Galois Sync-Up

Thomas Gilray & Kris Micinski
(Task 1, POLYMORPH V-SPELLS)
April 20, 2022

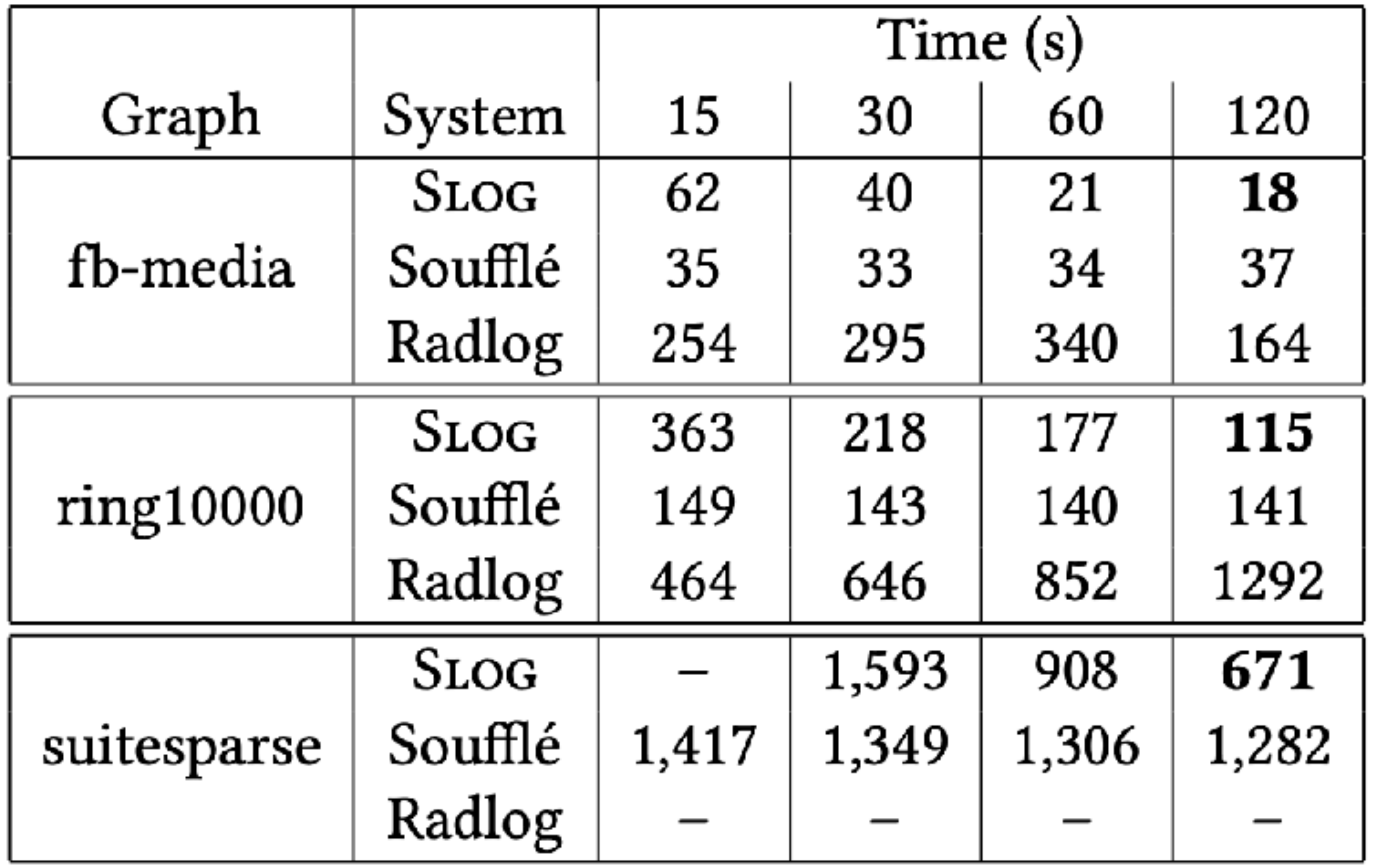
High-Level Updates

- Massively-parallel deductive inference for program analysis
  - **Slog**: structured Datalog via parallel relational algebra
- Apropos Task 1: component identification via static analysis
- Progress: data-parallel binary instruction seq. matching
- Planning next steps
  - Cloud-based data-parallel deductive analytics
  - Scaling Slog to more-fully-featured Datalog

Slog: Massively Parallel Deductive Inference

- State-of-the-art program analysis: Soufflé (Datalog)
  - Compiles to tight `for` loops in C++, efficient datastructures (trie, brie)
  - Almost no parallelism exploited
- Our new engine (V-SPELLS): **Slog**
  - Extends Datalog with *interning*, compiles to parallel RA kernels

