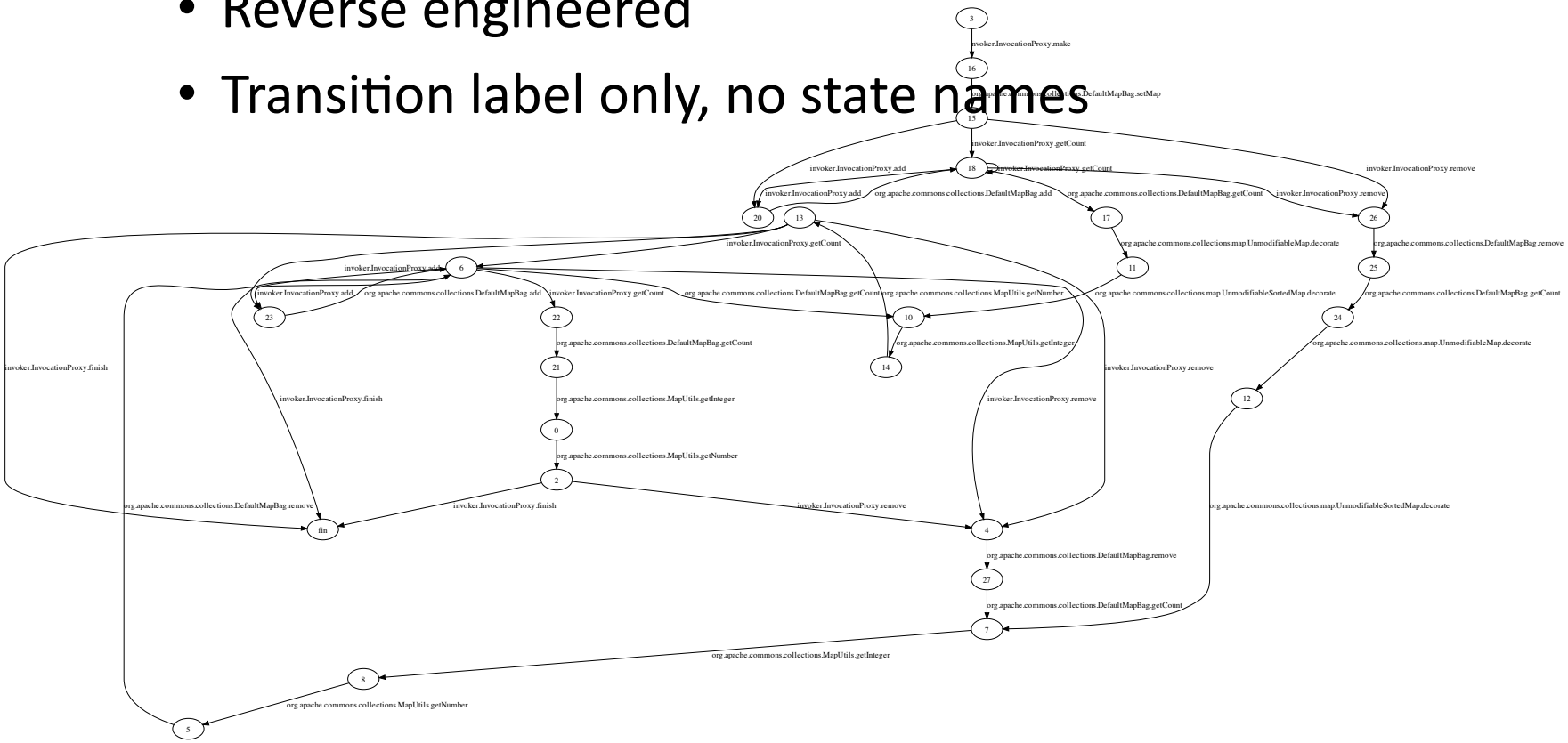
Crunch: Search-based Hierarchy Generation for State Machines

Mathew Hall

University of Sheffield

Background

- State machines
  - Reverse engineered
  - Transition label only, no state names



