Control Theory for Adaptive Heap Resizing

Jeremy Singer, David White, Jon Aitken, Richard Jones

University of Glasgow

The University of York

University of Kent

http://www.dcs.gla.ac.uk/~jsinger/pdfs/ismm13.pdf

Control Theory for Principled Heap Sizing

David R. White Jeremy Singer
School of Computing Science
University of Glasgow
{david.r.white,jeremy.singer}@glasgow.ac.uk

Jonathan M. Aitken
Department of Computer Science
University of York
jonathan.aitken@york.ac.uk

Richard E. Jones
School of Computing
University of Kent
r.e.jones@kent.ac.uk

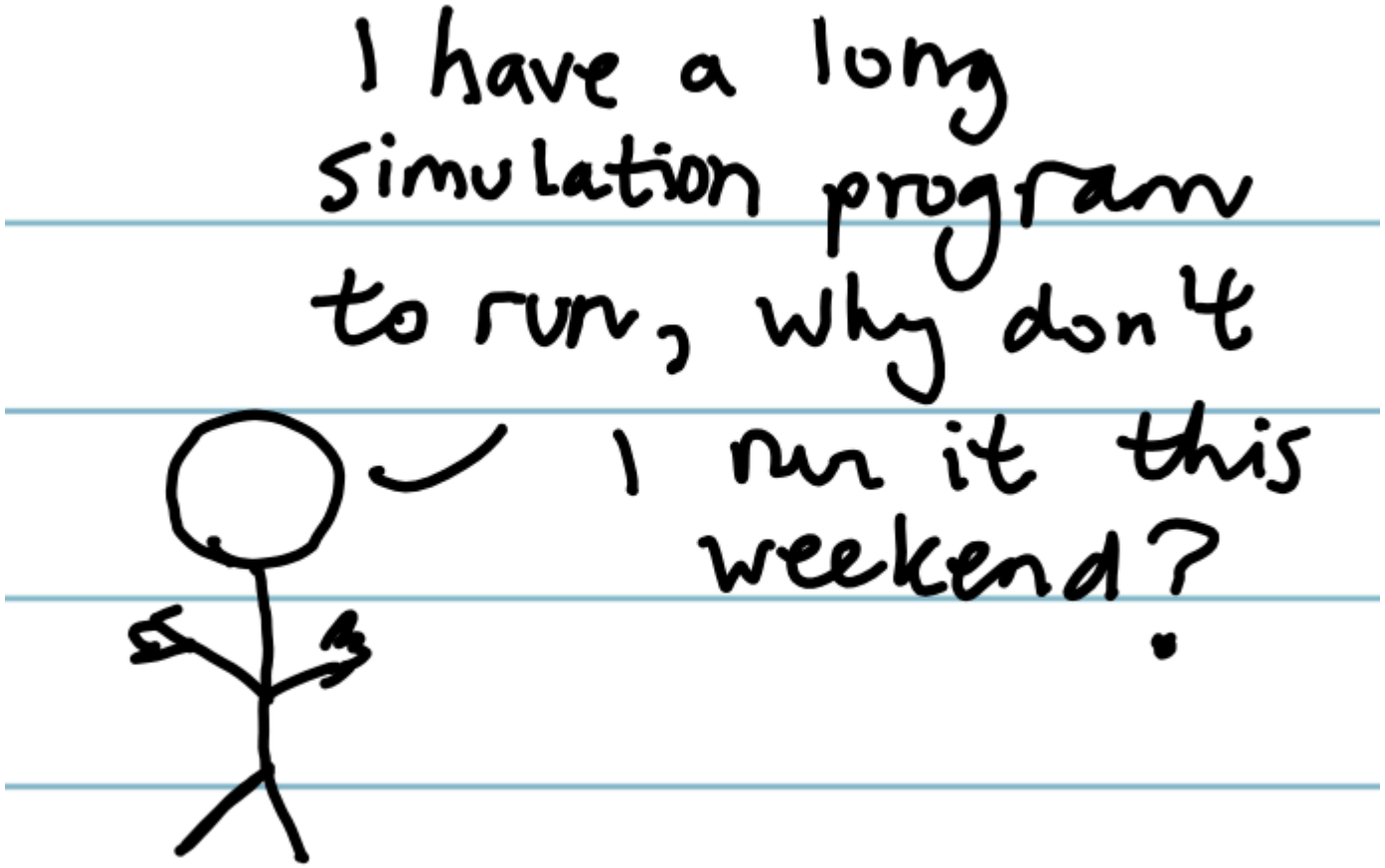
Abstract

We propose a new, principled approach to adaptive heap sizing based on control theory. We review current state-of-the-art heap sizing mechanisms, as deployed in Jikes RVM and HotSpot. We then formulate heap sizing as a control problem, apply and tune a standard controller algorithm, and evaluate its performance on a set of well-known benchmarks. We find our controller adapts

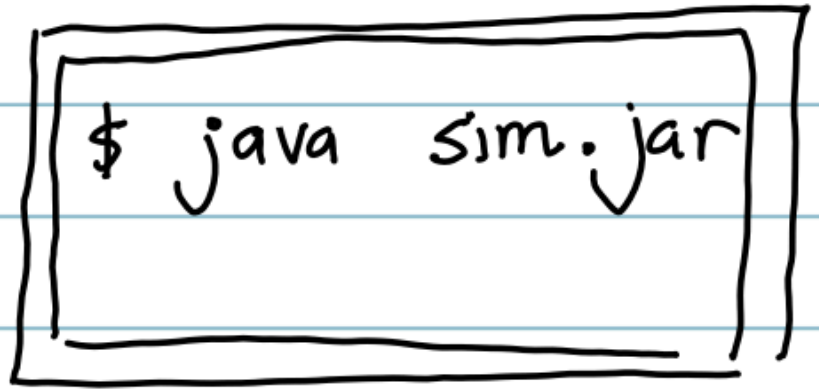
paging [36]. Setting a large static heap size is an inefficient use of memory; this should be avoided.

This paper proposes the use of *control theory* [24] to adjust heap sizes dynamically. In contrast to existing, heuristic-based techniques for heap sizing, control theory provides a principled mathematical approach. As virtual machines (VMs) become more sophisticated and widespread, a progression from expert-designed, hand-

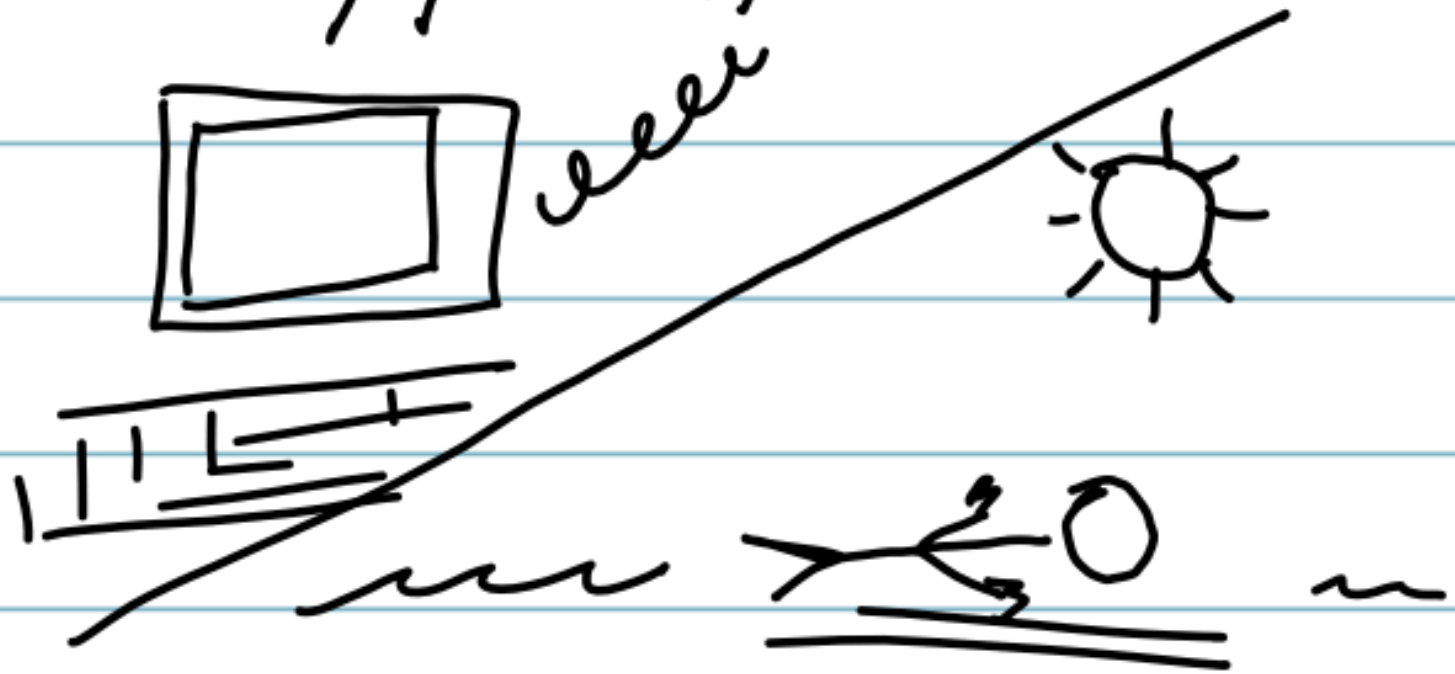
a sad story


I have a long simulation program to run, why don't I run it this weekend?

