University of Glasgow | School of Computing Science

Utility-Based Heap Sizing

Callum Cameron

Jeremy Singer

firstname.lastname@glasgow.ac.uk

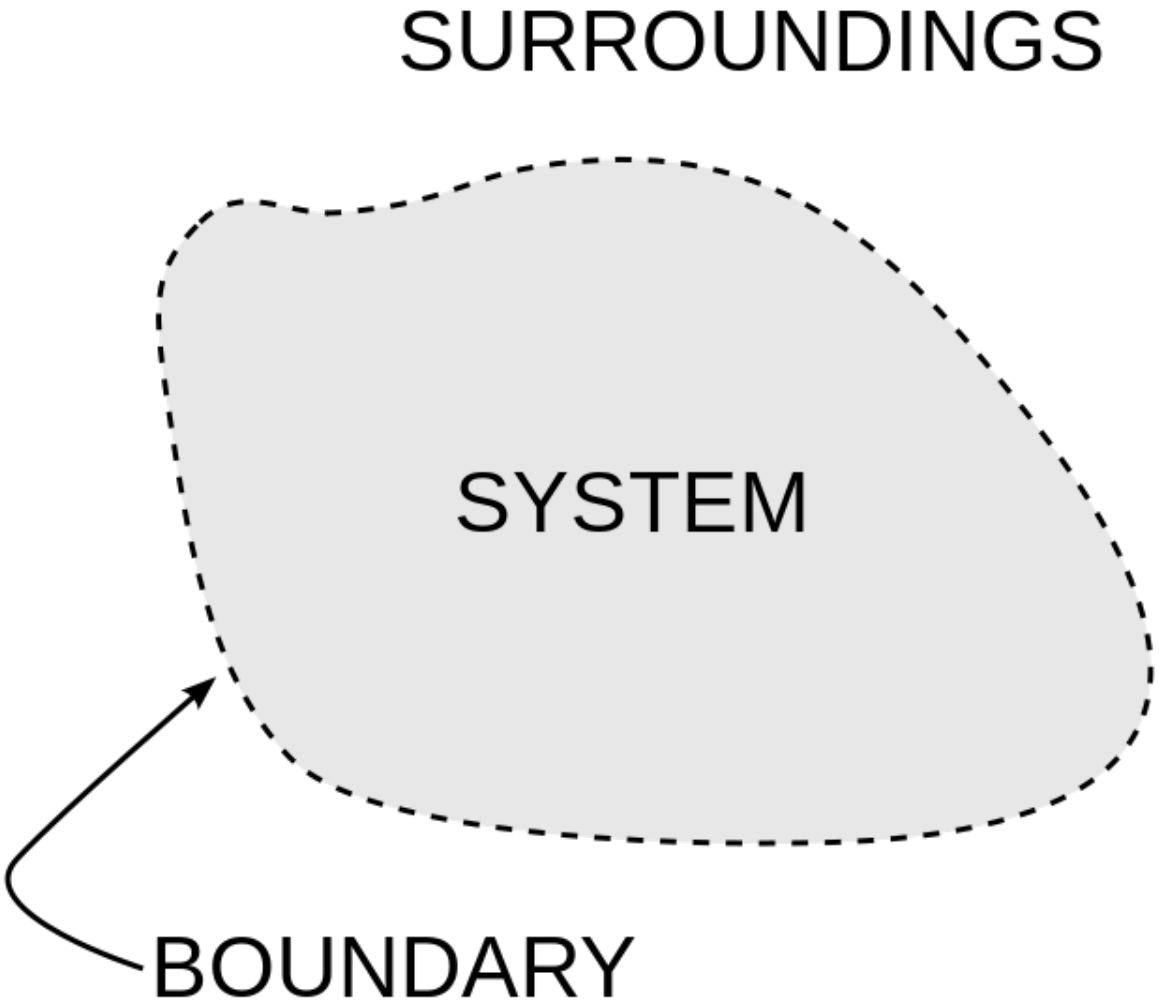
Analogies for GC

imaginative metaphorical interpretations of memory management

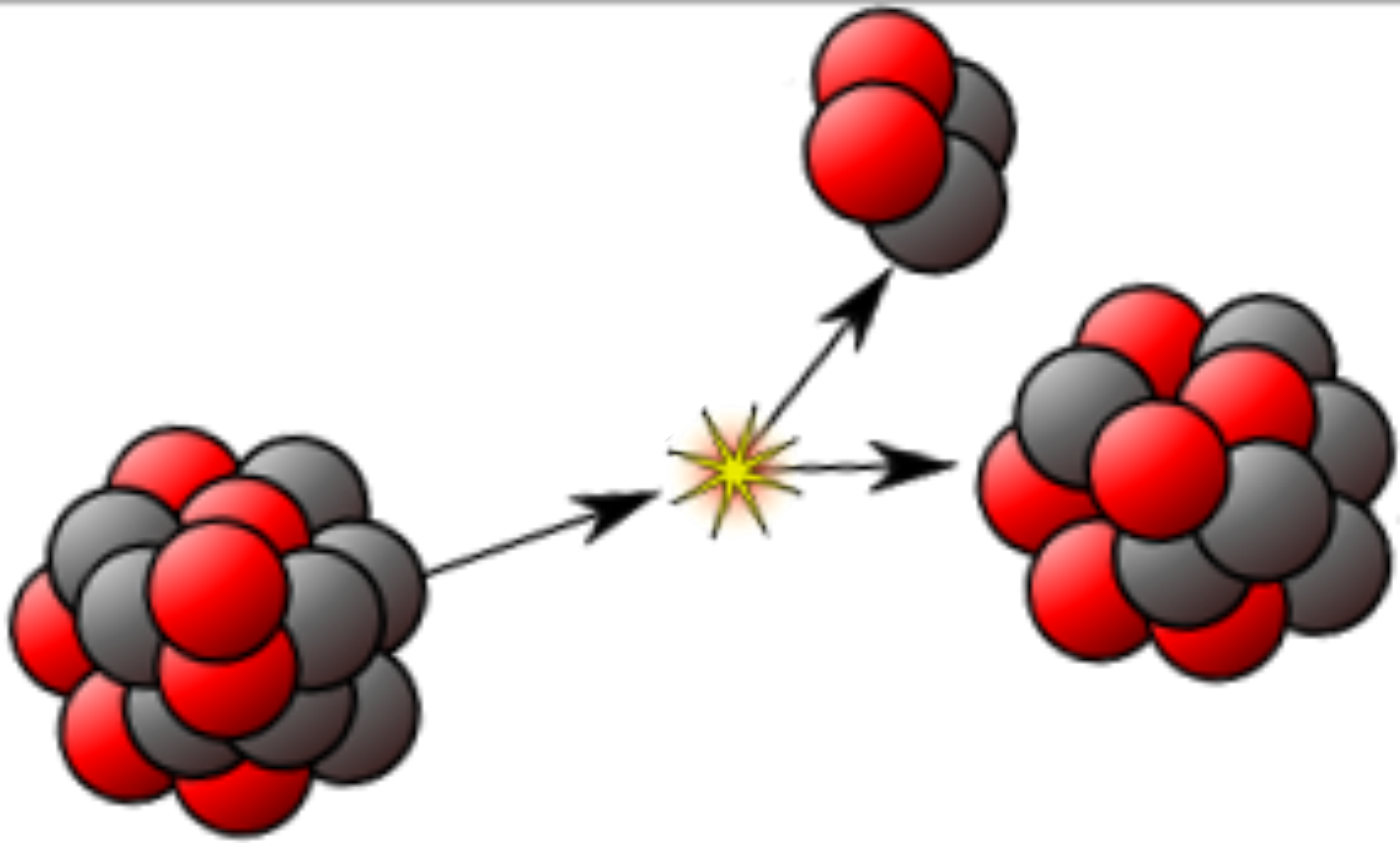
Analogies for GC

intuitive mathematical interpretations of memory management

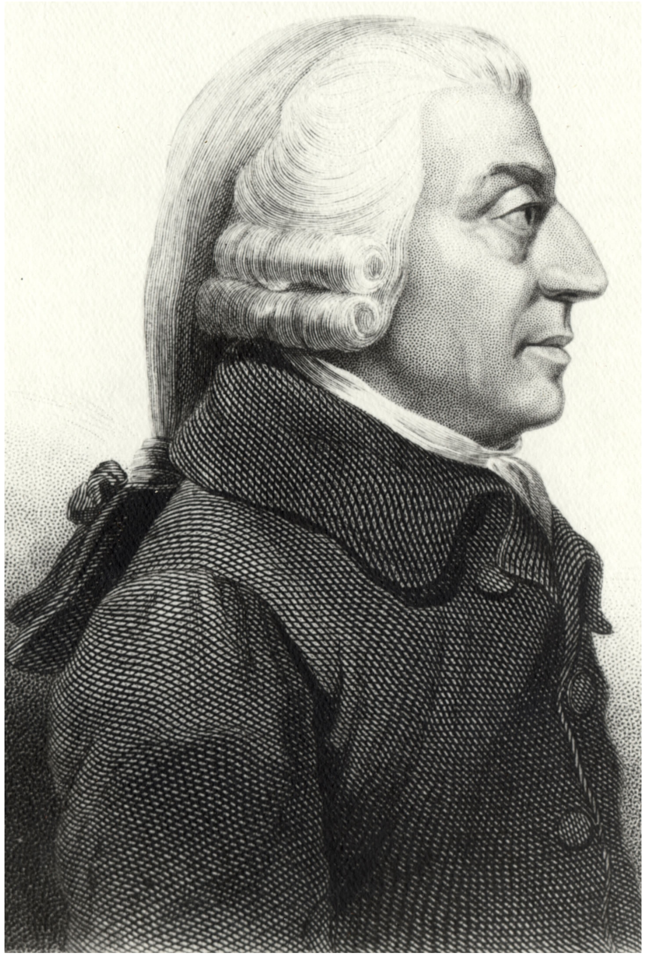
