An Empirical Exploration of Regularities in Open-Source Software Lexicons

by Derrin Pierret and
Denys Poshyvanyk

SEMERU

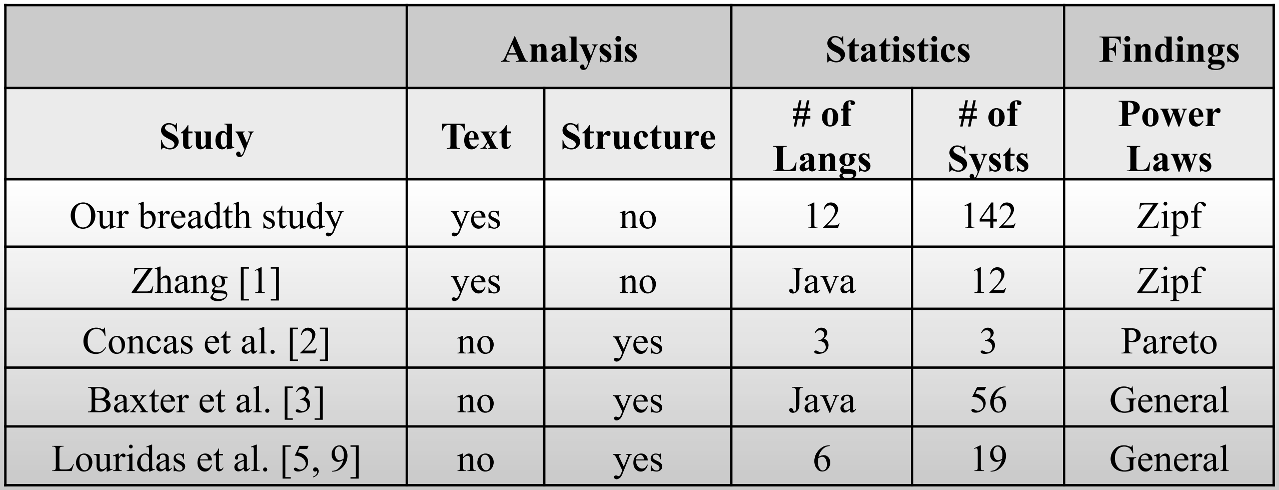
WILLIAM
& MARY

Introduction

- Lexicon: vocabulary used in a program
    - identifiers, keywords, symbols, etc.

- Lexicon metrics could distinguish programs

Recent Studies on Power Laws



	Analysis		Statistics		Findings
Study	**Text**	**Structure**	**# of Langs**	**# of Systs**	**Power Laws**
Our breadth study	yes	no	12	142	Zipf
Zhang [1]	yes	no	Java	12	Zipf
Concas et al. [2]	no	yes	3	3	Pareto
Baxter et al. [3]	no	yes	Java	56	General
Louridas et al. [5, 9]	no	yes	6	19	General

Background: Zipf-Mandelbrot Law

$$f = \frac{C}{(r + \beta)^{\alpha}}$$

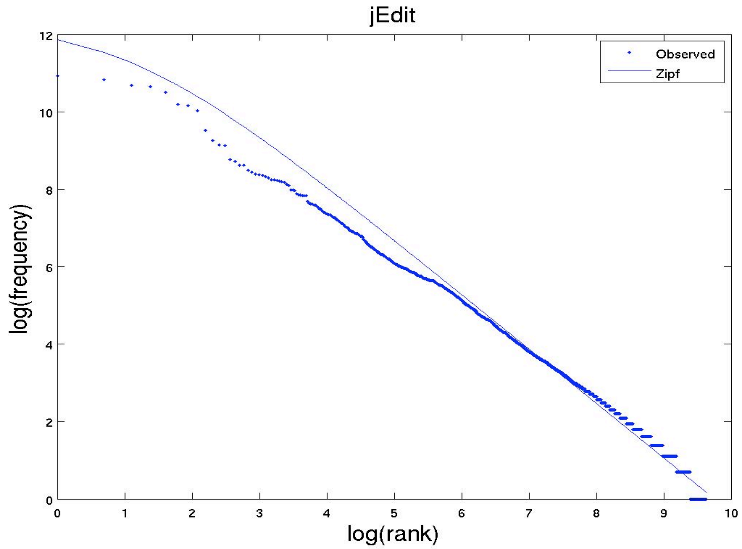
Where:

f = word frequency

r = word rank

(1st most common,
2nd, 3rd, etc)

α, β, C = constant for fitting



- Research Questions
  - Is equation reliable?
  - What do constants reflect?