FRIDAY EVENING



SATURDAY / SUNDAY



MONDAY MORNING

```
Fatal error.
Java OutOfMemoryError.
$
```

oh no!

Java -Xmx, Max memory on system

14

3

My Java application runs another Java application, by running the process "java -jar j.ja
known to use a LOT of memory depending on the dataset it is given, and often gets an
OutOfMemoryError heap. So I want to use -Xmx on it, so that I can allocate as much m
possible (or close to). I was thinking of getting the total memory on the system, then spe
90% of that in -Xmx.

Is there any solution to my problem? And, how does my solution sound?

Edit: I cant reduce the memory consumption as the memory being used is by Java's bu
compression, which I am using to pack some JAR files.

java memory-management

7 Answers

active oldes

4

Depending on your OS, this might work for getting the free and available memory size:

```
java.lang.management.OperatingSystemMXBean mxbean = java.lang.managemen
com.sun.management.OperatingSystemMXBean sunmxbean = (com.sun.managemen
long freeMemory = sunmxbean.getFreePhysicalMemorySize();
long availableMemory = sunmxbean.getTotalPhysicalMemorySize();
```

From there, you can figure out 80-90% and launch your jar with the max memory size yo

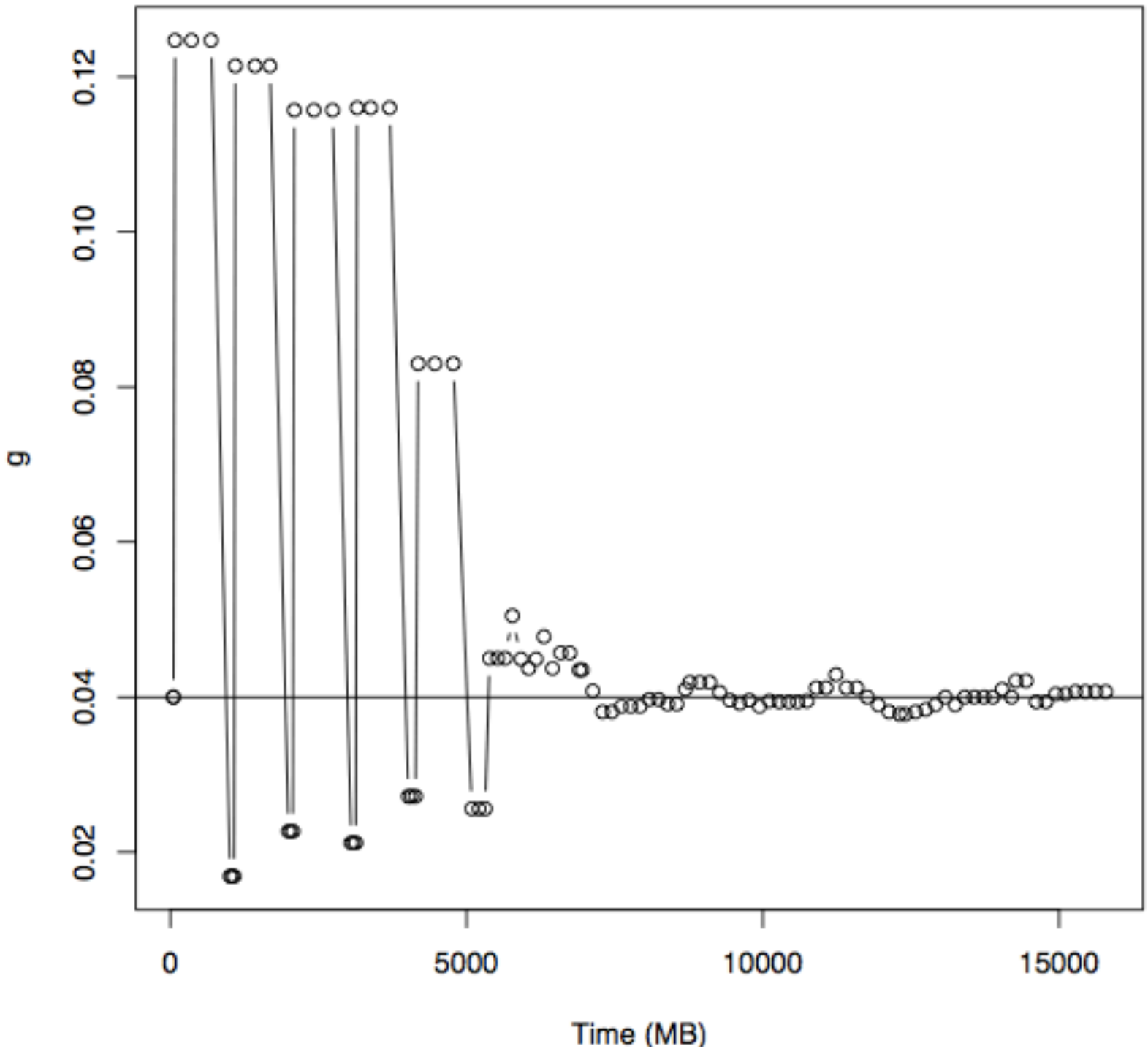
I don't know that this works with all OS's (i.e. Windows), but it worked when I tested it wi
and Linux.

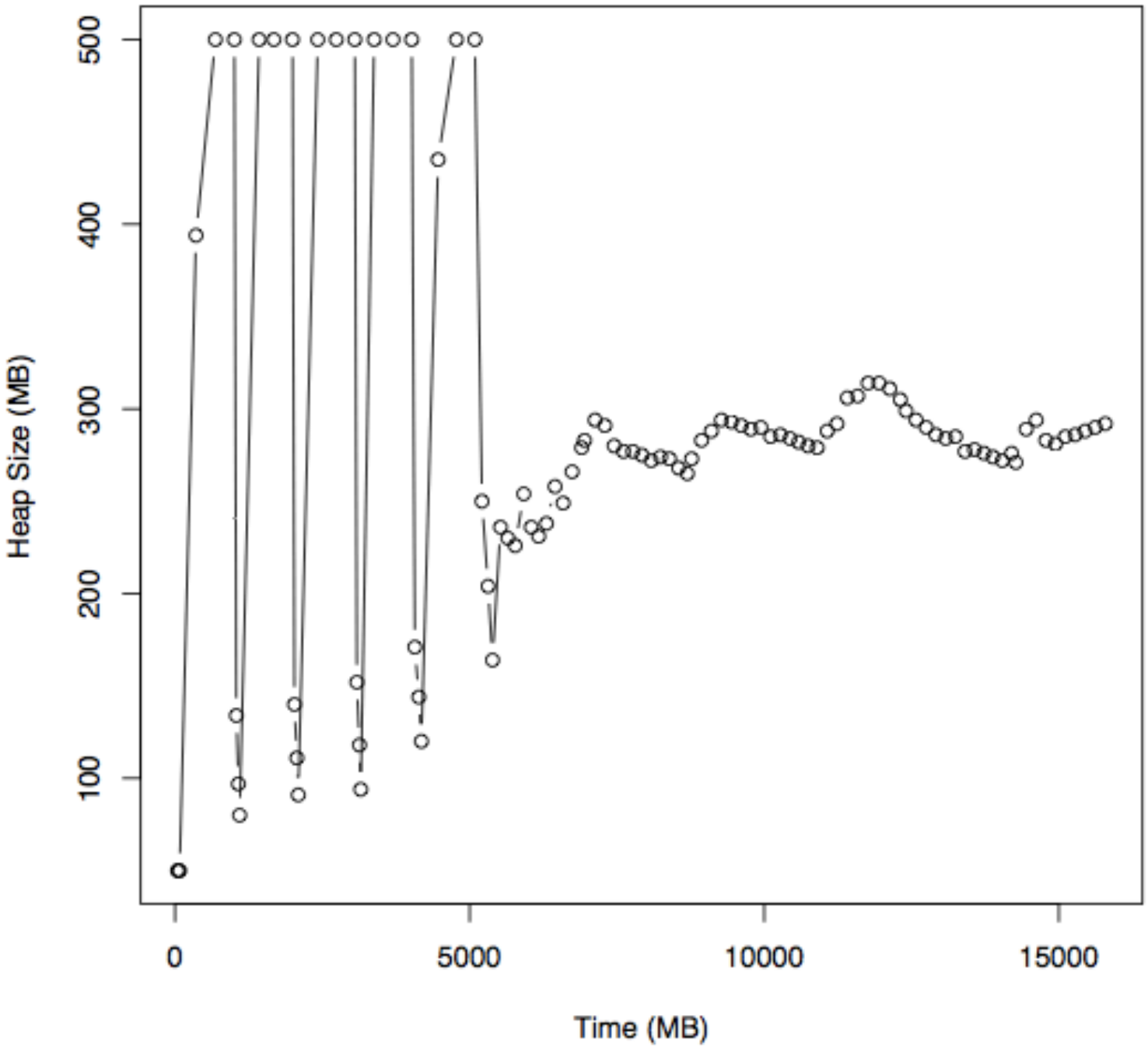
what we did

New heap sizing parameter

- User specifies target time to spend in GC
- (GC overhead, *g*)

