Verification of Fine-Grained Concurrent Programs

Christian J. Bell

Verification of Fine-Grained *Concurrent Programs*

- Concurrent programs can be simple
  - threads work independently of each other

- Concurrent programs can be complex
  - use locks, semaphores, CAS, shared stacks, shared queues, etc. to communicate
  - threads follow some protocol

Verification of Fine-Grained Concurrent Programs

- Easiest when we can reason about one thread at a time: local reasoning

- Most powerful when we can reason about all threads at once: global reasoning

Local Reasoning

- Example: Concurrent Separation Logic
- Advantage: modularity
- Disadvantage: cannot reason about many kinds of concurrency

Global Reasoning

- Example: Rely-guarantee
- Disadvantage: not very modular
- Can reason about complex protocols between threads

Examples

{x=v}

```
acquire(l)    ||  acquire(l)
x:= x + n     ||  x:= x + m
release(l)    ||  release(l)
```

{x=v+m+n}

parallel increment

Examples

```
{x>=v}
do
  m:= x
while CAS(x,m,m+n)=0
{x>=v+n}
```

...

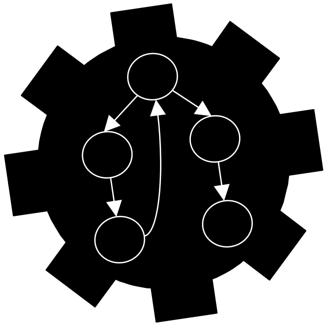
monotonically increasing shared variable

Finding a Balance

- Promising approach is to use a protocol to govern the shared state between threads
  - state machines
  - linear logic
  - “concurroids”
  - concurrent abstract predicates
  - …

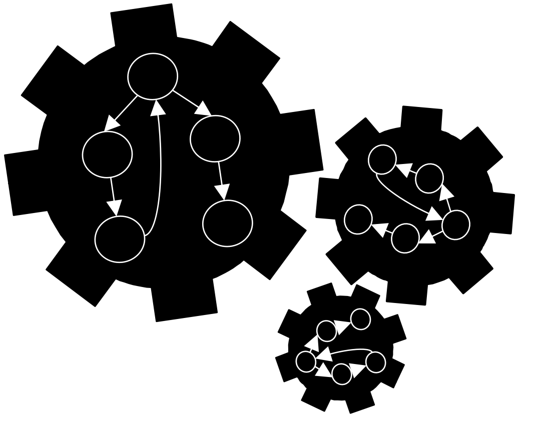
Research Questions

- Representing protocols?



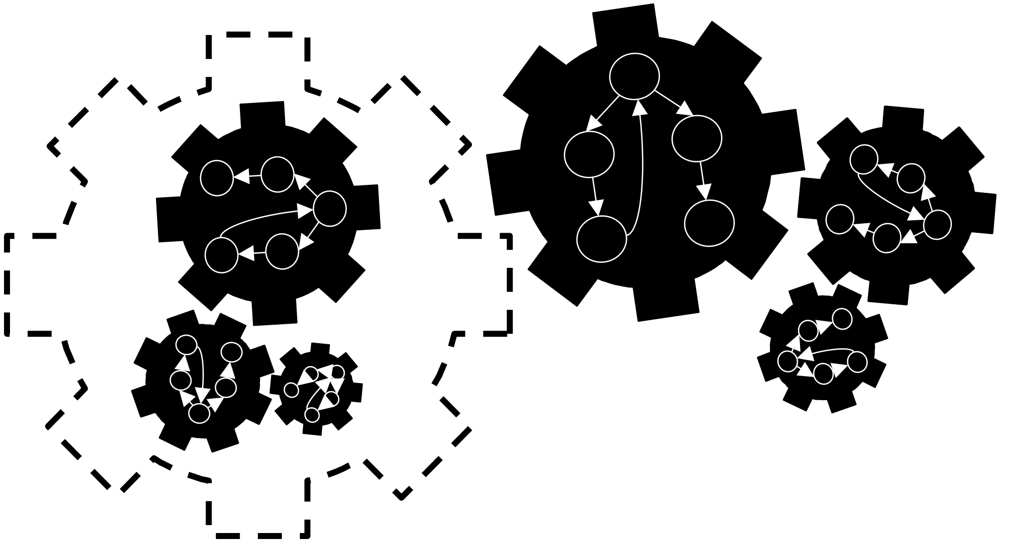
Research Questions

- Representing protocols?
- Composition?



Research Questions

- Representing protocols?
- Composition?
- Encapsulation?



Our Approach

- Formalize our proofs and techniques in a theorem prover *from the start*
  - harness higher-order logic
  - automate ugly technical details away
    - *easy to use in practice* **vs** *looking good on paper*

Our Approach

- Formalize our proofs and techniques in a theorem prover *from the start*
  - harness higher-order logic
  - automate ugly technical details away
    - *easy to use in practice* **vs** *looking good on paper*
- We call our protocols "monitors":
  - they observe the actions of all threads
  - detect "bad" actions
  - and evolve in response to actions

End

Thanks!