Goals

How well does Zipf's Law fit...

- token distributions amongst...
  - projects
  - languages
  - paradigms

- word distributions amongst...
  - documentation
  - bug reports

Case Study Design

4 Paradigms, 3 Languages each:

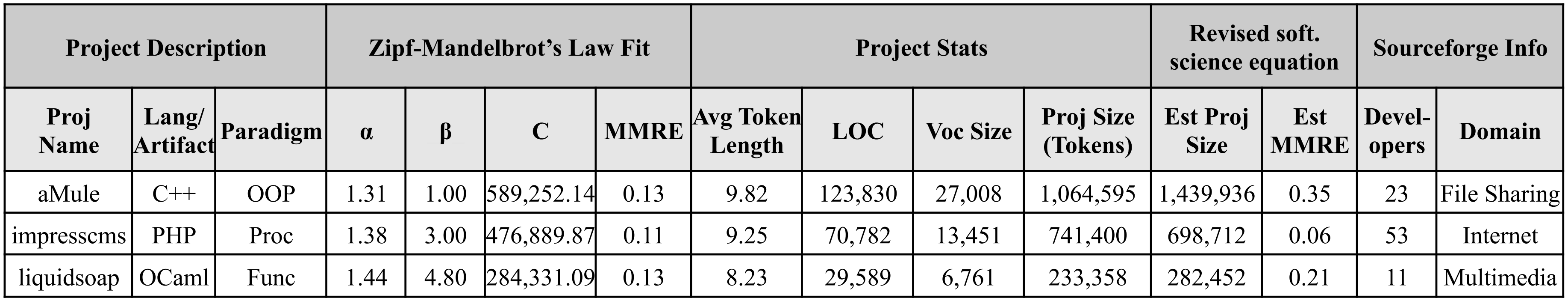
Object Oriented:	Java, C++, Smalltalk
Imperative:	Matlab, C, PHP
Markup:	HTML, XML, TeX
Functional:	Haskell, Scheme, OCaml