Slog scales well—even on one node—but some sequential overhead

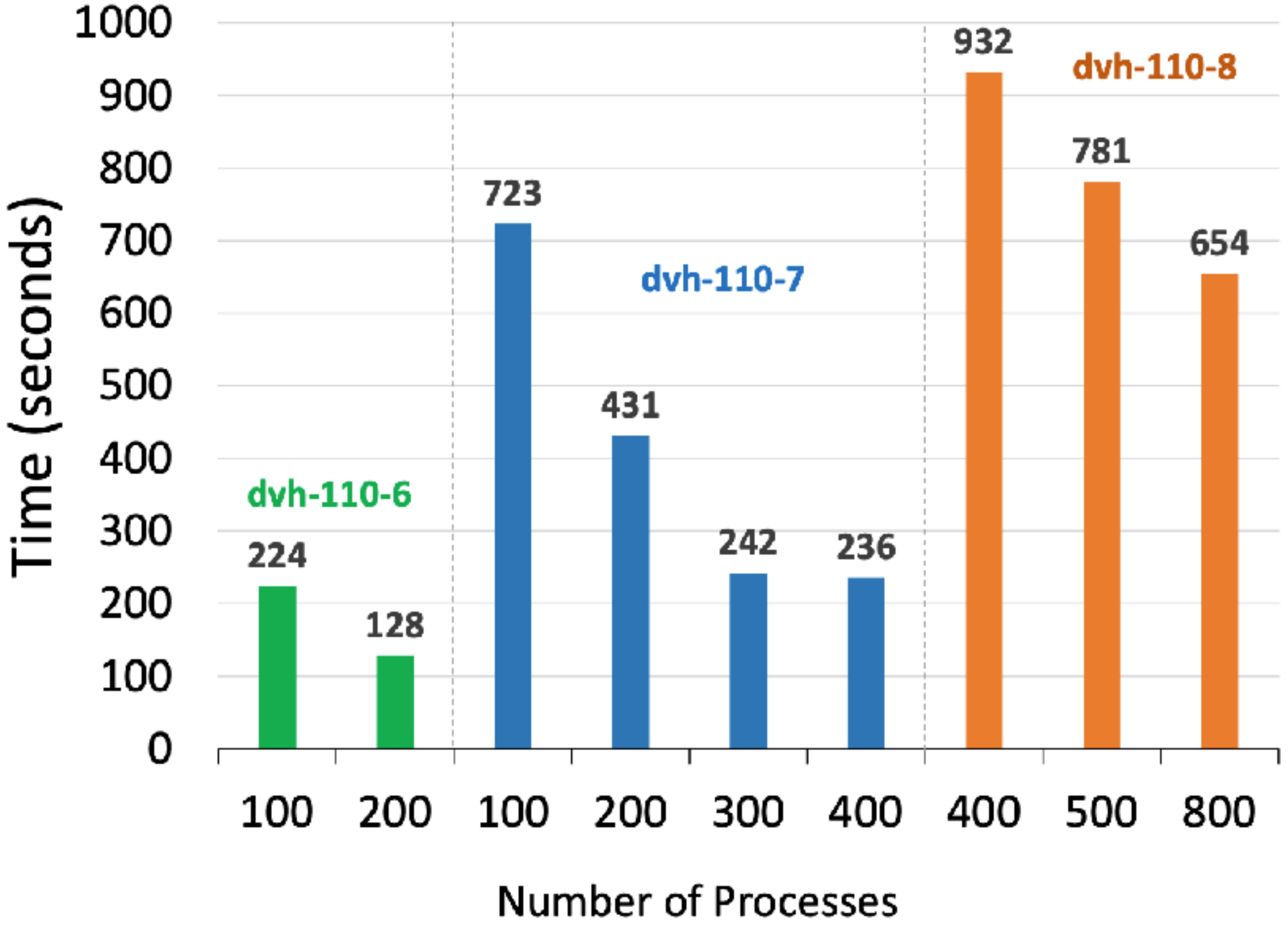


		Time (s)			
Graph	System	15	30	60	120
fb-media	Slog	62	40	21	**18**
	Soufflé	35	33	34	37
	Radlog	254	295	340	164
ring10000	Slog	363	218	177	**115**
	Soufflé	149	143	140	141
	Radlog	464	646	852	1292
suitesparse	Slog	–	1,593	908	**671**
	Soufflé	1,417	1,349	1,306	1,282
	Radlog	–	–	–	–

Transitive Closure Computation
Machine: 128 threads, 20TB RAM, Azure

Strong Scaling on the Theta supercomputer

- Scaled core control-flow analysis for the lambda calculus on the Theta SC
- Promising strong scaling results up to 800 processes
- Communication overhead dominates >1k threads
  - Currently innovating in this direction w/ Sidharth Kumar
  - Algorithmic innovation via two-phase variadic Bruck