Thermodynamics [Baker, 1994]

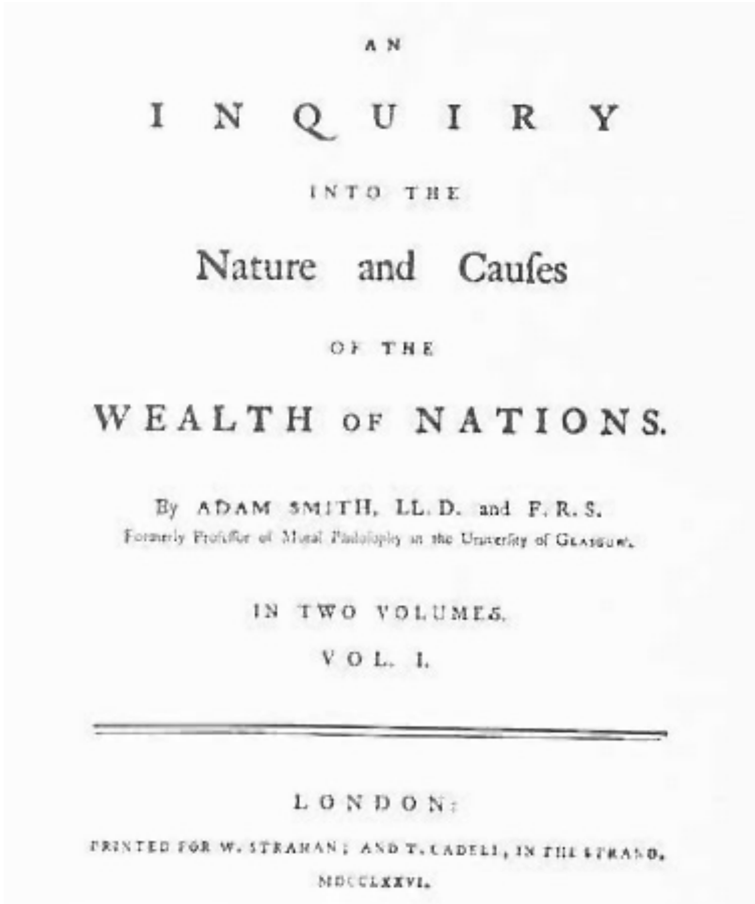


Radioactive half-life [Clinger, 1997]



Economic supply/demand [Singer & Jones, 2010]





AN

INQUIRY

INTO THE

Nature and Caufes

OF THE

WEALTH OF NATIONS.

By ADAM SMITH, LL. D. and F. R. S.
Formerly Profeffor of Moral Philofophy in the Univerfity of Glasgow.

IN TWO VOLUMES.

VOL. I.

LONDON:

PRINTED FOR W. STRAHAN; AND T. CADELL, IN THE STRAND.

MDCCLXXVI.

