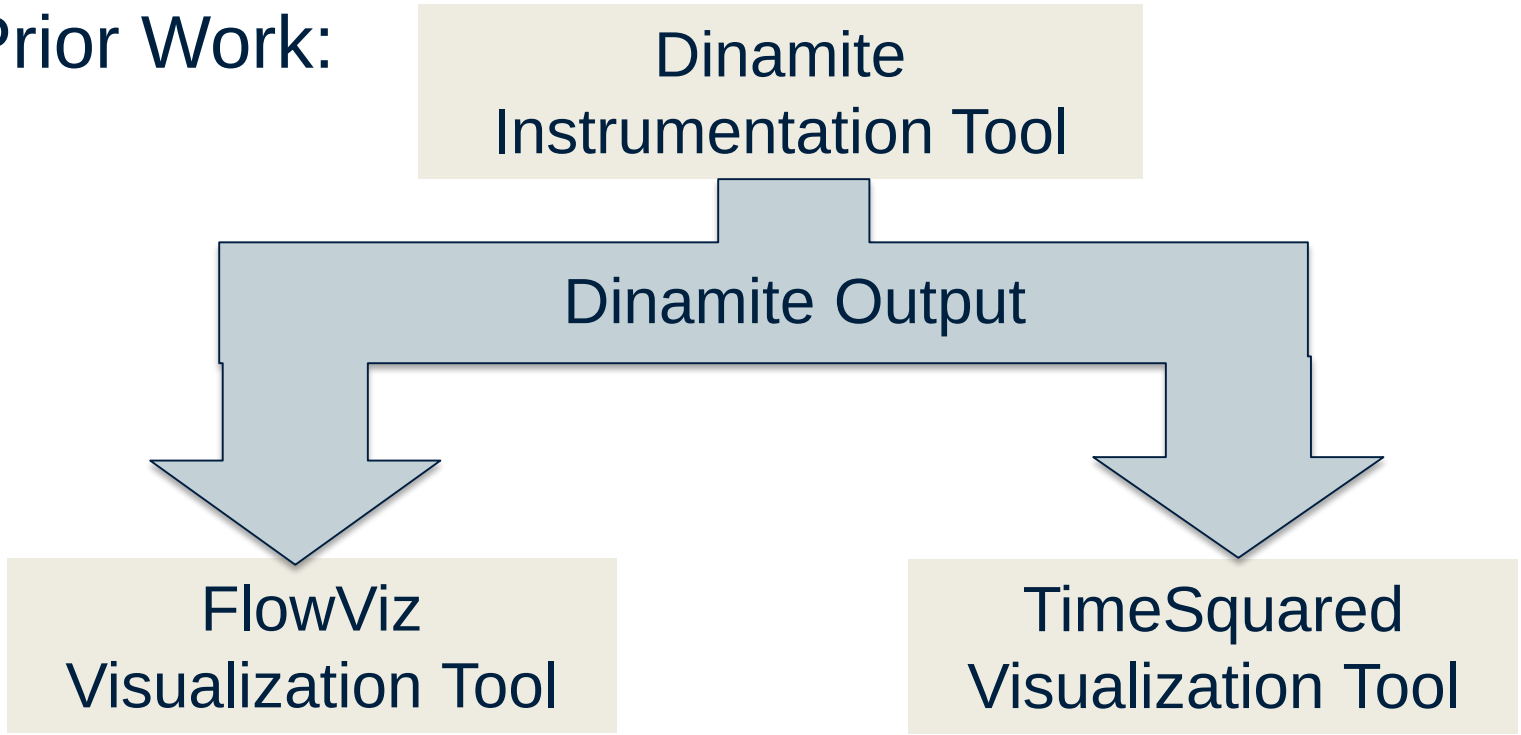
ThreadViewer: Visualizing Thread Behavior in a Program Execution