Roughly 10 programs per language

141 programs total, 9 non-program artifacts

Results

Sample of projects with gathered statistics



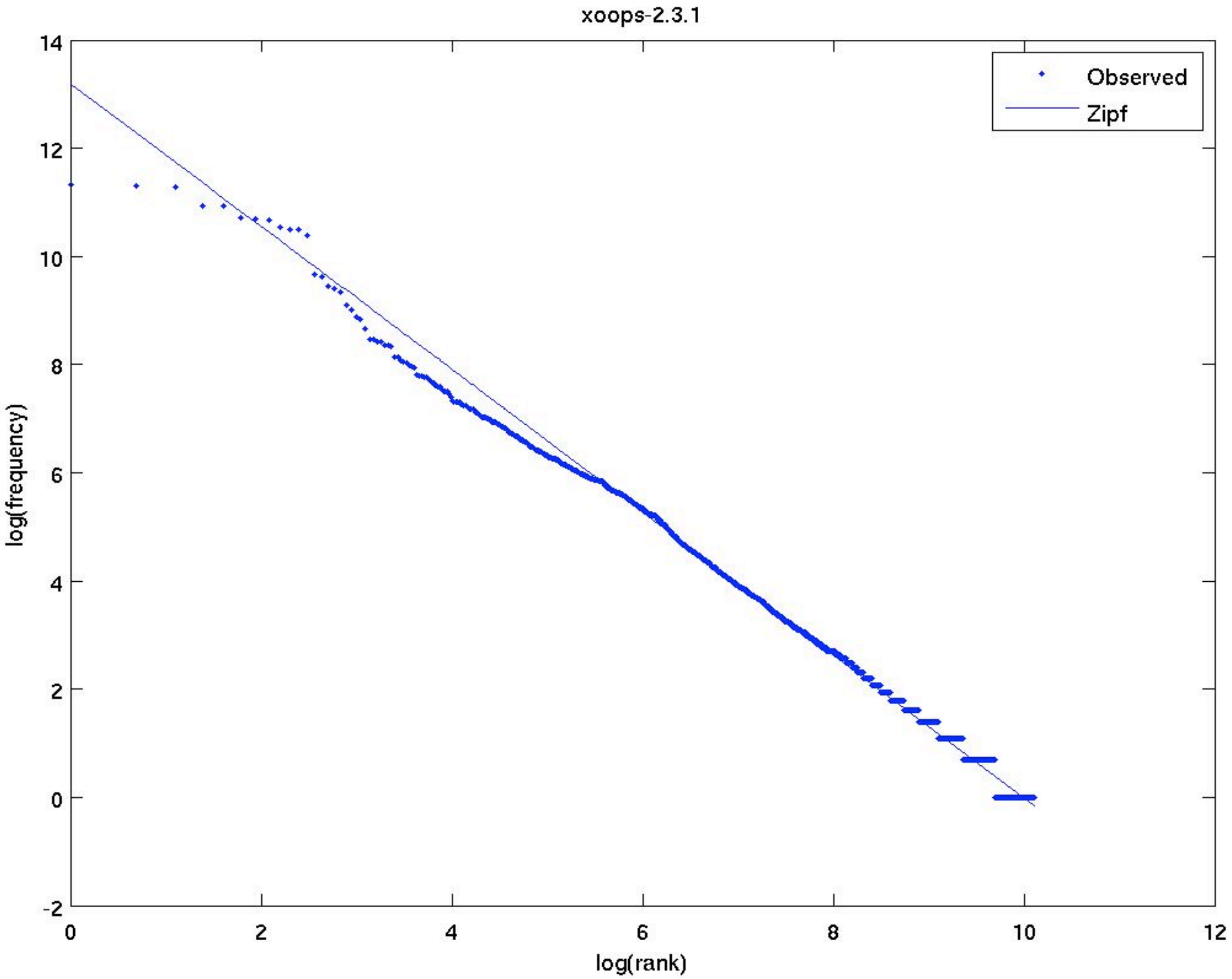
Project Description			Zipf-Mandelbrot's Law Fit				Project Stats				Revised soft. science equation		Sourceforge Info	
Proj Name	Lang/ Artifact	Paradigm	α	β	C	MMRE	Avg Token Length	LOC	Voc Size	Proj Size (Tokens)	Est Proj Size	Est MMRE	Devel- opers	Domain
aMule	C++	OOP	1.31	1.00	589,252.14	0.13	9.82	123,830	27,008	1,064,595	1,439,936	0.35	23	File Sharing
impresscms	PHP	Proc	1.38	3.00	476,889.87	0.11	9.25	70,782	13,451	741,400	698,712	0.06	53	Internet
liquidsoap	OCaml	Func	1.44	4.80	284,331.09	0.13	8.23	29,589	6,761	233,358	282,452	0.21	11	Multimedia