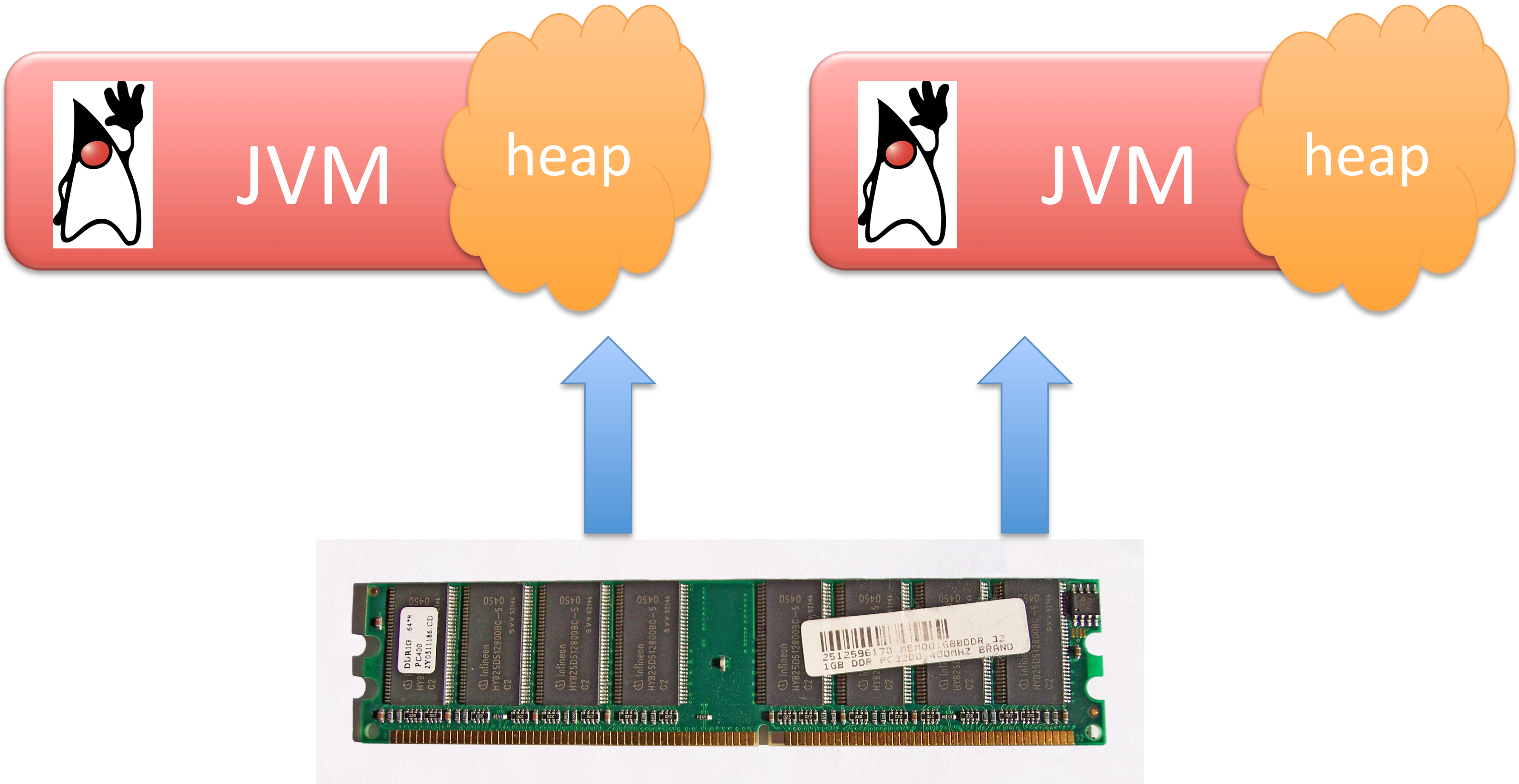
Our Problem

- multiple parties competing for shared resource
- limitations on resource availability
- want to maximize utility

Simple Example

- haggis and irn bru

Analogy with GC



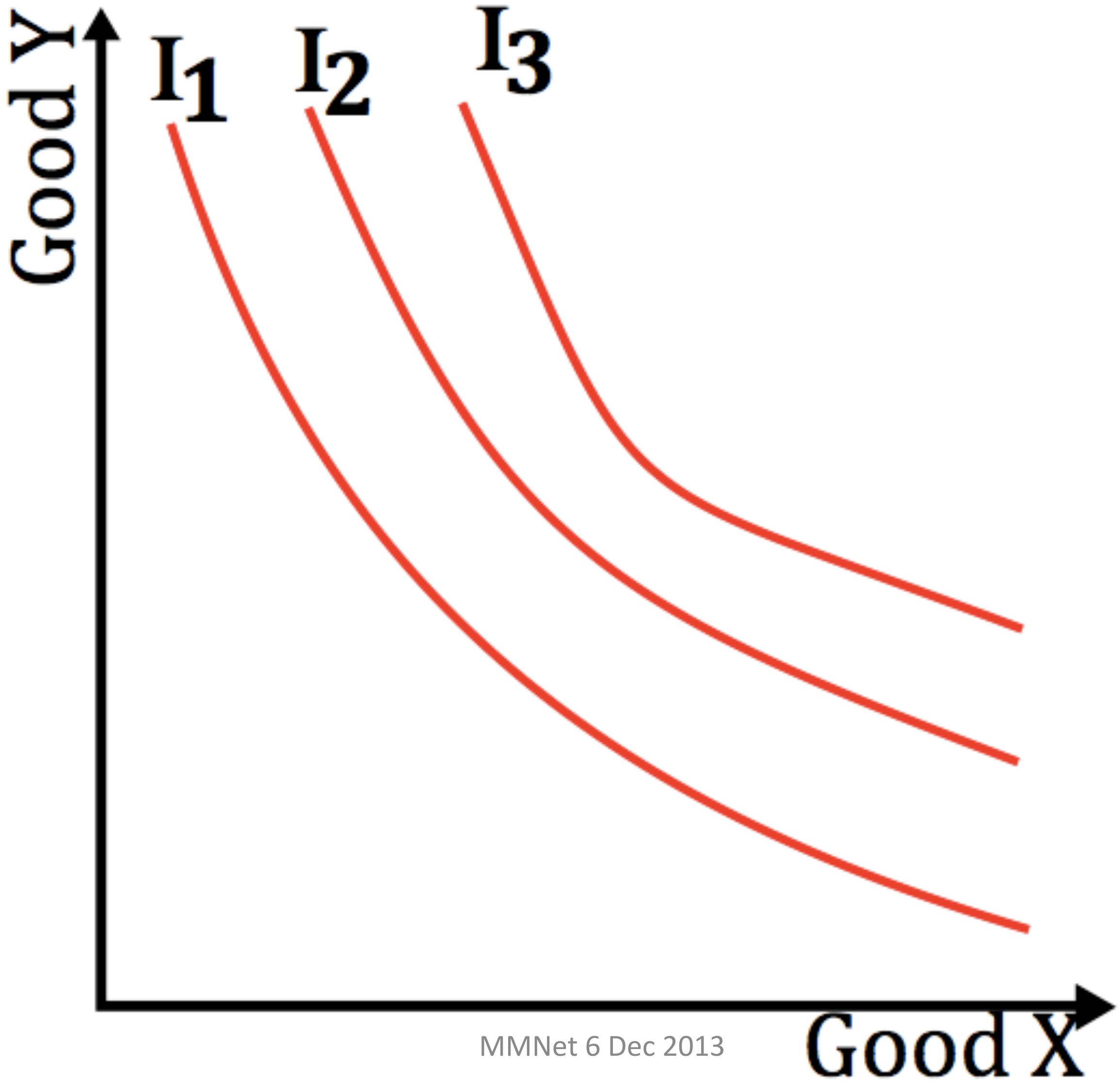
Utility for single bm

- for a single benchmark, we equate utility with throughput
- for DaCapo benchmarks, relate throughput with number of completed iterations in set time
- utility can be expressed as a function of heap size
- $U_{bm} = f(h_{bm})$

Combined utility for 2 bms

- multiply their individual utilities
- $U_{combined} = U_{bm1} * U_{bm2}$
- $= f(h_{bm1}) * g(h_{bm2})$
- Variations on this equation…