Augustine Wong

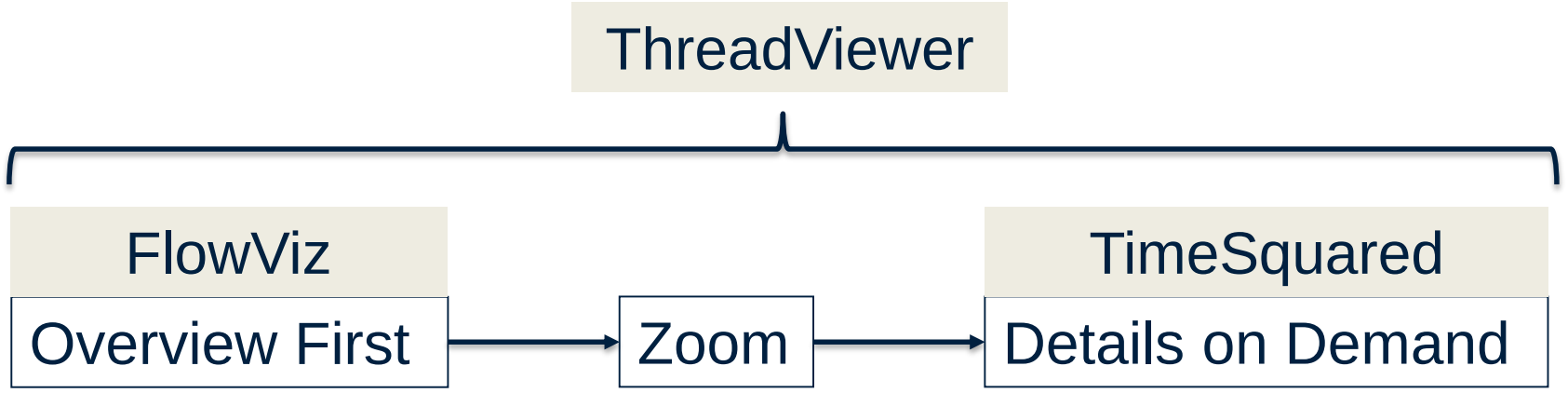
Project Background

- Goal of my research group: create software performance debugging tools
- Prior Work:



What is ThreadViewer?

- FlowViz and TimeSquared should be used together but were designed separately
- ThreadViewer is a visualization tool which integrates FlowViz and TimeSquared together

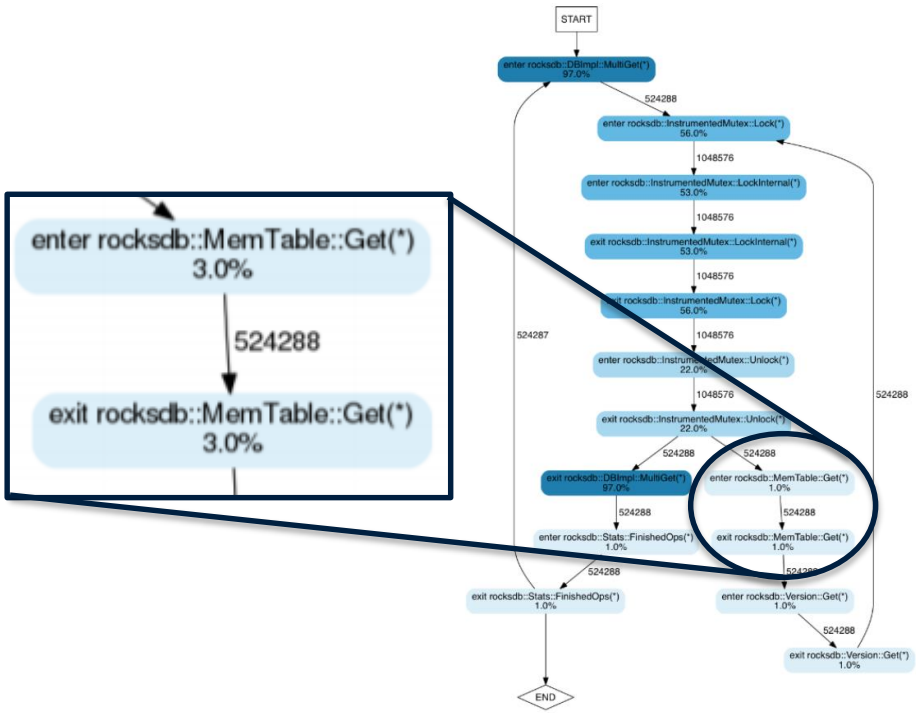


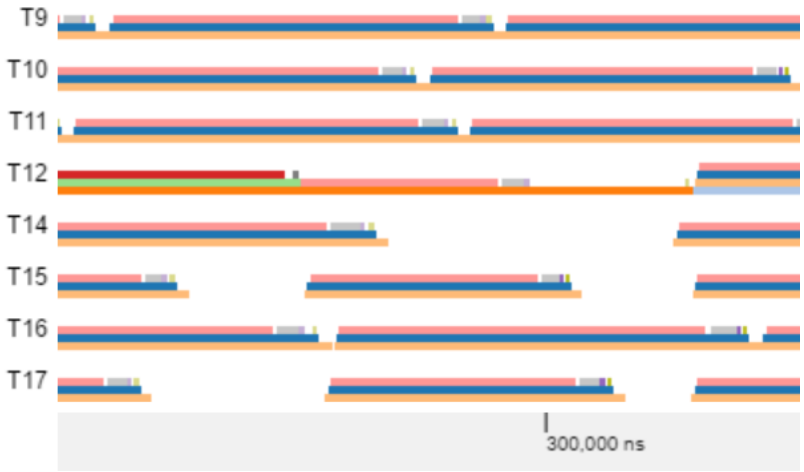
Data Abstraction of a Dinamite Trace

- Whenever a thread calls a function, Dinamite generates two records: function entry and function exit