- Controller adjusts heap size to meet target


g
0.02
0.04
0.06
0.08
0.10
0.12
0
5000
10000
15000
Time (MB)


Heap Size (MB)
500
400
300
200
100
0
5000
10000
15000
Time (MB)

Area under curve …

- embedded systems – power
- utility computing - cost

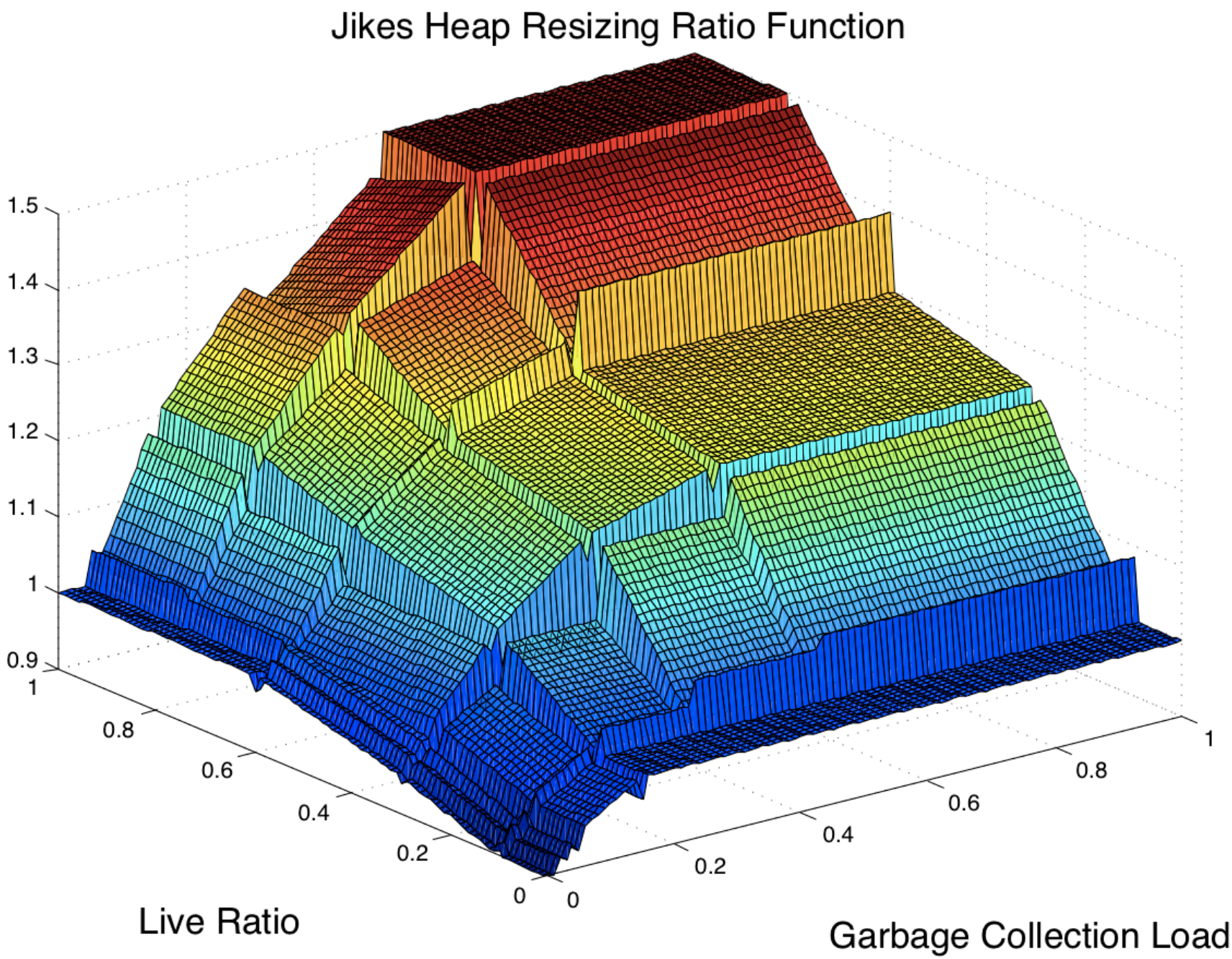
state of the art

Jikes RVM

- **`HeapGrowthManager`** computes heap resize ratio after major GC
- lookup table of resize ratios, indexed by:

$$g = \frac{\text{Time taken for most recent collection}}{\text{Time since last collection}}$$

$$l = \frac{\text{Amount of live data on the heap}}{\text{Current heap size}}$$


Jikes Heap Resizing Ratio Function
1.5
1.4
1.3
1.2
1.1
1
0.9
1
0.8
0.6
0.4
0.2
0
Live Ratio
0
0.2
0.4
0.6
0.8
1
Garbage Collection Load

Bug fix

Dashboards | Projects | Issues

RVM / RVM-943

MMTk HeapGrowthManager heap growth ratio computation has discontinuities

Log In

Details

Type:	Bug	Status:	Resolved
Priority:	Minor	Resolution:	Fixed
Affects Version/s:	None	Fix Version/s:	3.1.2
Component/s:	MMTk		
Labels:	None		
Environment:	affects all.		

- In HeapGrowthManager.computeHeapChangeRatio(), the current implementation determines the heap size change ratio by a lookup in the 2-dimensional function table (indexed by liveRatio and gcLoad). Given a current liveRatio X and gcLoad Y, the code finds the table rows and columns with nearest values above and below X and Y, then does interpolation from these table lookup values to determine the heap size change ratio.
- However, there is a bug in the interpolation. If X (or Y, respectively) is exactly equal to a row (or column, respectively) label value, then the interpolation still happens, between values in rows (cols) either side of row X (col Y). This leads to discontinuities in the heap sizing function - see attached graphs.
- The submitted patch suppresses interpolation (interpolation correction value becomes 0) in the case where X or Y falls on a label value exactly, so avoiding the discontinuity.

OpenJDK

- GC ergonomics system allows user to specify high-level goals for GC
  - desired max GC pause time
  - desired application throughput
  - minimum heap size

OpenJDK

- **`AdaptiveSizePolicy`** applies fixed rules to satisfy these targets:
  - if current pause time > pause time goal, then *decrease* heap size
  - else if application's throughput goal is not being met, then *increase* heap size
  - else *decrease* heap size to reduce memory footprint

OpenJDK

- David Vengerov [ISMM11]
- [The ergonomics system consists of] some heuristic rules that do not guarantee that the GC throughput will actually be maximised as a result

Poly/ML

- **`adjustHeapSize()`** called after major GC
- if l is live data, heap size changed to $K+l$
- K is constant determined by initial parameters

- source code comment: ‘somewhat naïve’

Dalvik

- Easy to grow heap, more difficult to shrink
  - non-moving objects in mature space

$$h' = \frac{\text{current size of live data}}{\text{target utilization rate}}$$

- Heap sizing policy entirely opaque to user

Dalvik heap resizing app

Find the best Android apps | Hot today | Hot this week | All-time popular | Top rated

All apps » Tools » VM Heap Tool (root only)

VM Heap Tool (root only)

by martino

Install

Free

50,000-250,000 downloads, **365** ratings (4.50 average), 51 kb, Permissions, Official Page, Contact