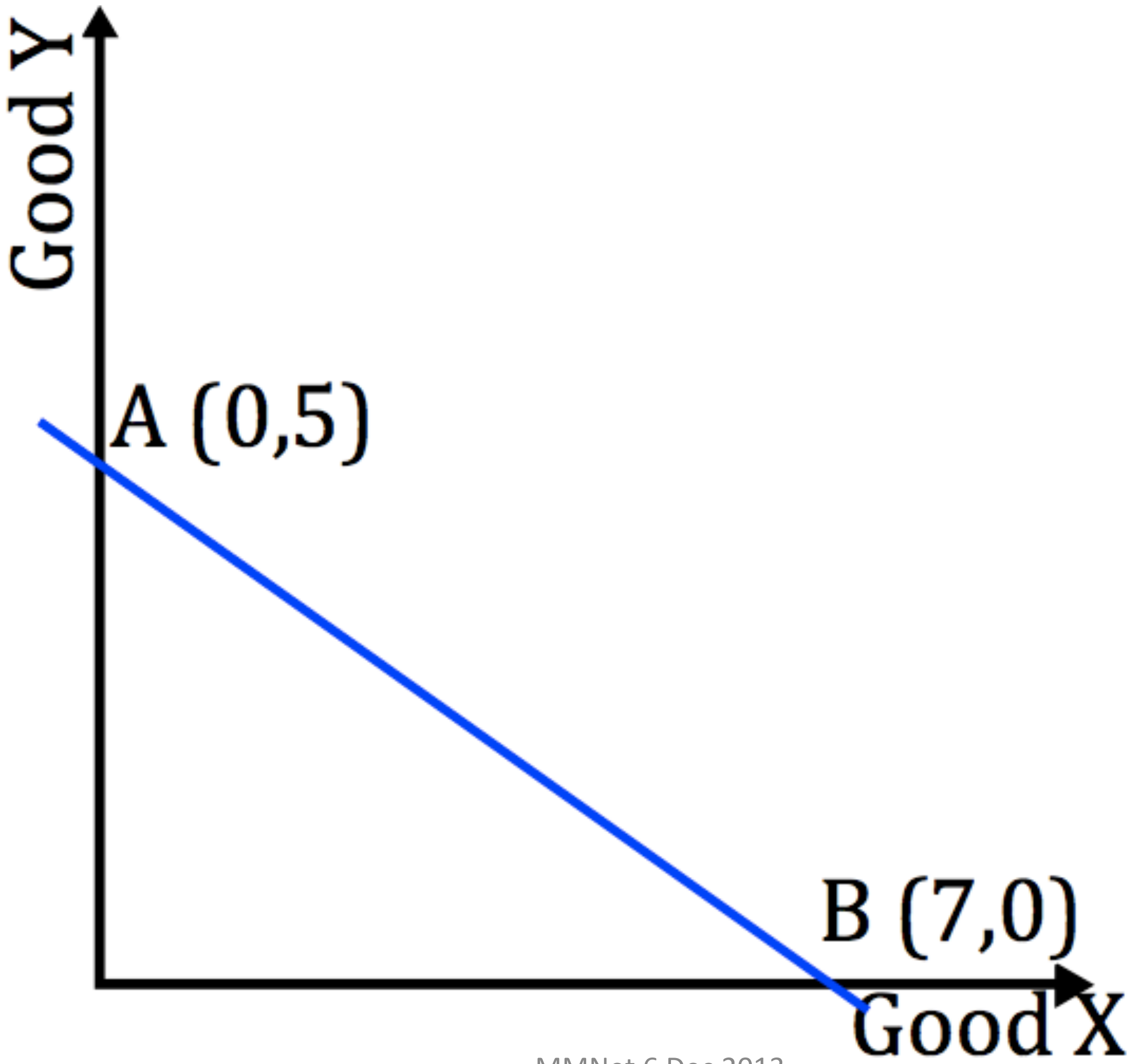
Iso-utility curves (indifference curves)



Maximum Heap Size Budget

- $h_{bm1} + h_{bm2} <= M$

Budget line



Intersection of lines

- Maximal utility for given budget when budget line just touches isoutility line
- (draw graph)

Analytical Solution

- substitute h_{bm2} for $(M-h_{bm1})$
  - since on budget line
- Differentiate combined utility equation wrt each variable to get marginal utilities and *marginal rate of substitution (gradient of indifference curve)*
- When consumers maximize utility with respect to a budget constraint, the indifference curve is tangential to the budget line

Provisional solution:

- Given individual utilities of the form:
  - $U_{bm1} = a\ (h_{bm1})^b$
  - $U_{bm2} = c\ (h_{bm2})^d$

- We can show that the combined utility is maximized when:
  - $h_{bm1} = M * b / (b+d)$
  - $h_{bm2} = M - h_{bm1}$

This makes some sense!

- When we run two benchmarks which are the same, then we should give M/2 to each.