Attribute	Type	Description	
Function	Categorical	Name of the function	✓
Direction	Categorical	Indicates if record is function entry or exit	✓
Thread ID	Categorical	Identifies the thread	✗
Time	Ordered, quantitative	Time when the record was made	✓

FlowViz and TimeSquared





FlowViz

State diagram of a thread's behavior, where a state is a function entry or exit

TimeSquared

Displays each function call in a horizontal timeline

Dataset Cardinality

- Project dataset is a Dinamite output capturing 22 seconds of one thread's activity in WiredTiger:
  - ~11 million function entry and exit records
  - 20 function attribute levels
  - Time attribute has ns resolution
- Challenge: Reduce data cardinality while keeping time attribute

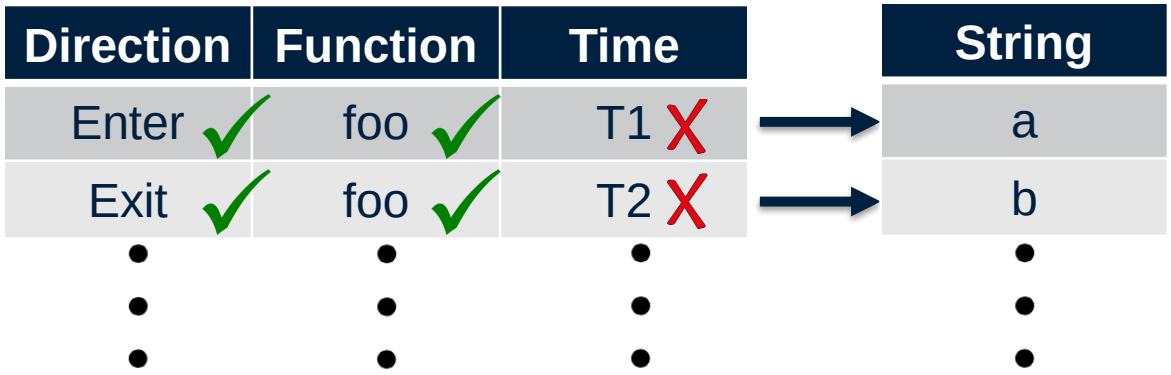
Summarize Dataset With Execution Patterns

- Threads tend to execute same sequences of functions repeatedly:
  - Thread repeatedly trying to acquire a lock
  - Thread repeatedly evicting memory pages

- Reduce dataset cardinality by finding execution patterns – sequences of function entries and exits which occur repeatedly throughout a Dinamite trace