Add to list ▼ | Like | +1 0 | Tweet 2 | Email | QR / more ▼

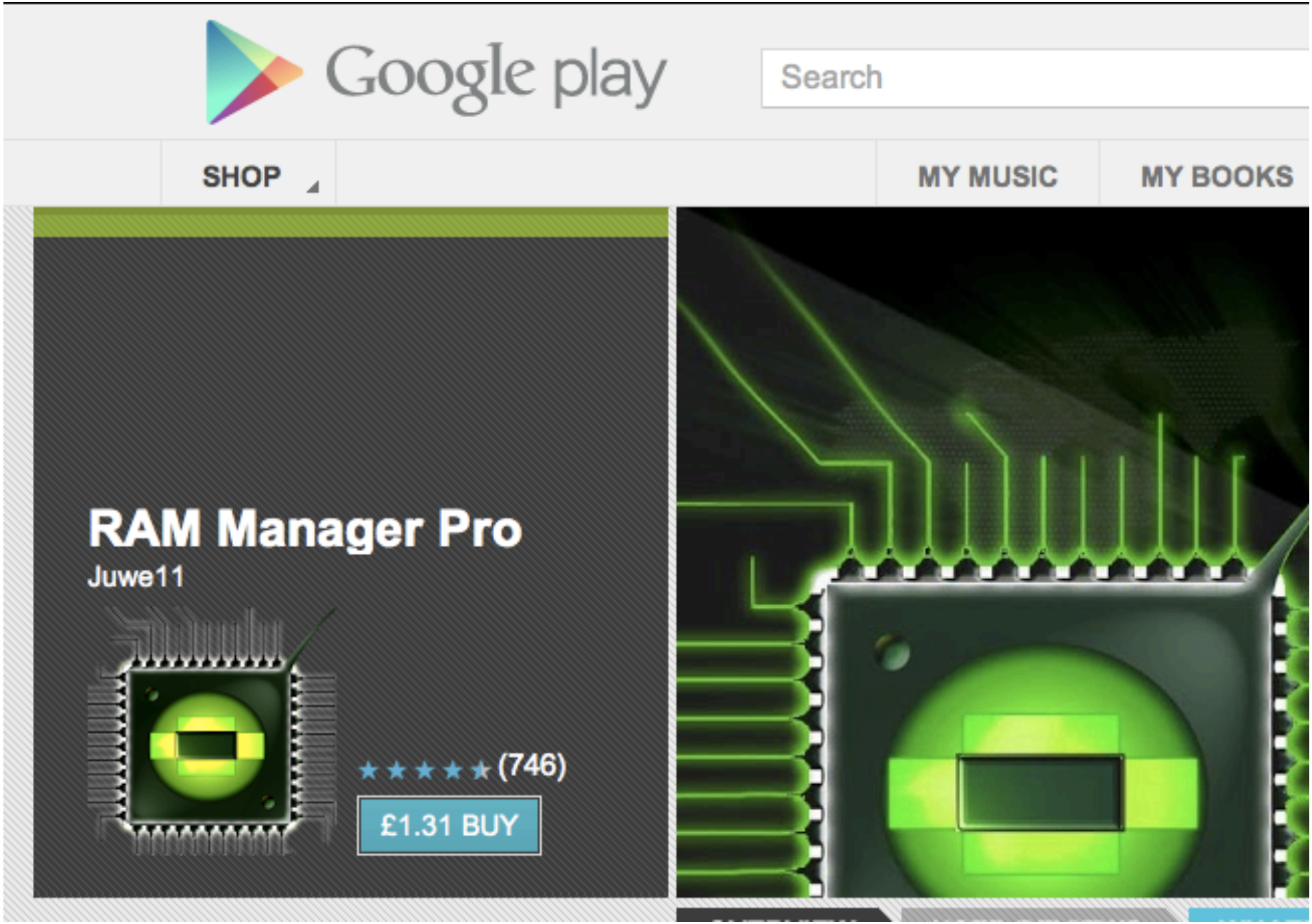
Requires *FULL ROOT* (aka S-OFF or NAND unlock) and *BUSYBOX*. Please use Busybox version 1.17.x as 1.18.x compile is broken!

Allows to change the Dalvik VM heap

A Google User - April 1, 2012 - HTC Desire HD with version 2.4.1

★★★★★ **Cool tool**

The phone is more stable now . Increased the vm heap size from 32 to 40 , battery life is superb , and I have to reset once at week . That is pretty awesome!!! Thanks Inspire 4G , running energy Rom

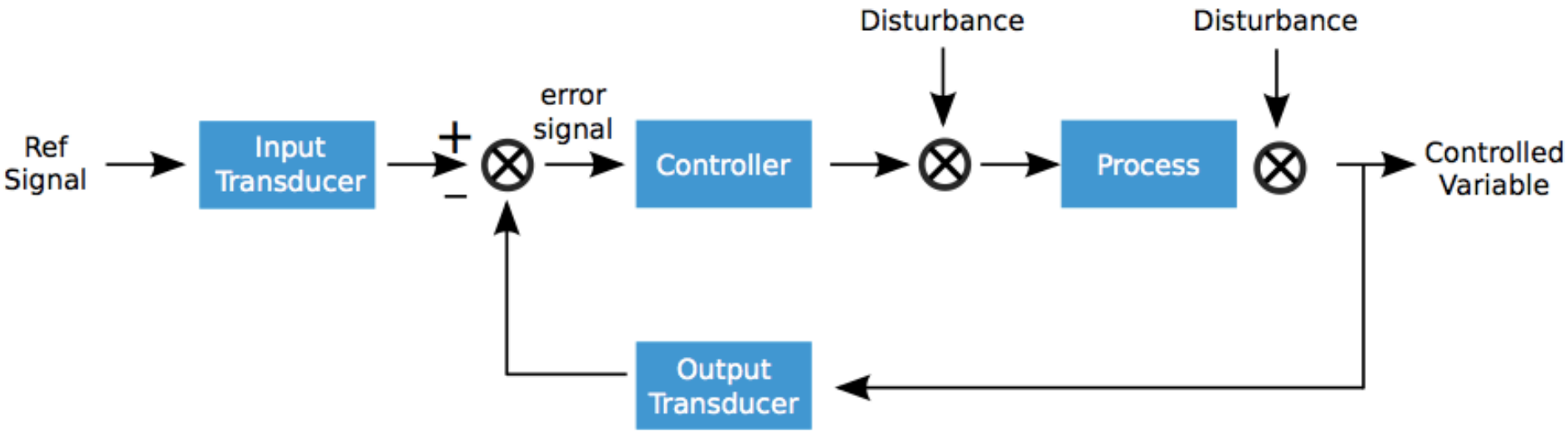

Google play
Search
SHOP
MY MUSIC
MY BOOKS
RAM Manager Pro
Juwe11
(746)
£1.31 BUY

Problems with existing approaches

- based on improvised heuristics
  - sometimes incorrect
  - need retuning
- generally conservative (prefer to grow, slowly)
- not goal-oriented
- generally stateless

Control Theory 101

Cruise Control


Disturbance
Disturbance
error
signal
Ref
Signal
Input
Transducer
+
−
Controller
Process
Controlled
Variable
Output
Transducer

PID Controller

control signal

gain

error

setpoint

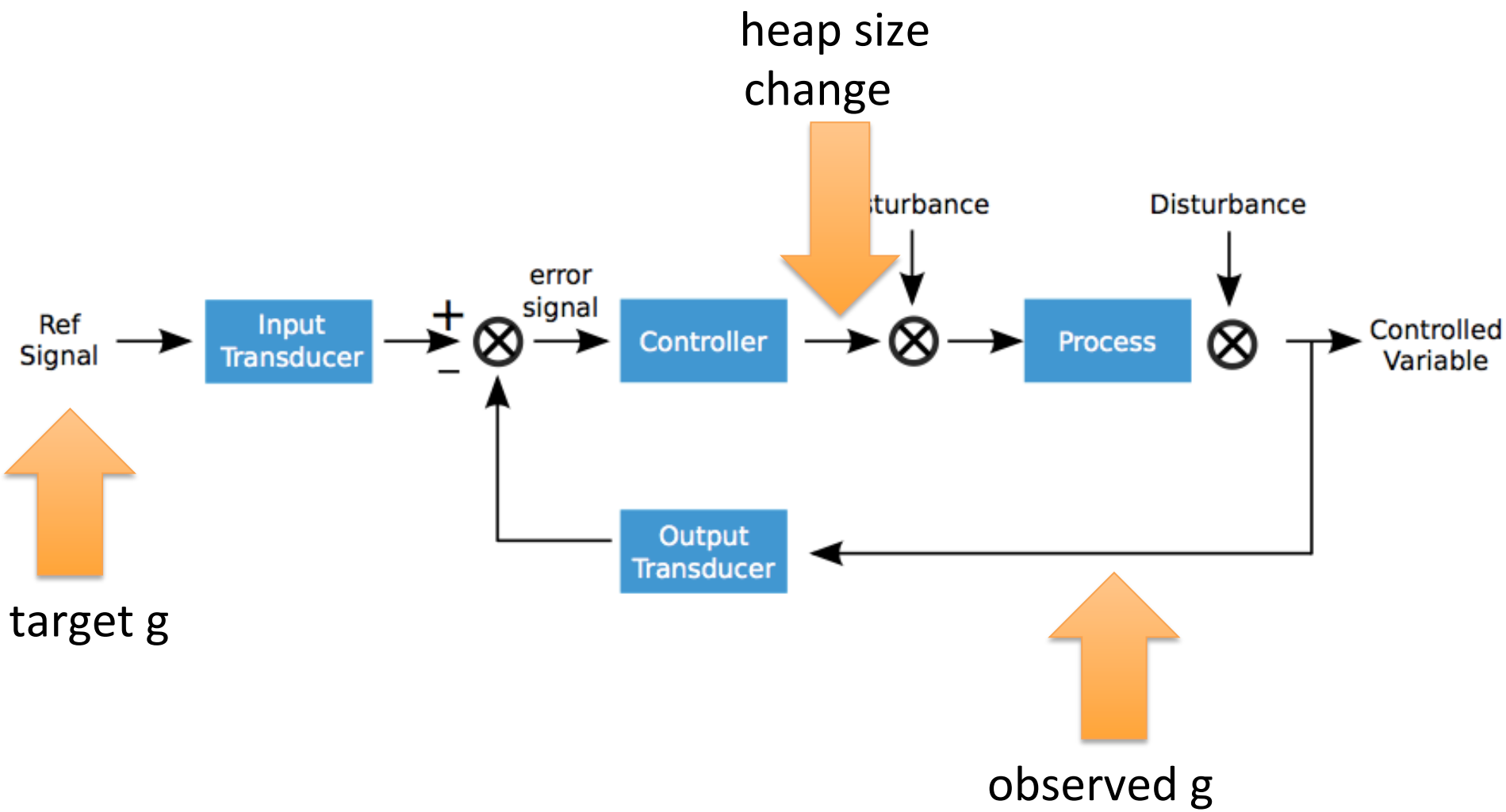
$$u(t) = K_c \left(\epsilon(t) + \frac{1}{T_i} \int_0^t \epsilon(t)\, dt + T_d \frac{d\epsilon(t)}{dt} \right) + b$$