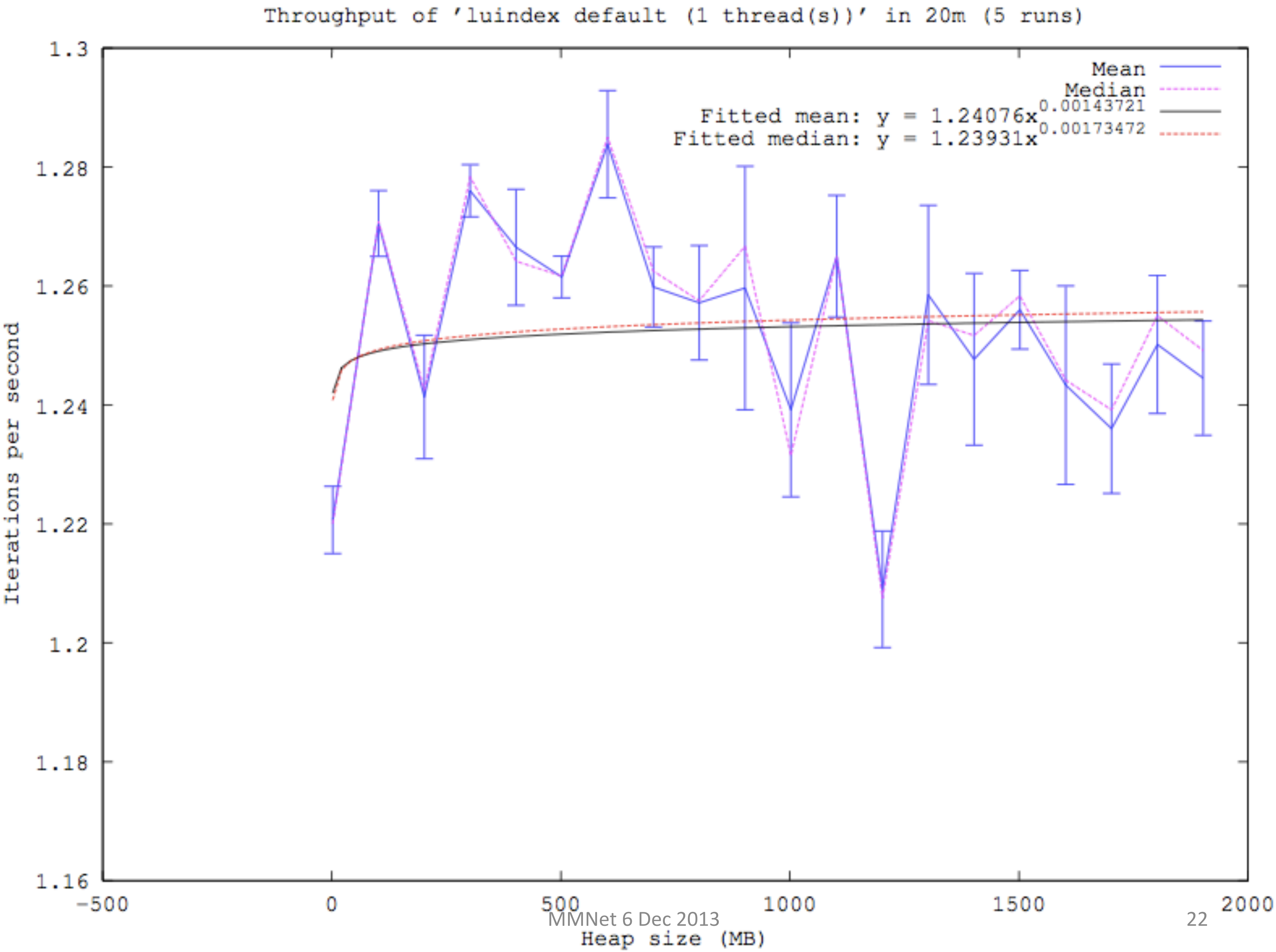
Experiments

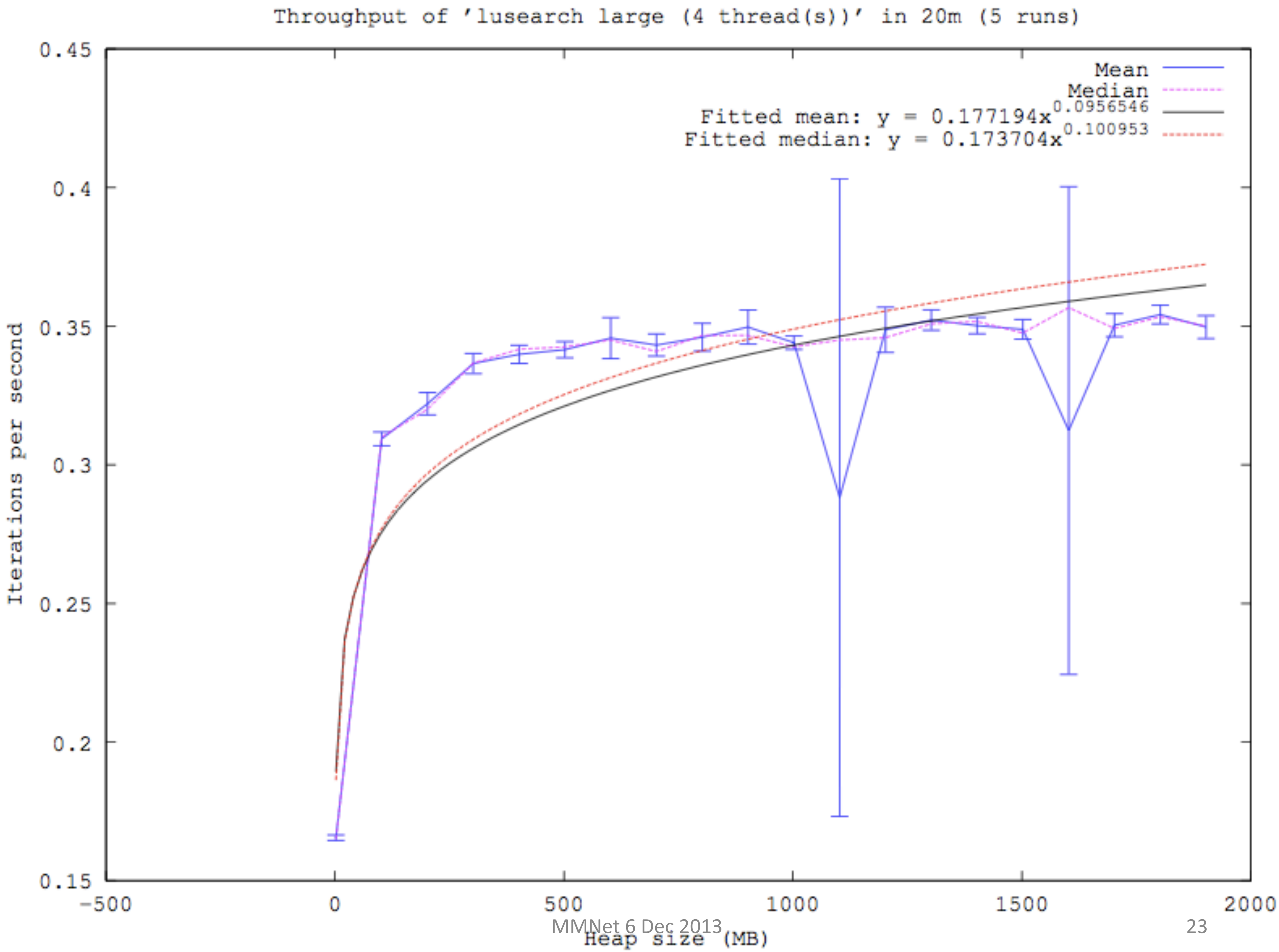
- Use OpenJDK with fixed heap size
- Use DaCapo 9.12 benchmarks
- Run on x86_64 Linux