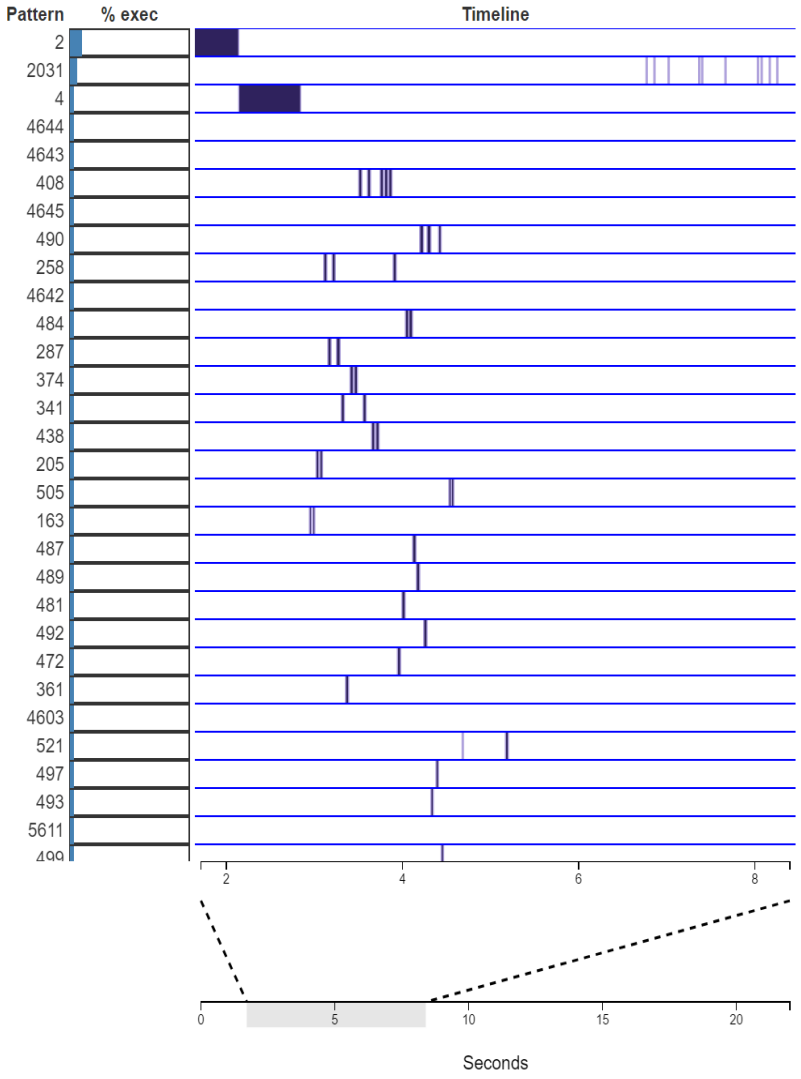
Finding Execution Patterns with Sequitur

- Walkinshaw et al proposed finding execution patterns using Sequitur algorithm
  - Treat Dinamite output as a string input to Sequitur
  - Found ~7K patterns
  - Wrote Python script to parse through original dataset to find the time intervals that the patterns occurred (add back time attribute)



[Walkinshaw, Neil, Sheeva Afshan, and Phil McMinn. "Using compression algorithms to support the comprehension of program traces." *Proceedings of the Eighth International Workshop on Dynamic Analysis*. ACM, 2010.]