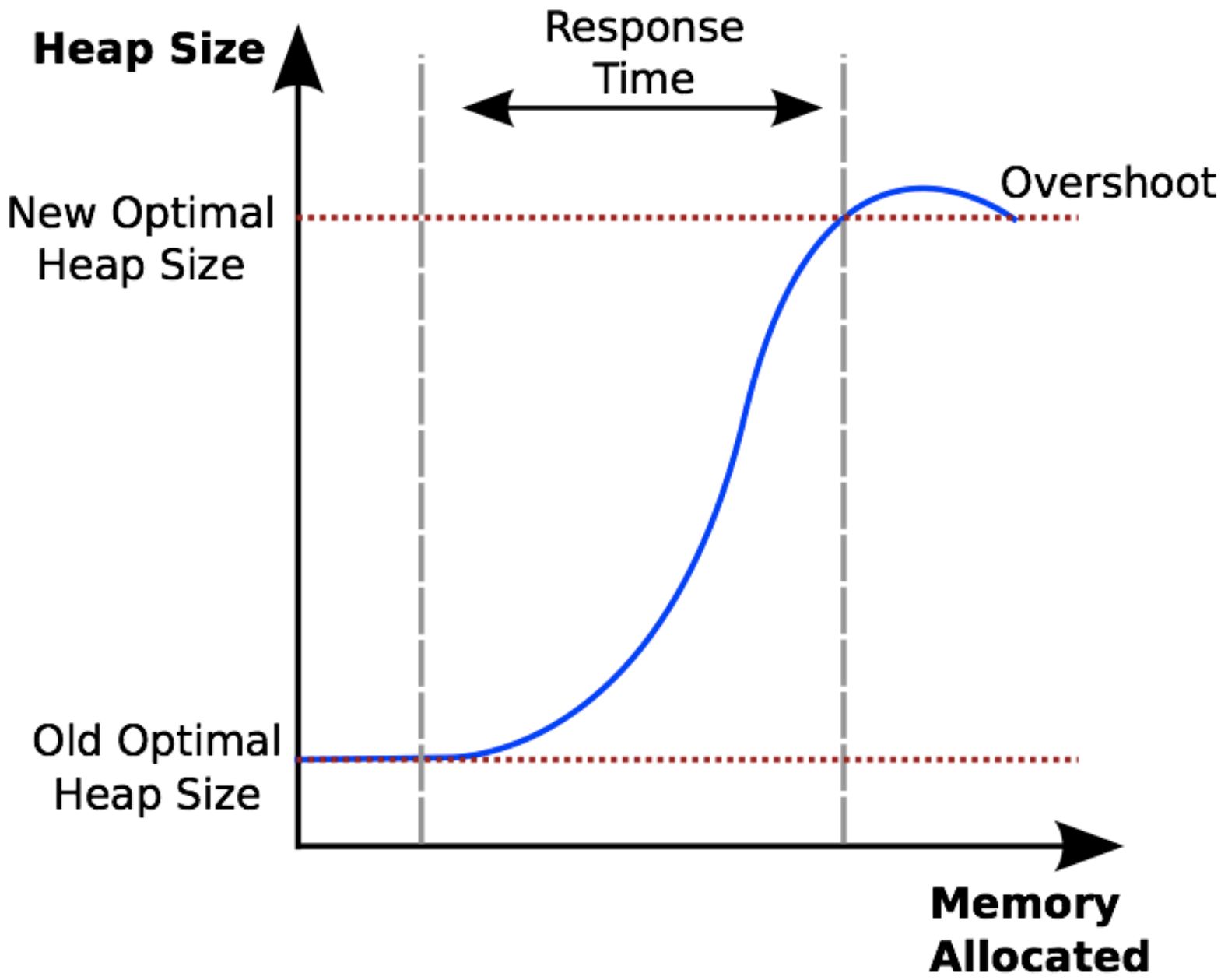
proportional term

integral term

derivative term

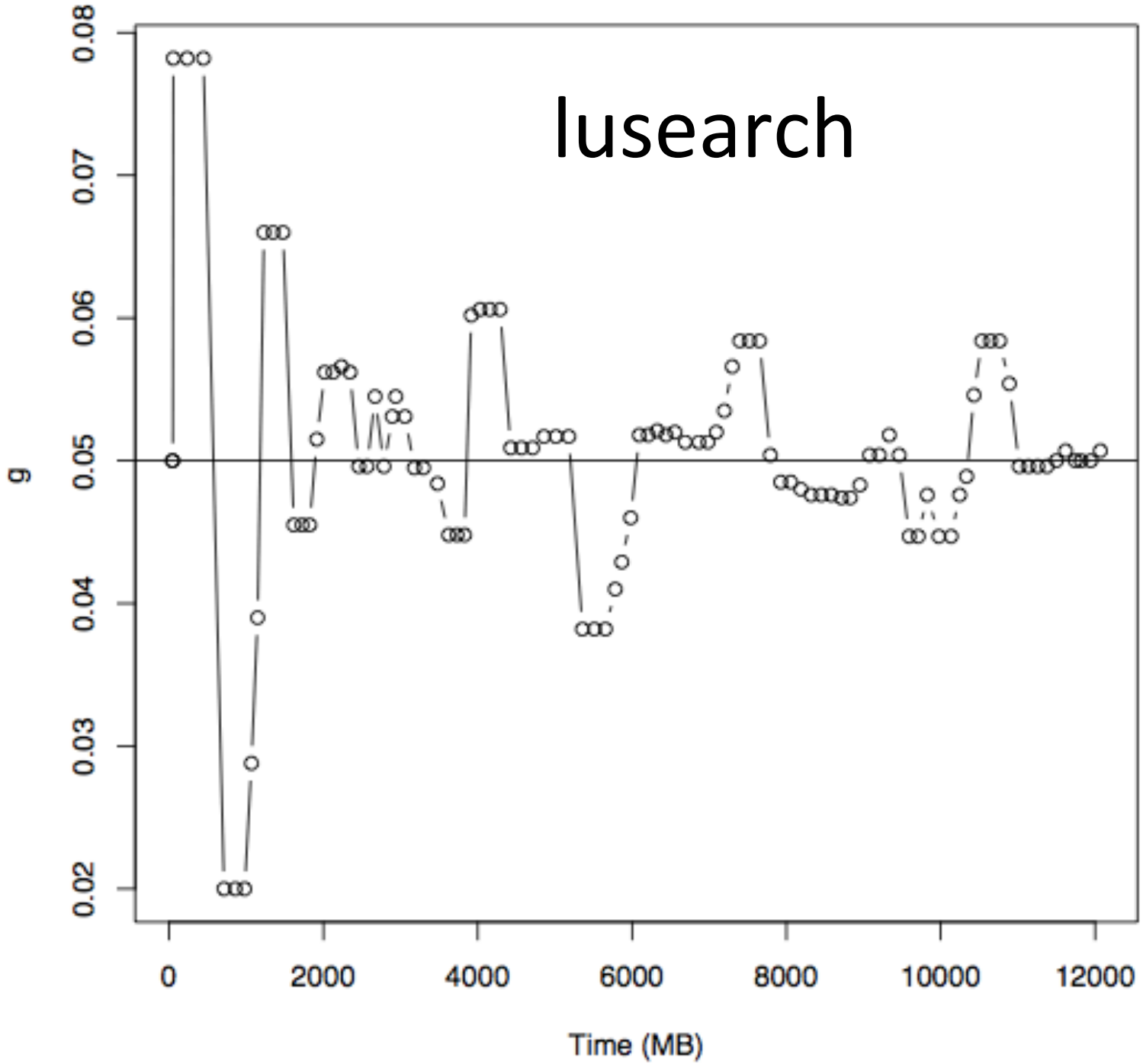
our approach

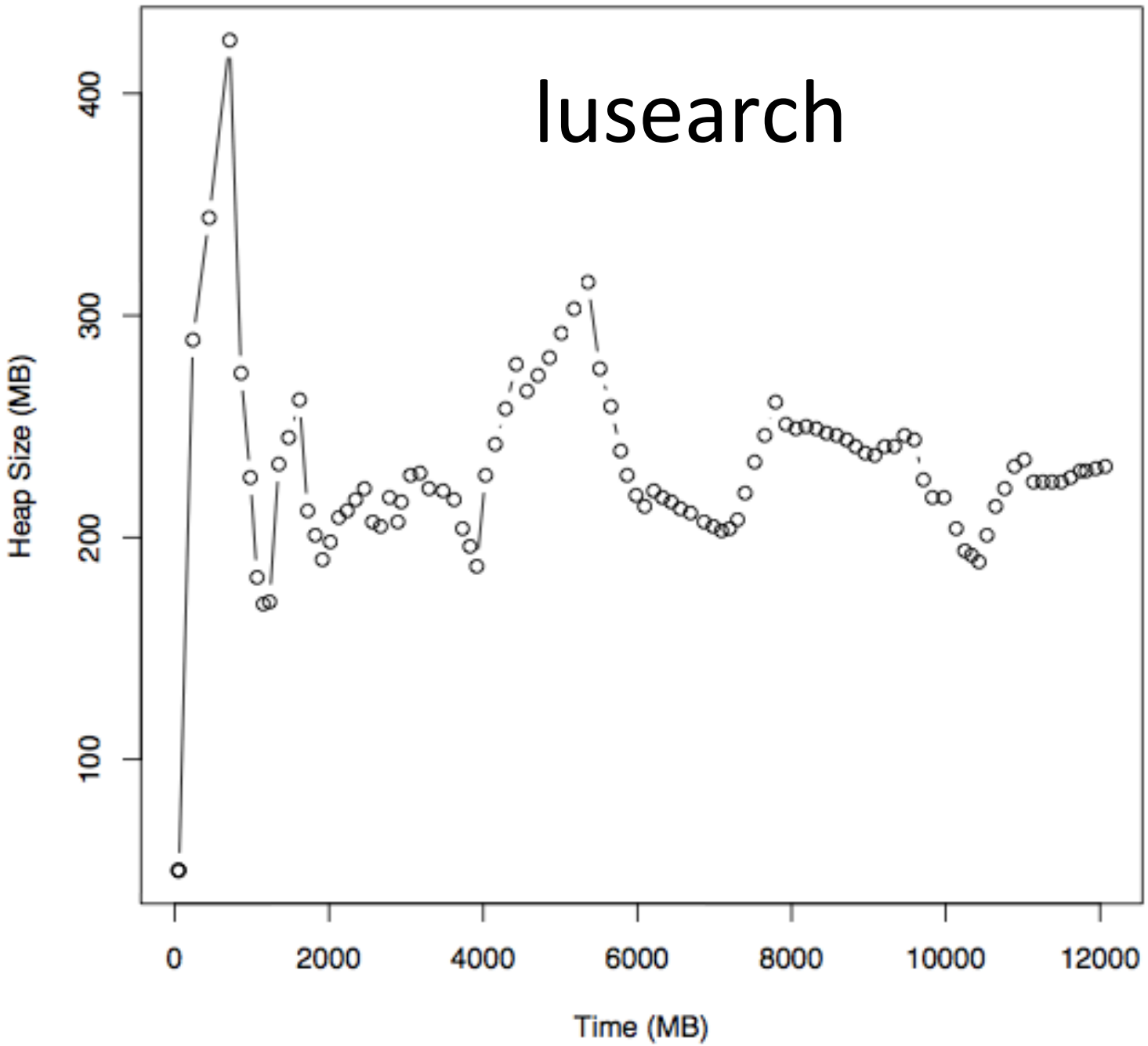

heap size
change
sturbance
Disturbance
Ref
Signal
Input