Obtain utility curve for each bm

- do empirical curve fitting
- $U_{bm} = a\,(h_{bm})^{b}$

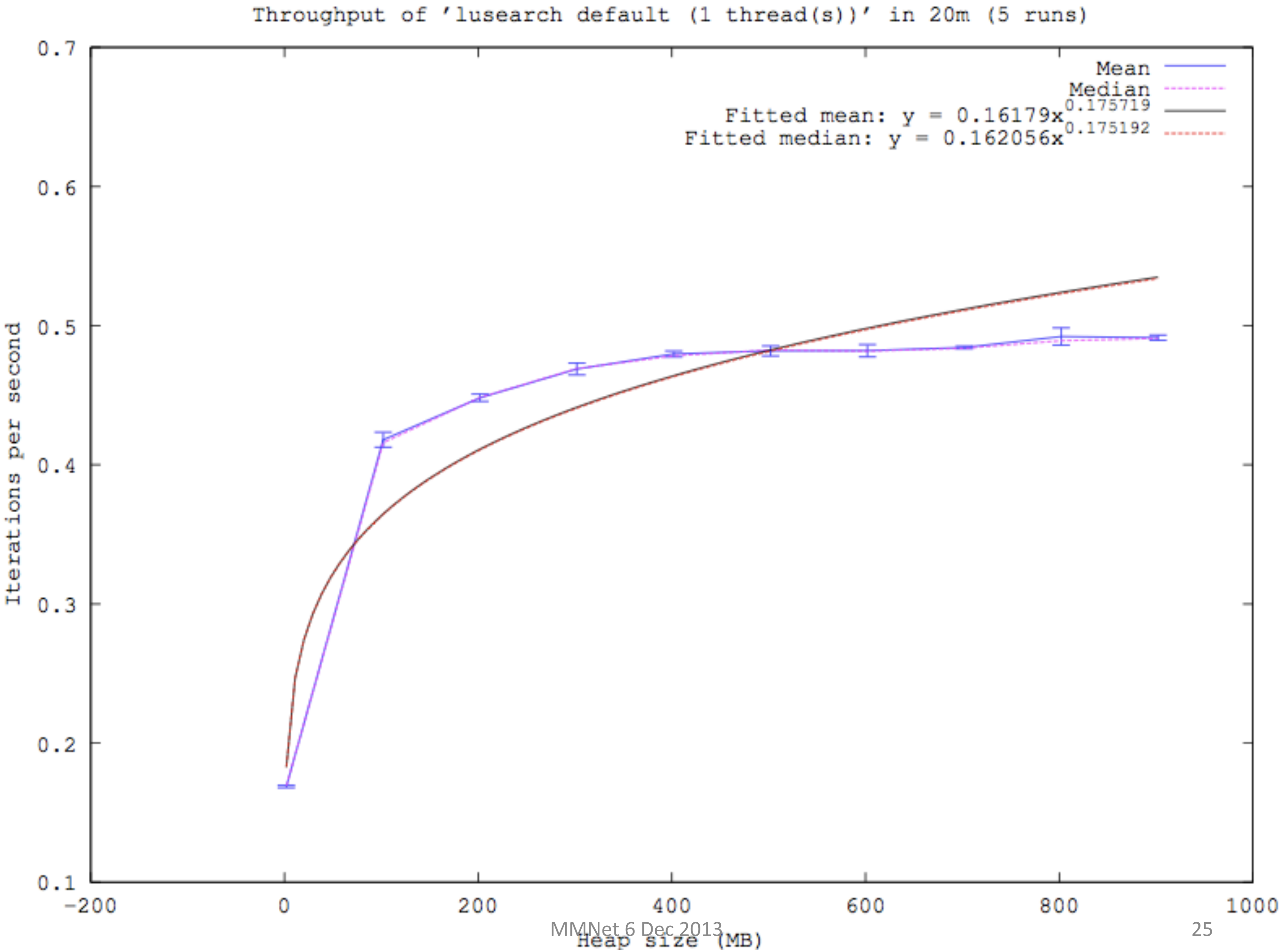
Aside: don’t use AWS


Throughput of 'luindex default (1 thread(s))' in 20m (5 runs)
Mean
Median
Fitted mean: y = 1.24076x^0.00143721
Fitted median: y = 1.23931x^0.00173472
Iterations per second
1.3
1.28
1.26
1.24
1.22
1.2
1.18
1.16
-500
0
500
1000
1500
2000
Heap size (MB)

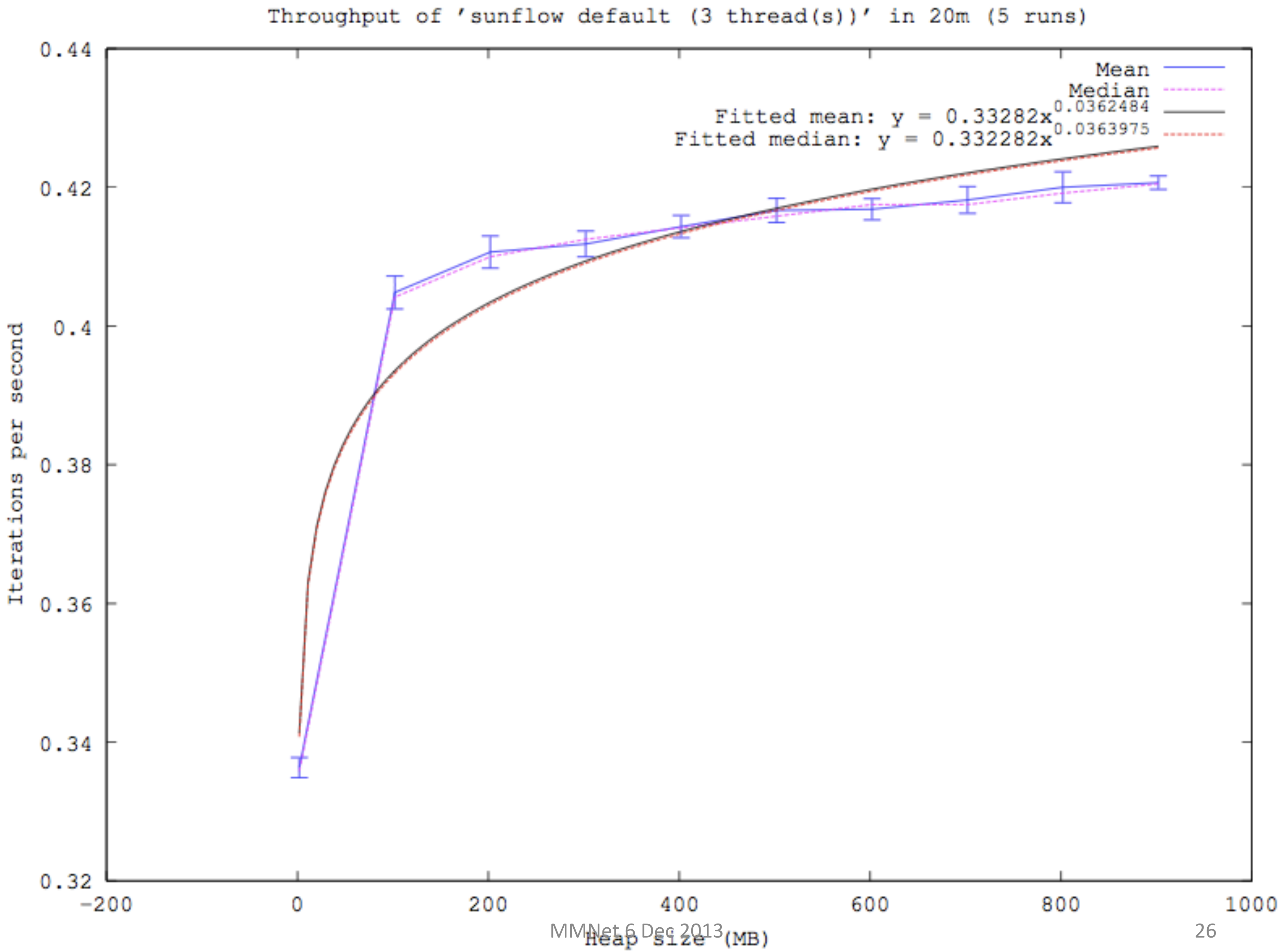


Throughput of 'lusearch large (4 thread(s))' in 20m (5 runs)
Mean
Median
Fitted mean: y = 0.177194x^0.0956546
Fitted median: y = 0.173704x^0.100953
Iterations per second
Heap size (MB)
0.45
0.4
0.35
0.3
0.25
0.2
0.15
-500
0
500
1000
1500
2000

