

Comp314: Project 2 (Gzip)

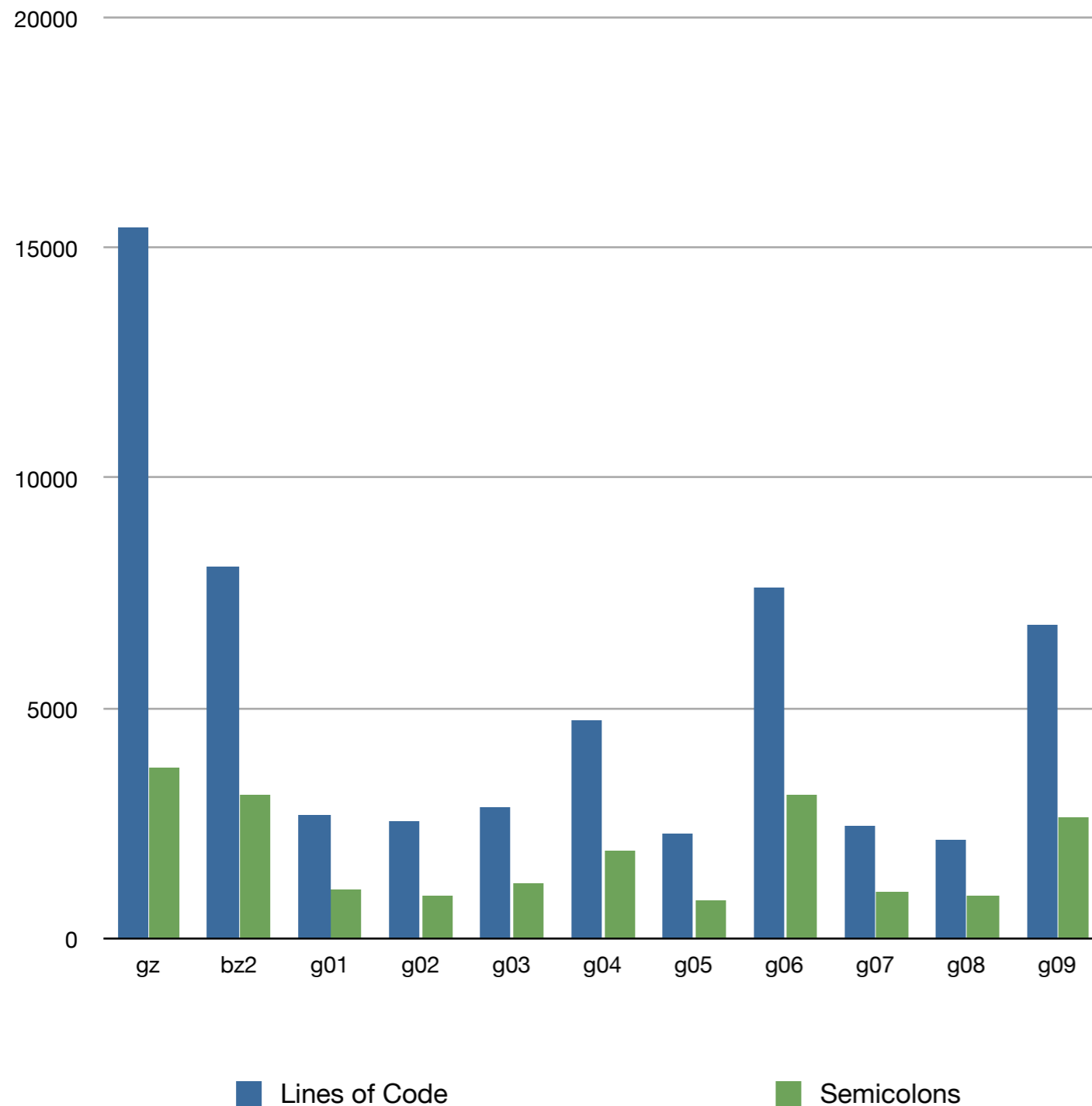
Post-mortem

March 20, 2008

You made it!

- Probably one of the harder projects you've ever done
- Will sound terribly impressive if/when you go on interviews

Lines of code



- gz = gzip 1.3.12
- bz2 = bzip2 1.0.5
- Most of you were much smaller (some of the benefit of writing in Java)
- But how well did you perform?

There were some problems

- g01: worked!
- g02: seemed to run forever
- g03: seemed to run forever
- g04: fast compression, slow decompression!
- g05: compression and decompression broken
- g06: doesn't compile (weird errors about *enum*)
- g07: seemed to run forever on compression / decompression crashed
- g08 seemed to run forever
- g09: seemed to run forever / decompression crashed (*ERROR - Invalid block type specified*)
- *We'll sort this out later...*

Performance numbers so far

- Speeds in MB/s
(on Dan's Mac mini)
- g01 compressing fastest
g07 decompressing fastest
- g04 having speed issues
- g07 competing with real gzip for
compression rates

	CTime	DTime	CSpeed	DSpeed	FileSize	CRatio
gzip	0.144	0.023	12.658	79.249	464,158	74.5%
bzip2	0.375	0.119	4.861	15.317	386,467	78.8%
g01	1.610	1.960	1.132	0.930	697,441	61.7%
g04	2.880	79.083	0.633	0.023	997,511	45.3%
g07	3.940	0.430	0.463	4.239	476,926	73.8%

Discussion

Usefulness of “official” specs?

RFCs?

Emily’s helpful GZip decoder?

Lecture on on how GZip-style compression worked?

Experimental findings

What did you learn?

What did you wish you knew going in?

GZip Decompression

Getting test data to drive your program?
Handling all the different block types?

GZip Compression

Code reuse from GZip decompression?

Handling all the different block types?

Compression / runtime performance tradeoffs?

Advice for next year's students / instructors?

Would you have preferred a simplified zip-like algorithm?

Benefits of doing “real” GZip?

Sneaky things you learned without realizing it

Keeping an index on the window of processed data similar to other search problems (e.g., iTunes search interface)

Even “official” real-world specs often lack the detail that you would prefer

Good vs. bad performance comes as much from your algorithms as from careful hand-tuning or from a good compiler / language system