Transducer
+
−
error
signal
Controller
Process
Controlled
Variable
Output
Transducer
target g
observed g

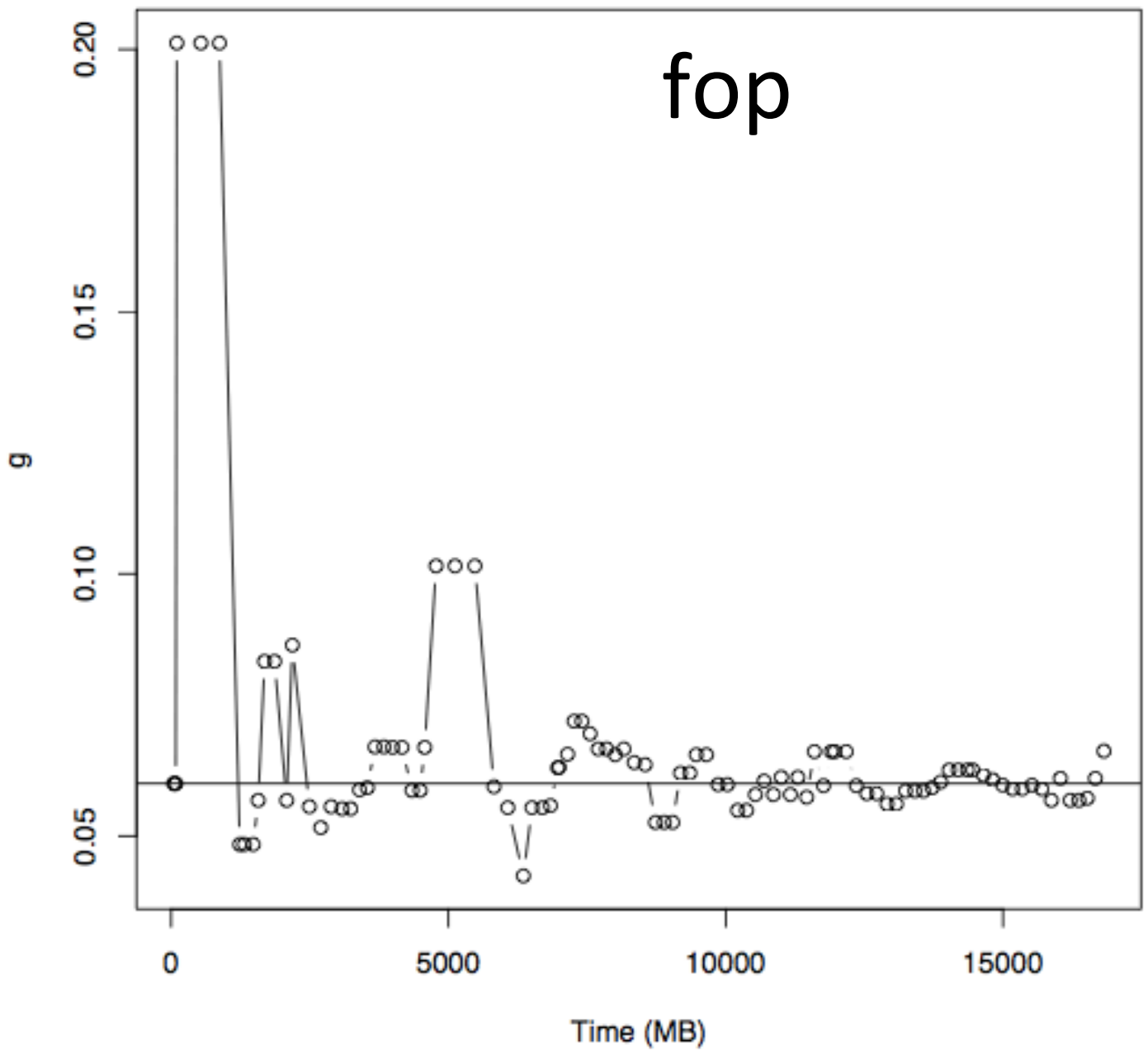

Heap Size
Response
Time
Overshoot
New Optimal
Heap Size
Old Optimal
Heap Size
Memory
Allocated