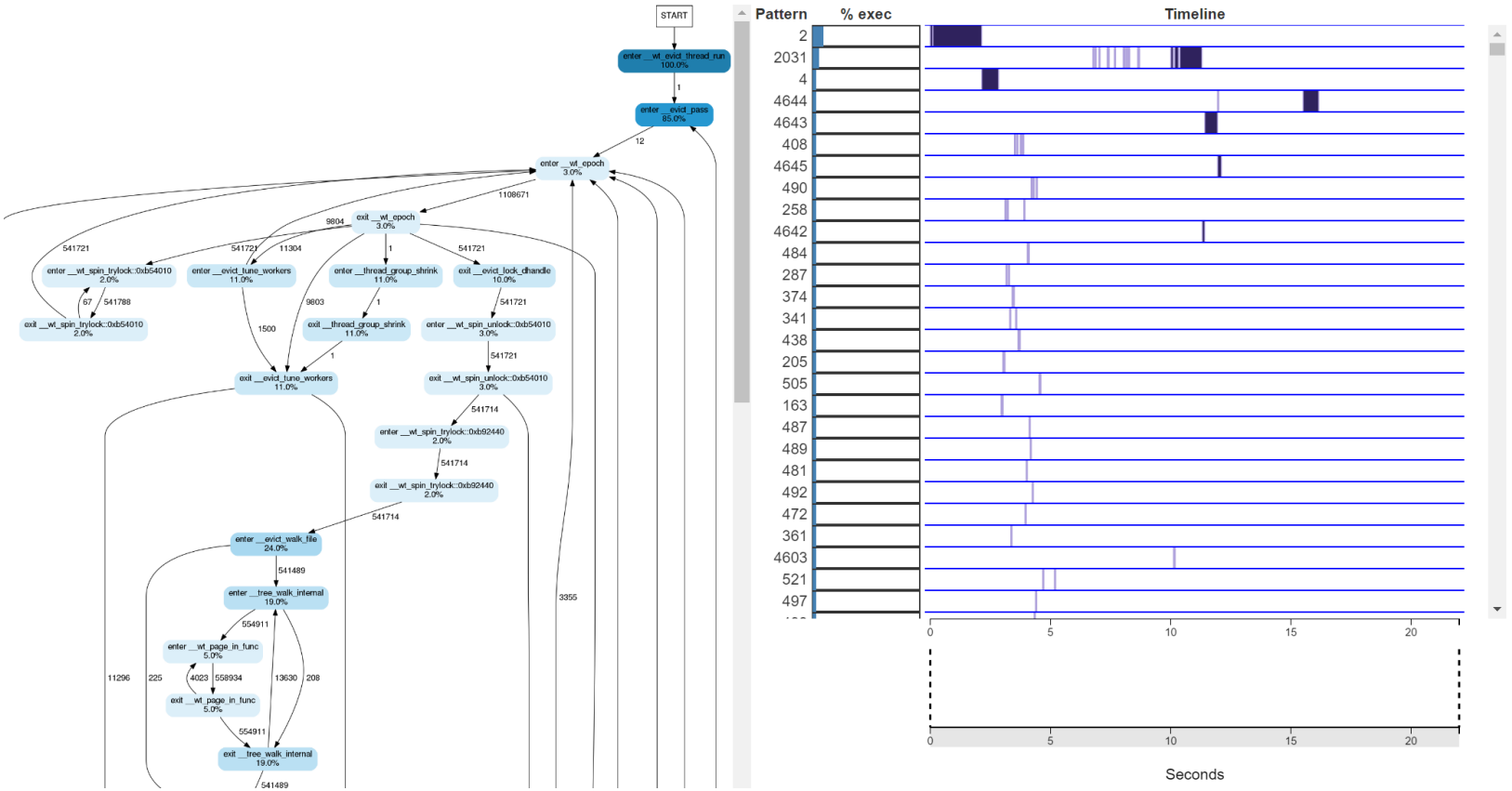
Showing Patterns on a Timeline



- Vertical bar graph to show % thread runtime for each pattern
- Each pattern's timeline aligned with its bar in the bar graph
- Navigate timeline via link navigation

- Where's FlowViz?
- How do we see the contents of the patterns?