Experiments on local machine



Throughput of 'lusearch default (1 thread(s))' in 20m (5 runs)
Mean
Median
Fitted mean: y = 0.16179x^0.175719
Fitted median: y = 0.162056x^0.175192
Iterations per second
Heap size (MB)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
-200
0
200
400
600
800
1000

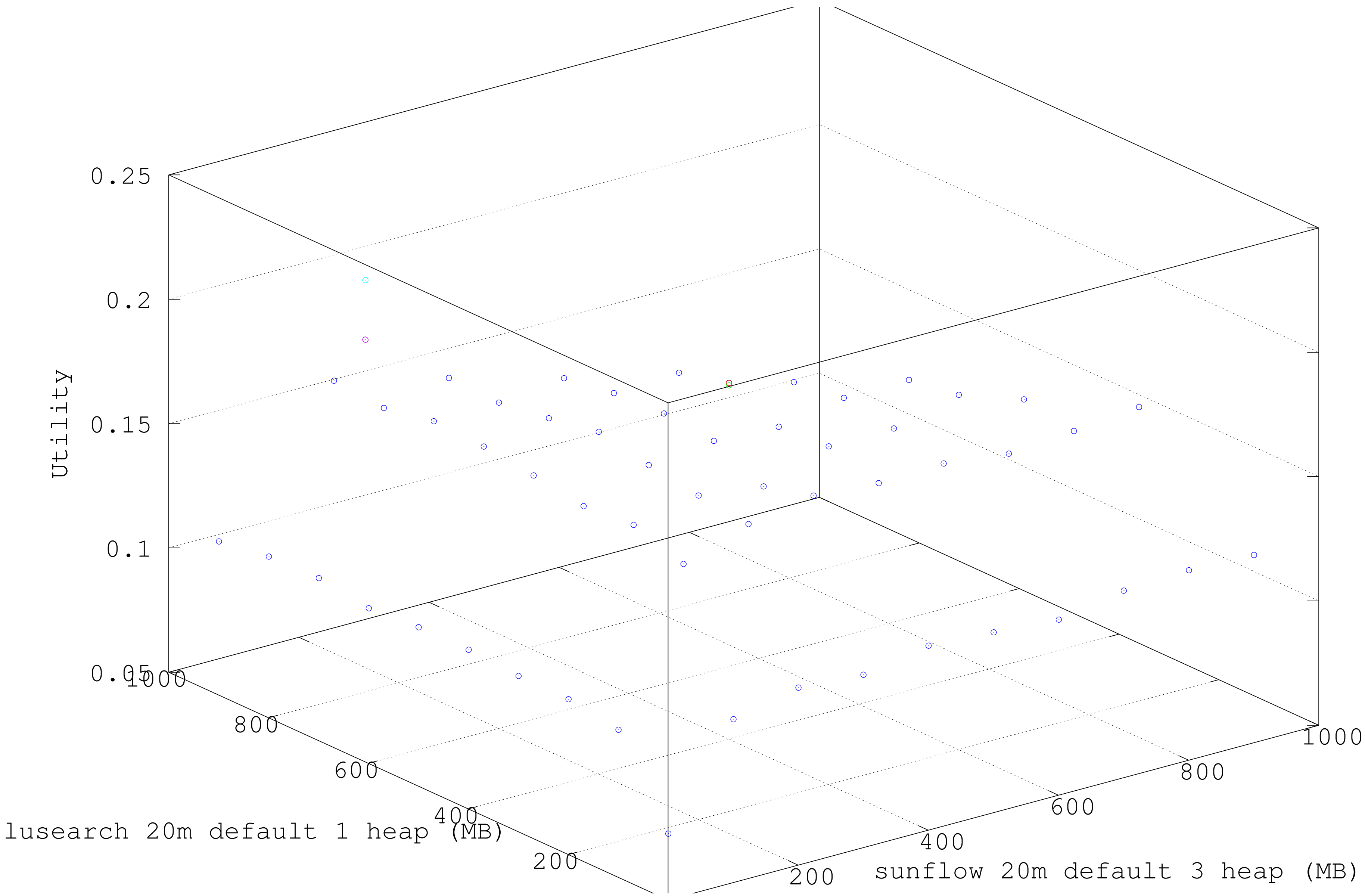

Throughput of 'sunflow default (3 thread(s))' in 20m (5 runs)
Mean
Median
Fitted mean: y = 0.33282x^0.0362484
Fitted median: y = 0.332282x^0.0363975
Iterations per second
Heap size (MB)
0.44
0.42
0.4
0.38
0.36
0.34
0.32
-200
0
200
400
600
800
1000

Utility curve-fitting

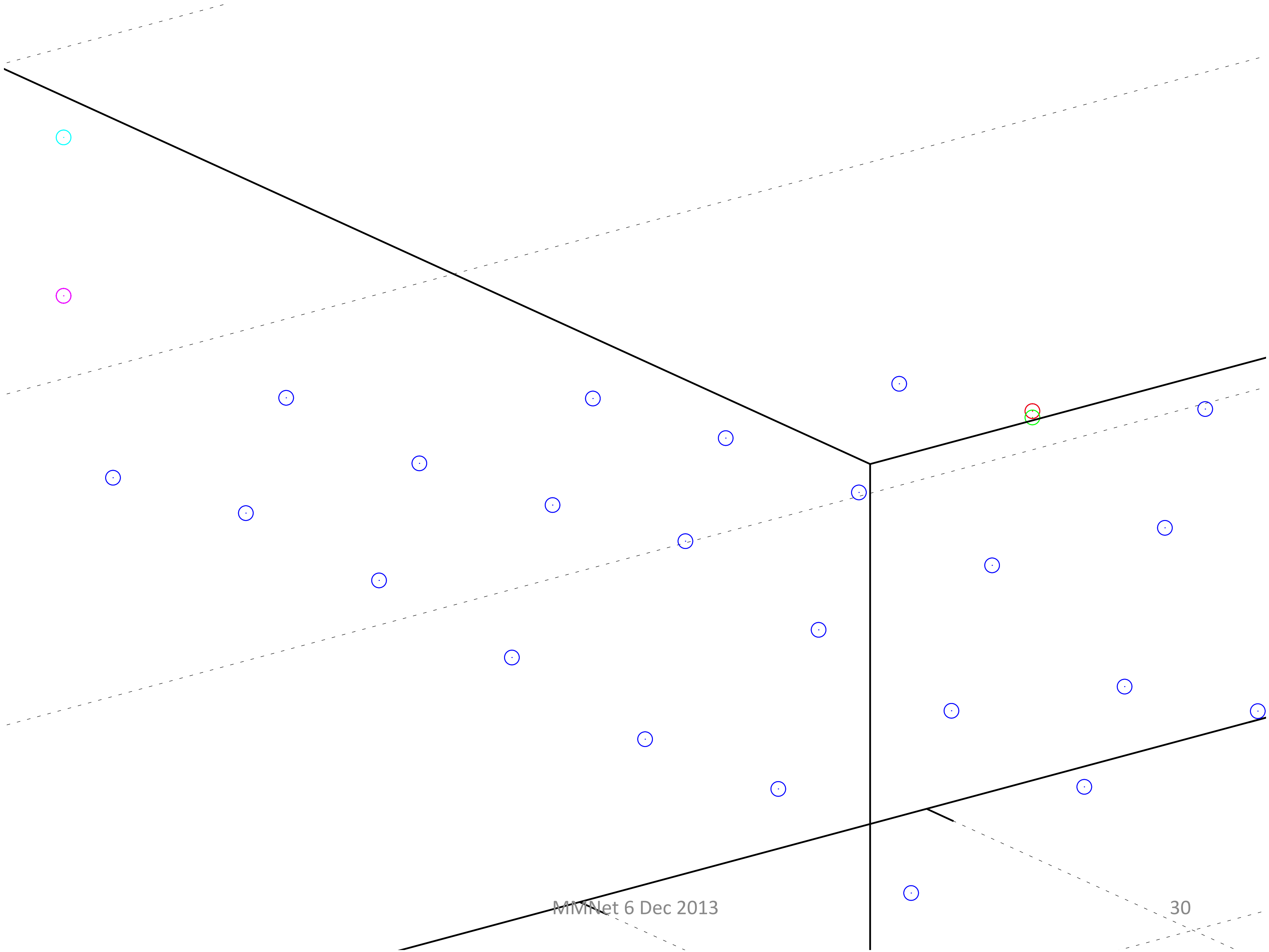
- Assume form $U_{bm} = a\ (h_{bm})^b$
- least-squares regression
- How does this work for maximising combined utility?
- …