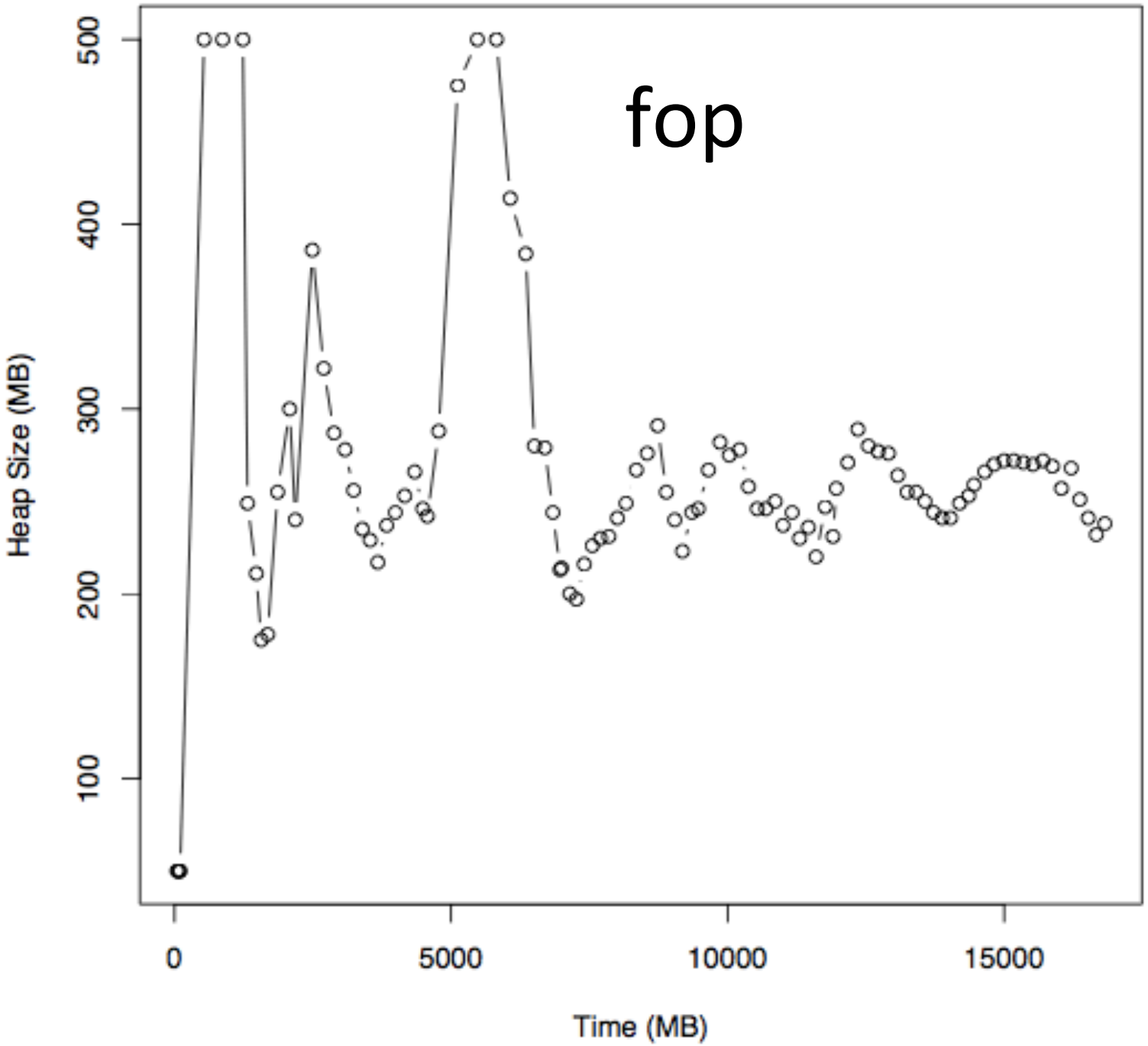
Experiments

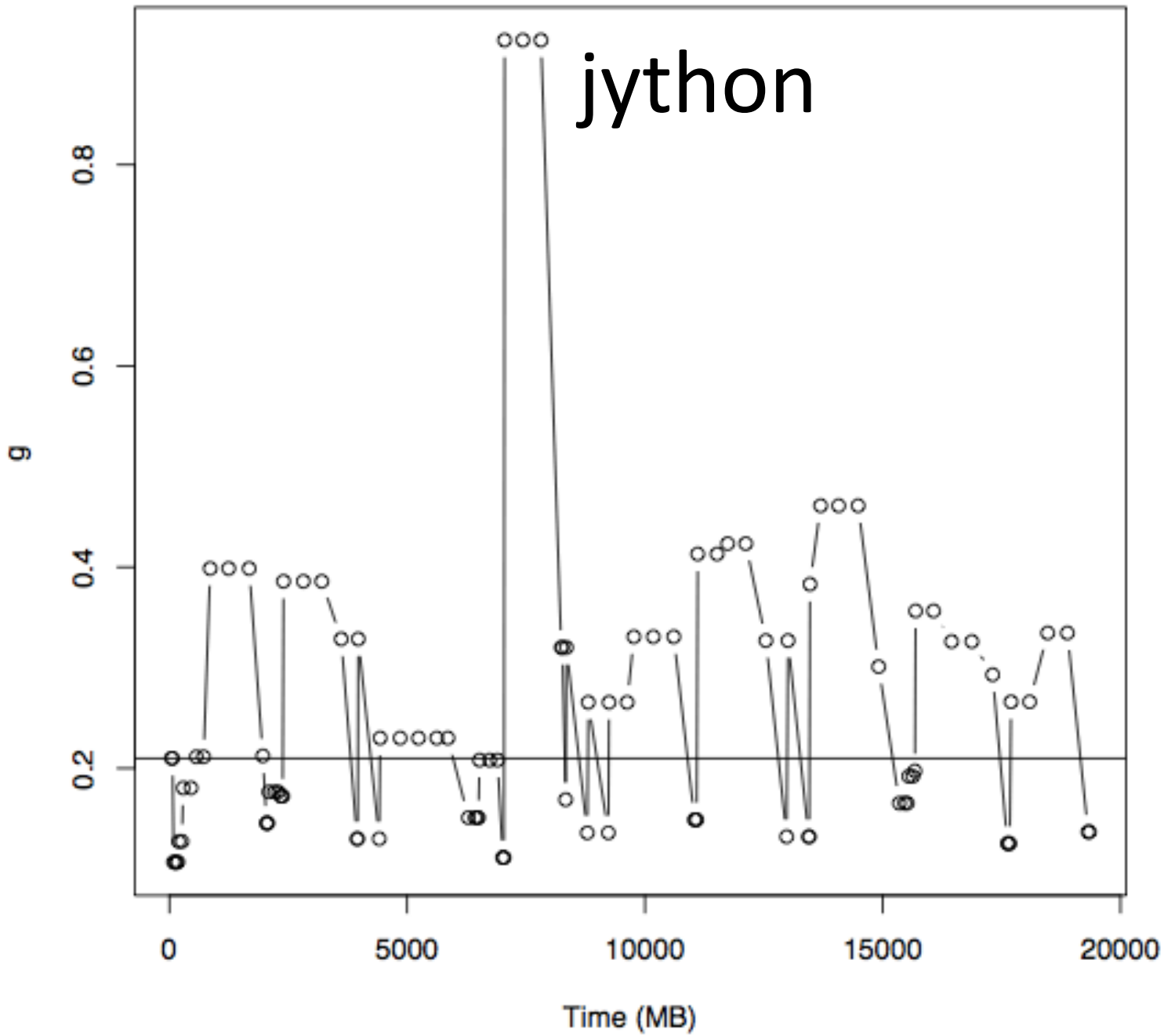
- Determine appropriate targets (run benchmarks in default VM)
- Tune controller (parameters)
- Run benchmarks with tuned controller
- Run a phased benchmark

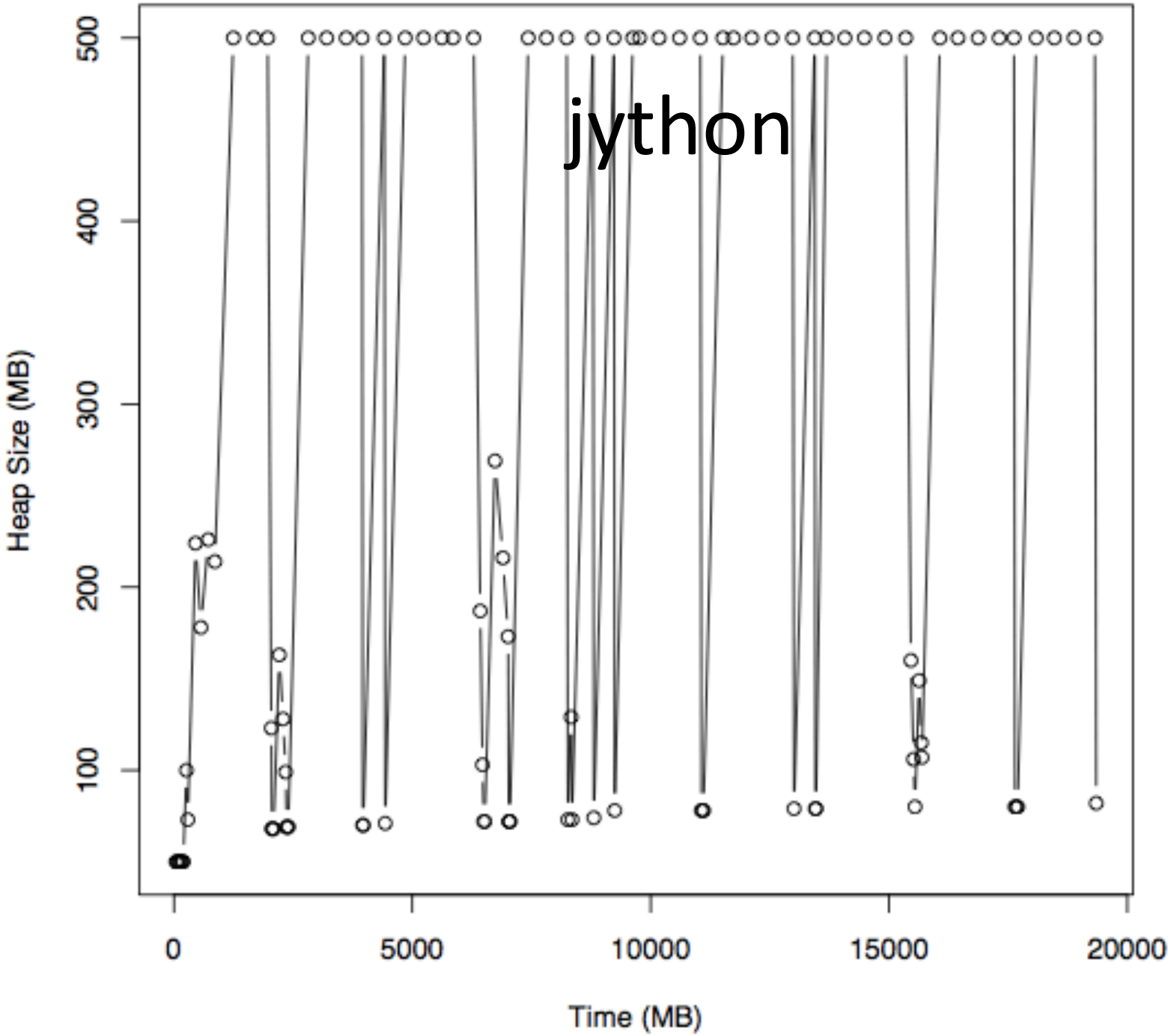

lusearch
g
0.02
0.03
0.04
0.05
0.06
0.07
0.08
0
2000
4000
6000
8000
10000
12000
Time (MB)