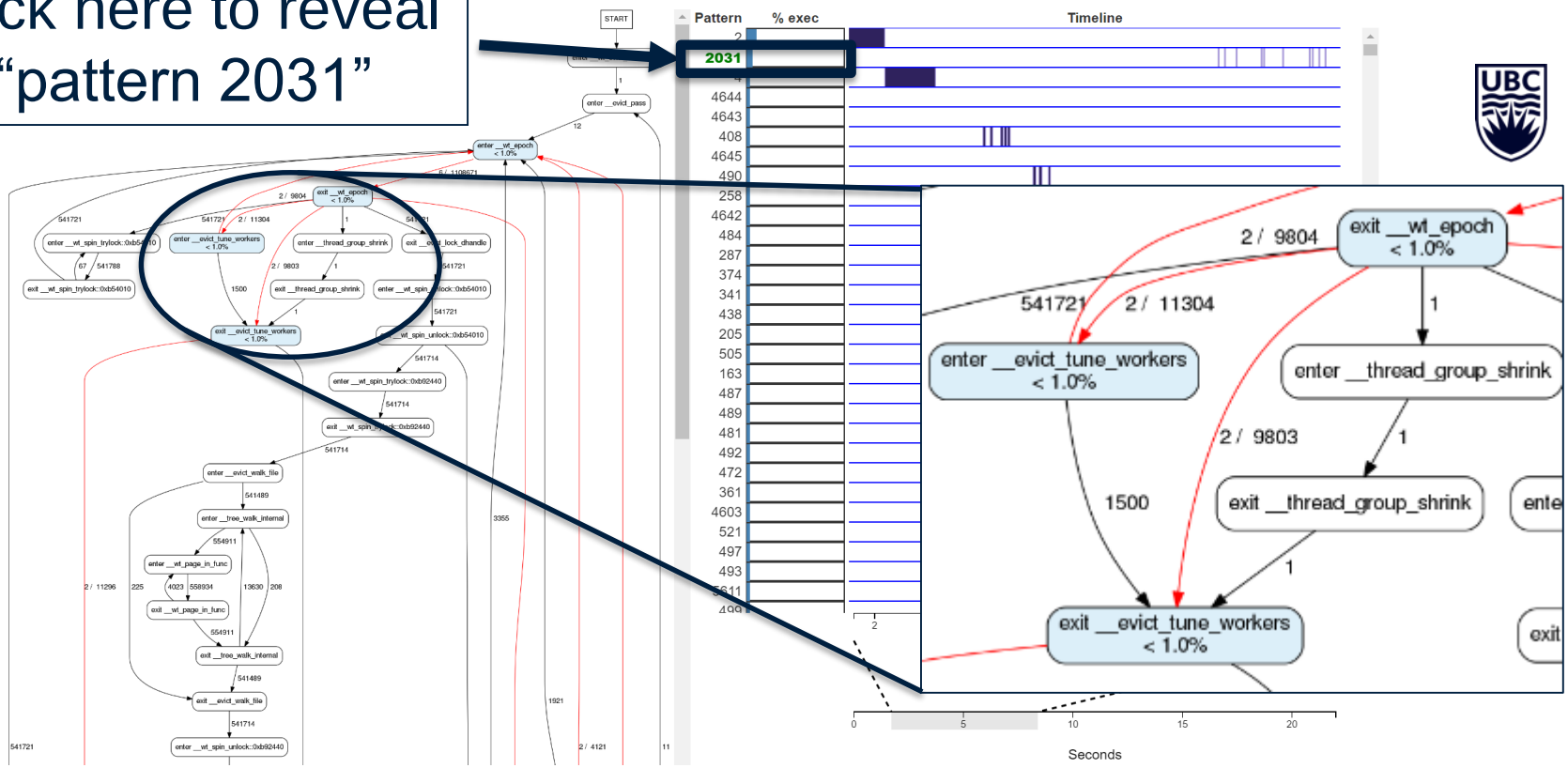
FlowViz in ThreadViewer



- ThreadViewer divided into two panels:
  - Left panel shows FlowViz state diagram
  - Right panel contains bar graph and timeline
- Still can't see contents of the patterns…

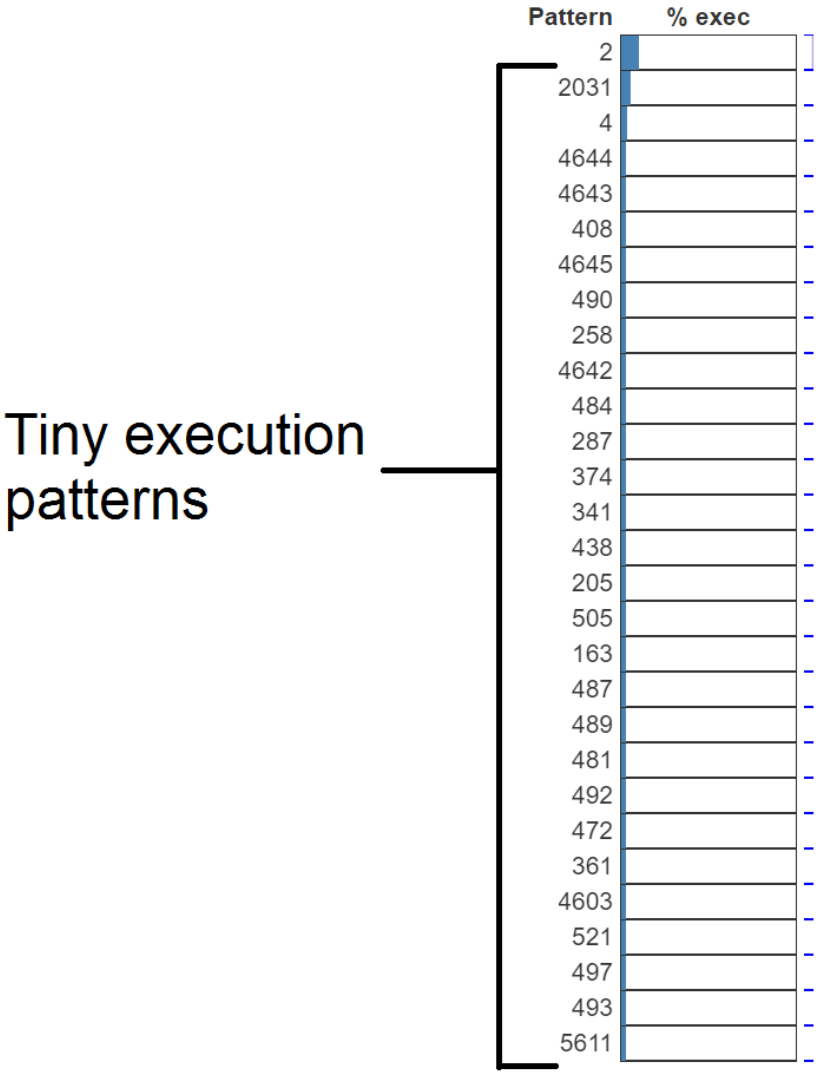
Patterns as FlowViz Subgraphs

Click here to reveal "pattern 2031"



- Link highlighting reveals pattern as subset of the state diagram on the left panel
- Colored FlowViz nodes represent states in pattern
- Transitions between states in pattern shown in red