Utility Space Exploration

- run combined experiment – measure overall throughput
- sample space of heap sizes,
- 3d graph
  - h_{bm1}, h_{bm2} on x and y
  - $U(h_{bm1},h_{bm2})$ on z
- e.g. …



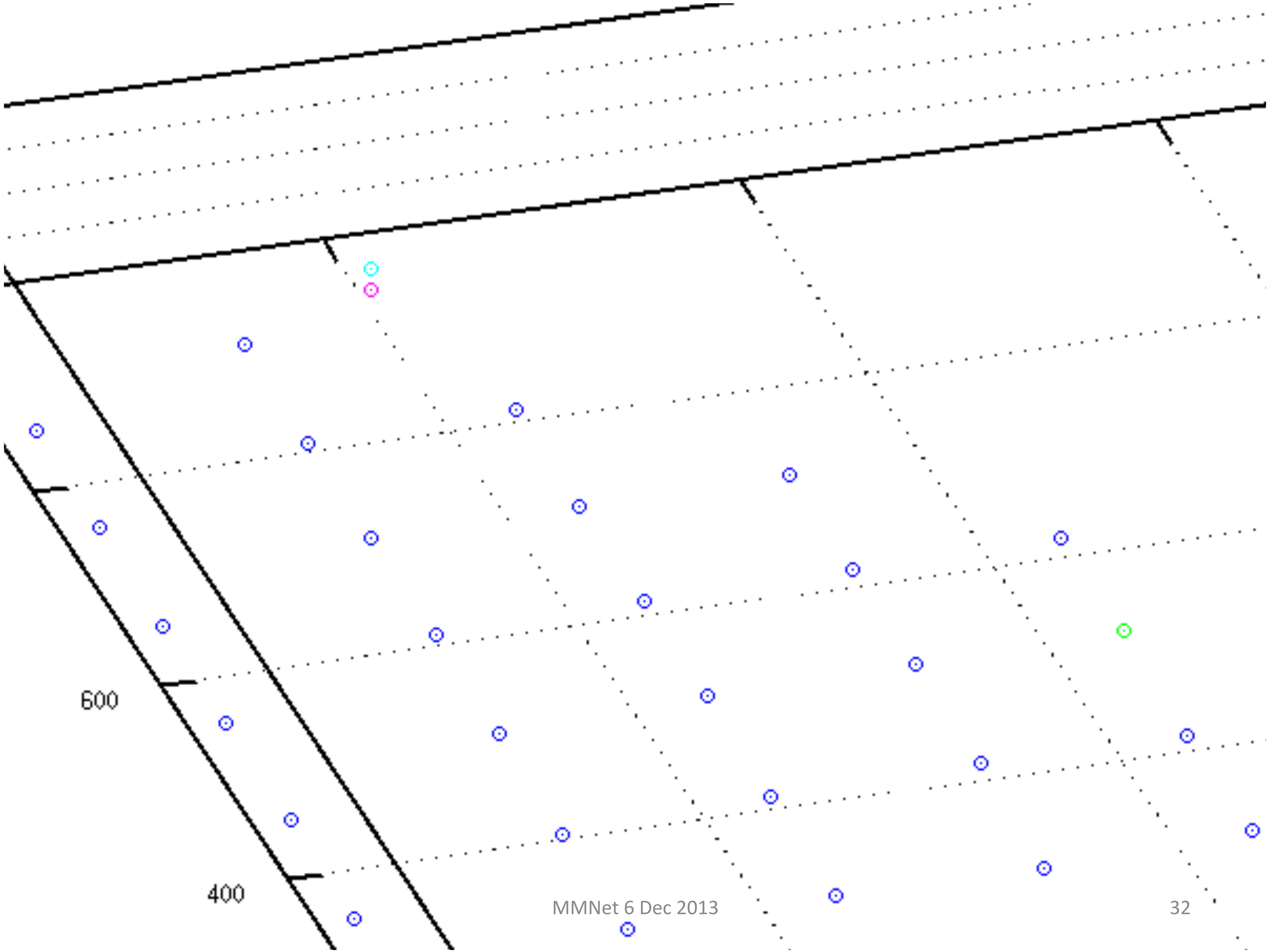
Utility
0.25
0.2
0.15
0.1
0.05
1000
800
600
400
200
lusearch 20m default 1 heap (MB)
200
400
600
800
1000
sunflow 20m default 3 heap (MB)



Evaluation

- compare predicted optimal performance with observed

- single example: predicted performance is 91% of optimal.


600
400

To do

- gather more data, more benchmarks
- fit a better equation to our curves
- deal with normalizing utilities
- deal with prioritizing benchmarks

More ambitious extensions

- handle more than two VMs
- incorporate paging into utility space measures, see [Hertz, 2011]
- allow dynamic resizing based on utility calculations
- heterogeneous runtimes (Jikes RVM, Poly/ML, GHC)
- apply to other shared resources (cores)

Conclusions

- Economic theory gives a principled way to *share resources*
- We have applied utility maximization to *static heap sizing* for two concurrent JVMs
- Lots more to do!