State Machines

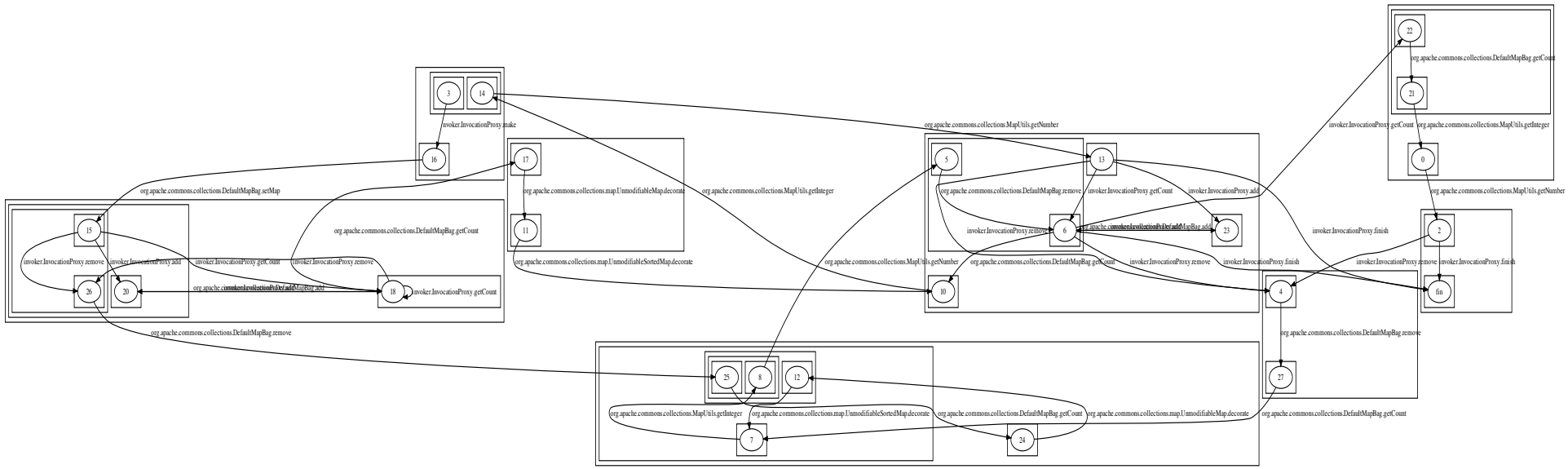
Small & Manageable

http://www.flickr.com/photos/gattou/2285955438/sizes/m/

Large and difficult

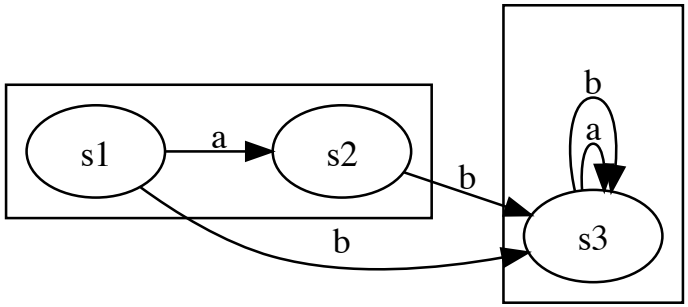
http://www.flickr.com/photos/kevenlaw/2887820440/sizes/m/in/photostream/

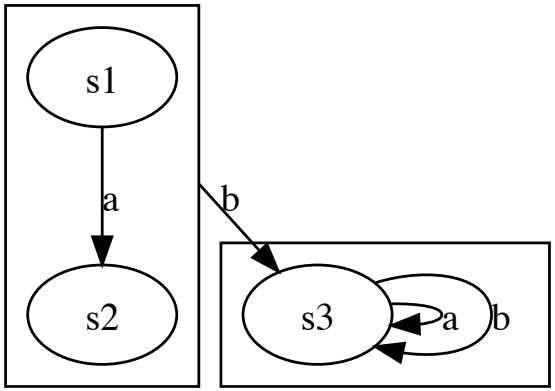
Solution: Add a Hierarchy



Complexity Reduction

- Abstraction
  - Only look at what you want to see
- Cyclomatic complexity
  - E - N + 2
  - E is reduced when a transition from a group replaces individual transitions