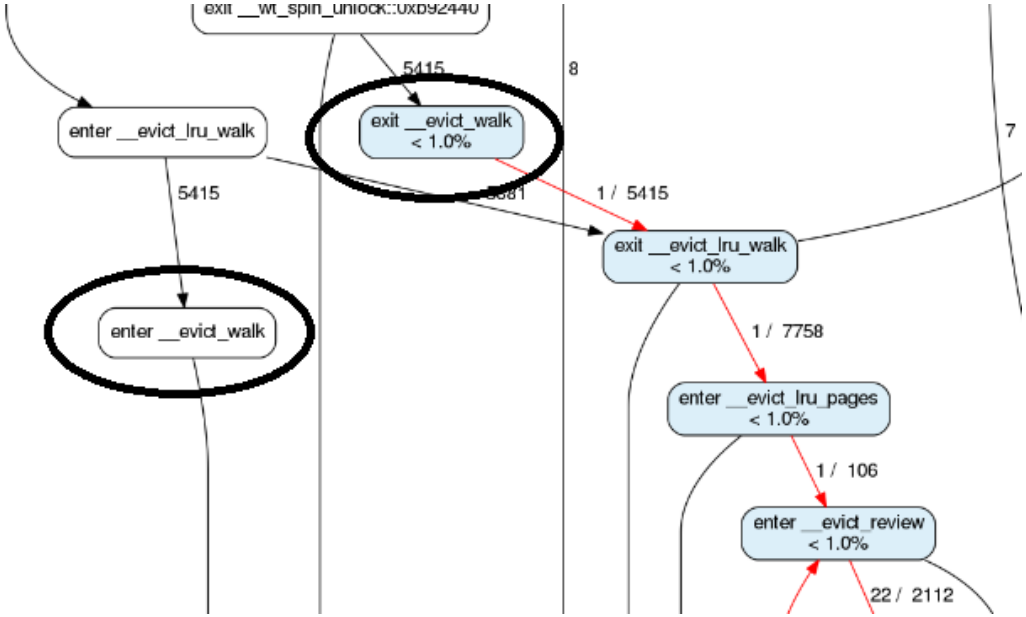
Discovery with ThreadViewer: Low % Execution Times



- Discovered that the vast majority of patterns take up < 1% of a thread's runtime (between ~0.001% and 10%)

Discovery with Thread Viewer: Confusing Execution Patterns

- Some patterns do not have complete pairs of function entries and exits
  - Pattern below shows thread exiting function __evict_walk but not entering it



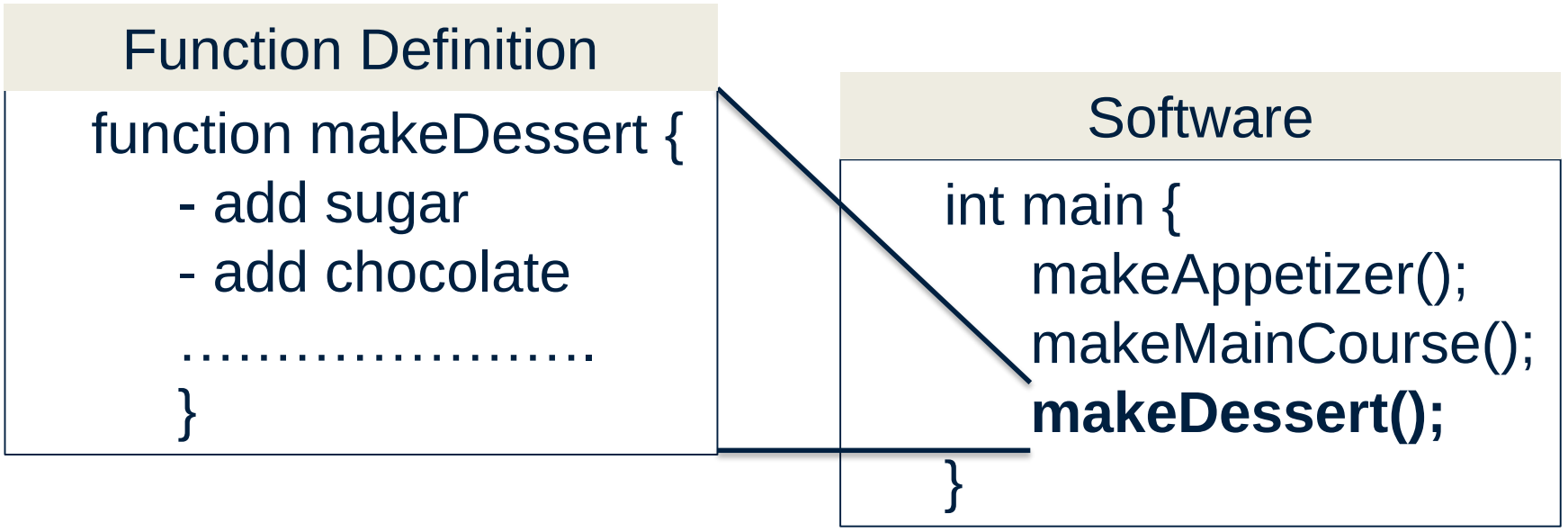
Future Work

- Address the pattern detection problems discovered by ThreadViewer
  - Update Sequitur?
  - Create new algorithm?
- Good news: we now have a visualization tool to evaluate the quality of our pattern detection strategies