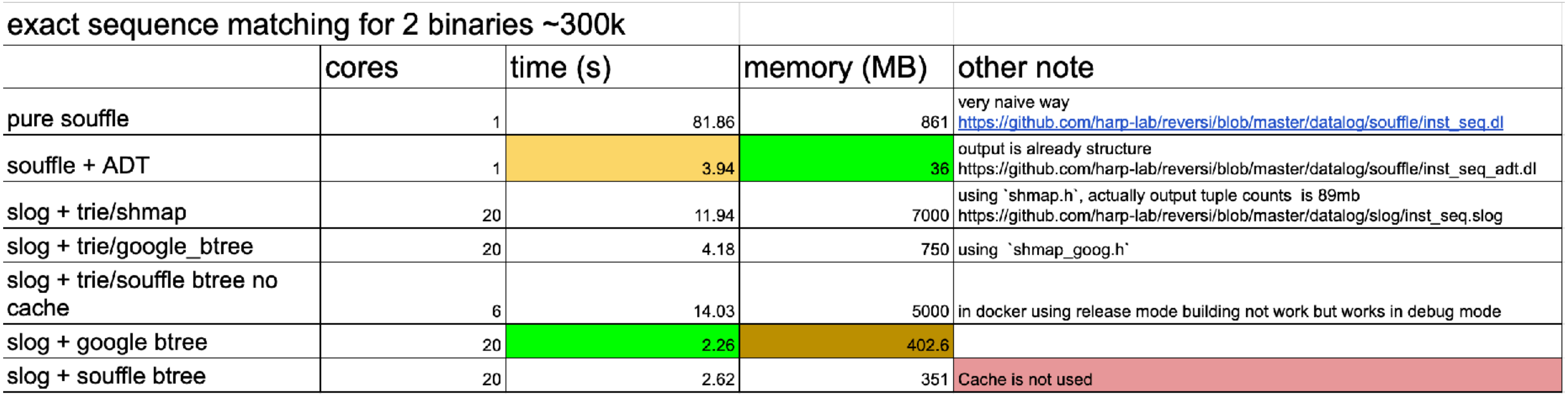


Task: Component Identification

- Slog achieves: parallel semantic-enabled deductive reasoning
- Goal (Task 1): **identify matching components in large codebase**
- Ported fragments of **ddisasm** (Datalog disassembly) to Slog
- Initial goal: identify equivalent instruction sequences in binaries
    - Sequences of 2, 4, 8, …
    -
```
[(inst_seq8 Start1 _ Bin_name1 Ins_tree)
 (inst_seq8 Start2 _ Bin_name2 Ins_tree)
 (=/= Bin_name1 Bin_name2)
 -->
 (exact_same_sequence Bin_name1 Start1 Bin_name2 Start2 Ins_tree)]
```

Instruction Sequence Identification–Results

- Good absolute scaling results vs. soufflé
- k-ary joins in Slog incur higher algorithmic complexity
  - Now working on strategies to mitigate this
- Memory usage in Slog was unnecessarily high
  - Rewrote our data structure to be ~2-3x vs. Souffle’s optimized version



exact sequence matching for 2 binaries ~300k

	cores	time (s)	memory (MB)	other note
pure souffle	1	81.86	861	very naive way https://github.com/harp-lab/reversi/blob/master/datalog/souffle/inst_seq.dl
souffle + ADT	1	3.94	36	output is already structure https://github.com/harp-lab/reversi/blob/master/datalog/souffle/inst_seq_adt.dl
slog + trie/shmap	20	11.94	7000	using \`shmap.h\`, actually output tuple counts is 89mb https://github.com/harp-lab/reversi/blob/master/datalog/slog/inst_seq.slog
slog + trie/google_btree	20	4.18	750	using \`shmap_goog.h\`
slog + trie/souffle btree no cache	6	14.03	5000	in docker using release mode building not work but works in debug mode
slog + google btree	20	2.26	402.6	
slog + souffle btree	20	2.62	351	Cache is not used

Roadblocks / Questions for Galois

- Ordered servers from Dell
    - Chip shortage and staffing issues causing these to be delayed
    - Projecting they will be set up and running by summer
    - Will continue to rent machines from AWS / Azure to benchmark
- Possible questions:
    - Is Galois using Datalog for their performance on Task 1?
    - Potential direction—port cclyzer, to analyze LLVM code via Soufflé

Deductive Inference on the Cloud

- We understand Slog to be the state of the art in parallel deductive inference
- In the future, we expect deductive inference to be performed in the cloud
- Want to measure: deductive inference on AWS via ParallelCloud
    - Dollar-for-dollar: Soufflé vs. Radlog vs. Slog
        - VLDB experiments and benchmark track