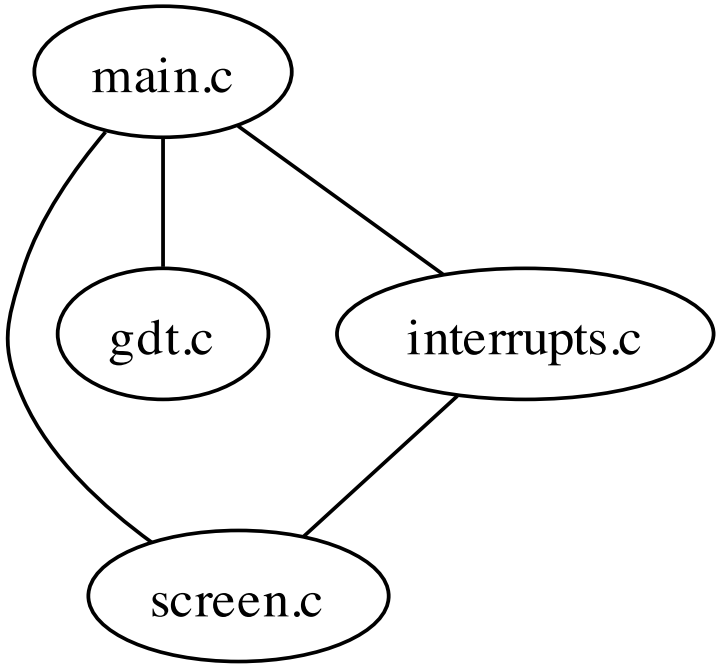


Hierarchy Generation = Clustering

- Bundle related states together
- What does ‘related’ mean?
  - Generally
  - Given limited information available?

Bunch: Search-Based Clustering

- Classic example: Bunch [Mitchell & Mancoridis]
- Similar problem
  - Software module dependency graph (MDG)
  - Group similar modules together



An example MDG

Bunch: Search-Based Clustering

- Multiple Hill Climbs
- Maximises a fitness function
  - MQ
  - Minimise number of edges between groups
  - Designed with low coupling, high cohesion in mind

Crunch: Search Based State Machine Clustering

- Bunch works (very well!) but isn't ideal for state machines
- Crunch mitigates problems with Bunch:
  - Takes a state machine (reflexive edges and all)
  - GA, evolves a linear sequence of merges
  - Designed with fitness function development in mind
- Open source: http://code.google.com/p/sbgct

Crunch

Getting results, but how good are they?

Evaluating Hierarchies

- Generally:
  - Compare with a known good solution
- Bunch:
  - Stability: compare fitness values of many solutions
  - Ask developers
- These mostly assess how good the algorithm is at approximating existing results, not how good the result is

Evaluating Crunch Output

- No ‘right’ answer
- No expert to discuss a result with
- Currently just using a set of metrics:
  - Cyclomatic complexity
    - Flat state machine after merging edges
    - Top level
  - Number of edges between & within clusters
  - MQ

Work so far...

- Crunch
- Technique for getting state machines via AspectJ & StateChum
- Fitness functions:
  - MQ
  - Similarity (based on connectivity of two states)
  - Cyclomatic complexity
  - and more…

Results: Commons HashBag

Fitness Function		Intra Edges	Extra Edges	Top Level CC	Number of Clusters	MQ
MQ	Avg.	23.1	14.9	9.8	7.1	4.129825779
	Min	19	10	6	6	3.207142857
	Max	28	19	15	9	5.183333333
Cyclomatic	Avg.	9.1	28.9	23.9	7	0.914081416
	Min	5	23	20	5	0.662337662
	Max	15	33	26	9	1.162337662
FeatureSet	Avg.	28.9	9.1	8.6	2.5	0.917875196
	Min	20	1	1	2	0.8125
	Max	37	18	18	3	1
Lutz	Avg.	23.2	14.8	13.9	2.9	0.935533301
	Min	9	2	2	2	0.564102564
	Max	36	29	27	4	1.191176471
XOR	Avg.	9.5	28.5	28.5	2	0.482550944
	Min	6	26	26	2	0.315789474
	Max	12	32	32	2	0.613636364
Random	Avg.	18	20	18.4	3.6	0.874152384
	Min	10	4	4	2	0.668269231
	Max	34	28	26	6	0.974545455

Future Work

- Improve evaluation method
  - Better metrics
  - human-based study
- Develop fitness functions
- In-depth case studies