THANK YOU!

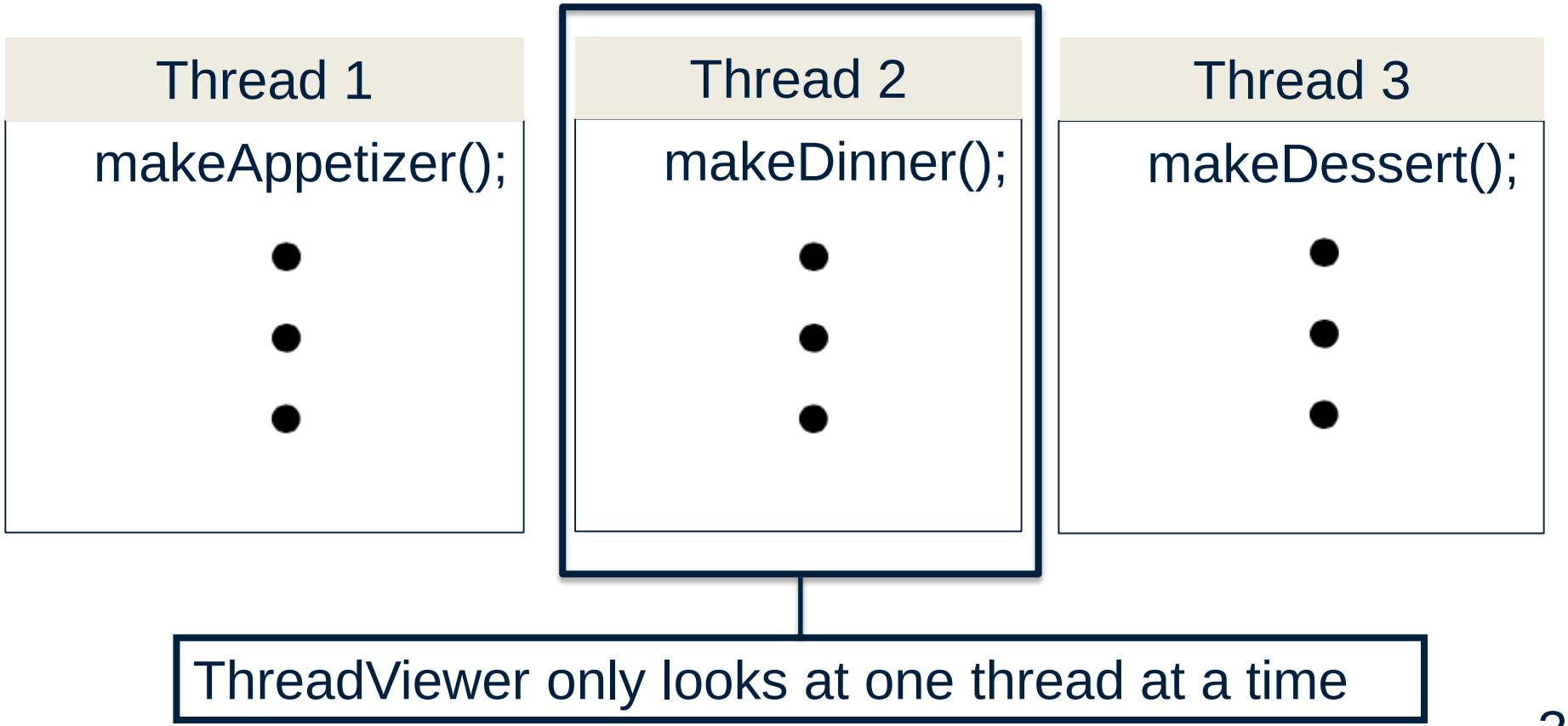
What is a Function?

- Software is comprised of functions:
  - Functions are a list of instructions which make up a specific task



What is a Thread?

- Multi-threading boosts software performance by executing sequences of functions in parallel



Sequitur Output