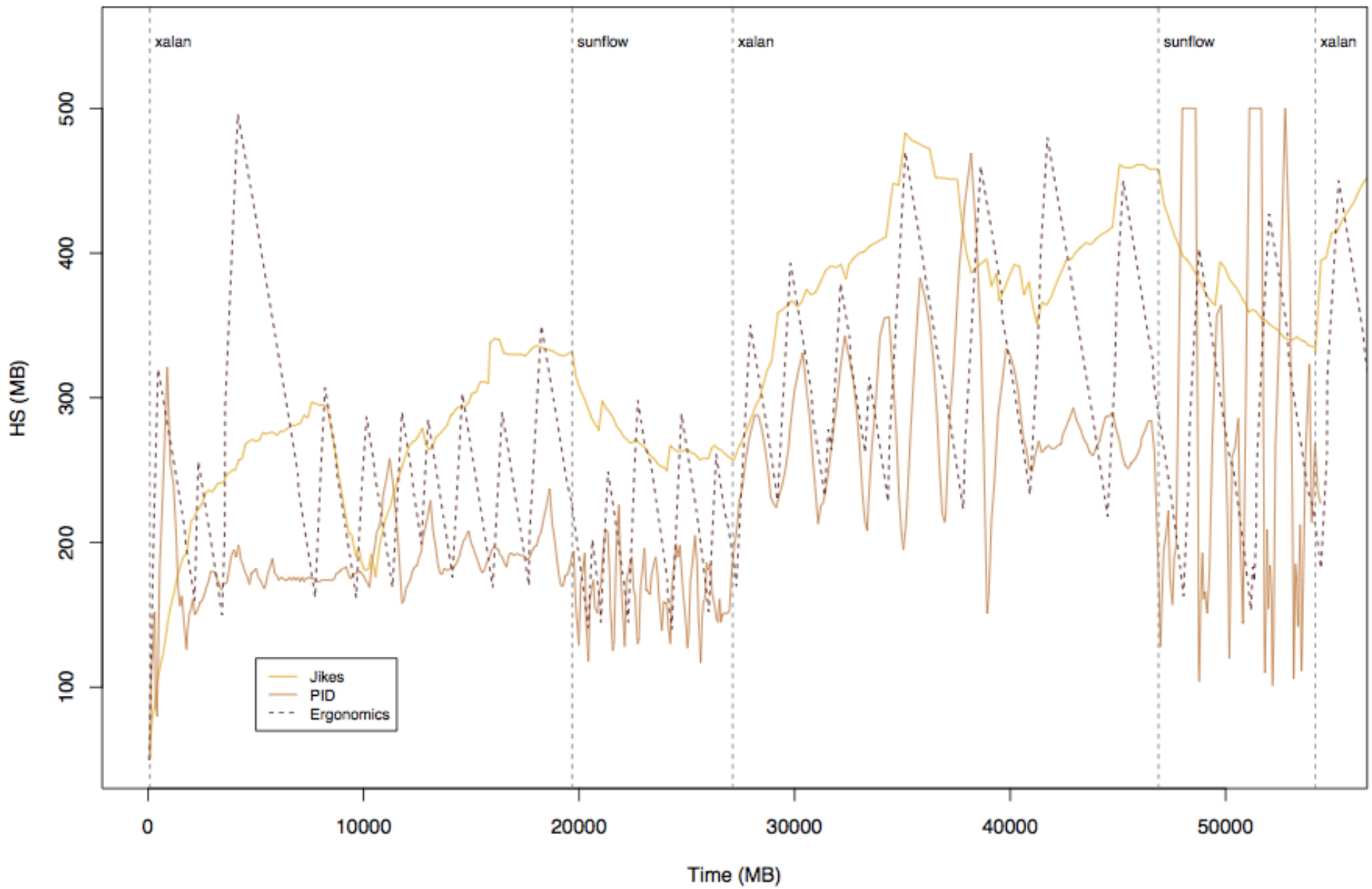

lusearch
Heap Size (MB)
100
200
300
400
0
2000
4000
6000
8000
10000
12000
Time (MB)

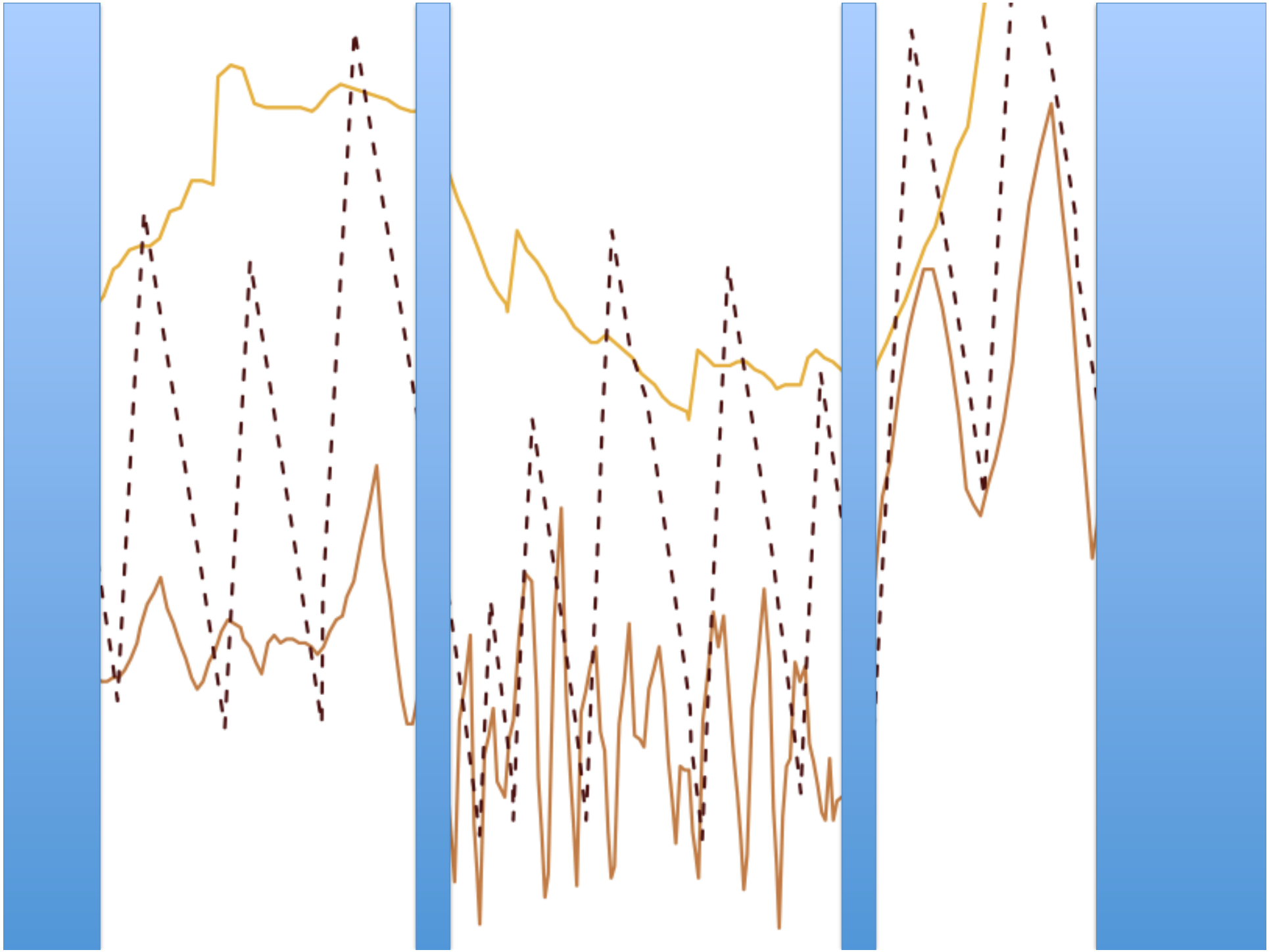

fop
g
0.05
0.10
0.15
0.20
0
5000
10000
15000
Time (MB)


fop
Heap Size (MB)
500
400
300
200
100
0
5000
10000
15000
Time (MB)


jython
g
0.2
0.4
0.6
0.8
0
5000
10000
15000
20000
Time (MB)


jython
Heap Size (MB)
500
400
300
200
100
0
5000
10000
15000
20000
Time (MB)


xalan
sunflow
xalan
sunflow
xalan
HS (MB)
100
200
300
400
500
Jikes
PID
Ergonomics
0
10000
20000
30000
40000
50000
Time (MB)



discussion

Pros and Cons of Control Theory for Heap Sizing

Conclusions

- Control theory is systematic approach
- Better than ad-hoc heuristics

- Requires careful tuning
- May be useful for data-center resource management