"Good Fit": Mean Magnitude of Relative Error (MMRE) < 0.25

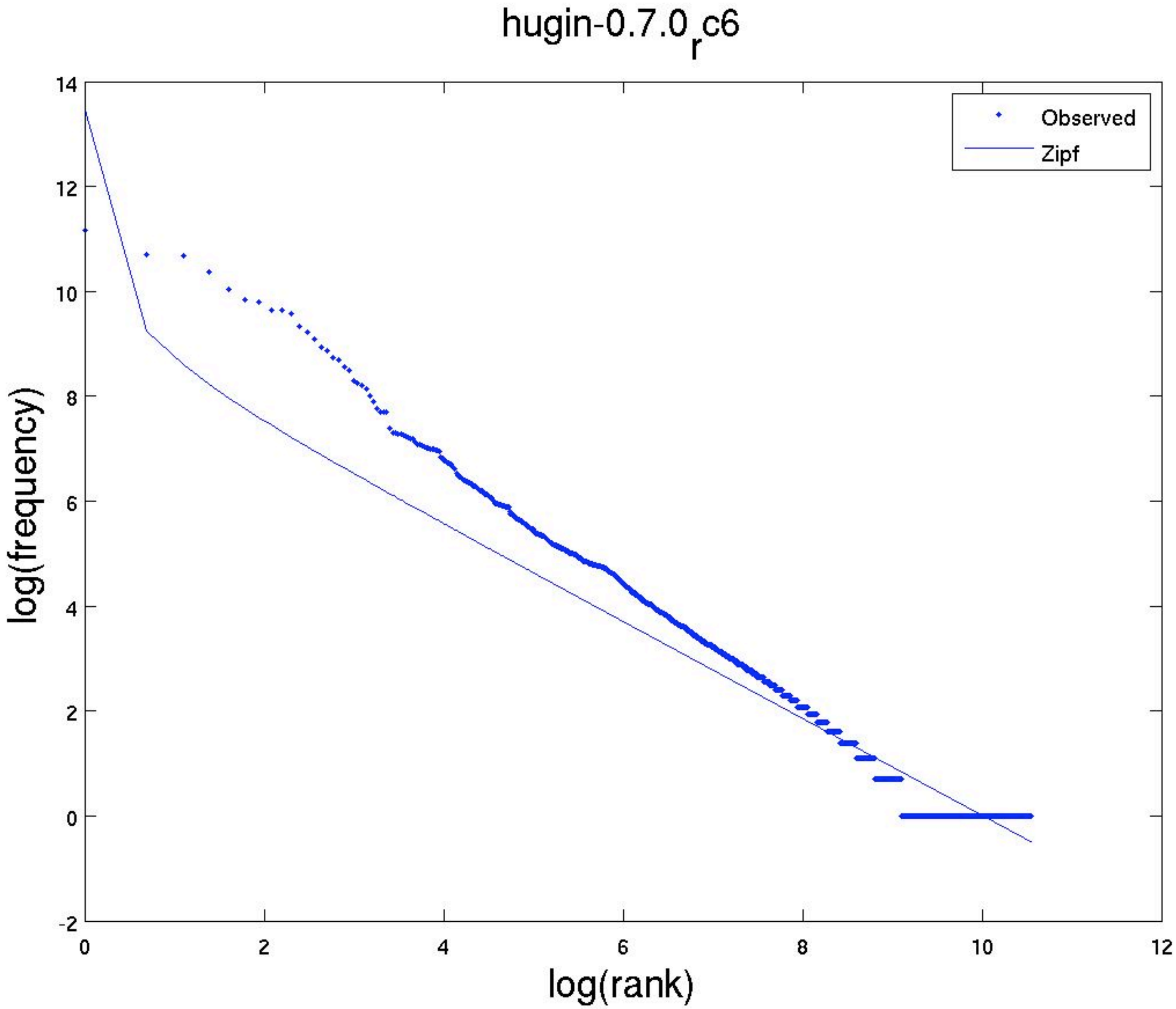
- ○ MRE: % of the actual value that the estimate is off by
  - ▪ | actual = 10,000 | estimate = 8,000 |
  - ▪ | 2000 = 10,000 * 0.2 | MRE = 0.2 |
- ○ MRE collected for each token in program
- ○ Project MMRE is average of the token MREs

Most projects met criteria for "good fit"

Good Fit (MMRE = 0.11)



Bad Fit (MMRE = 0.32)



Average MMRE Values

Object Oriented	Java	C++	Smalltalk
0.17	0.18	0.17	0.17
Imperative	Matlab	C	PHP
0.17	0.18	0.17	0.17
Markup	HTML	XML	TeX
0.18	0.21	0.19	0.15
Functional	Haskell	Scheme	OCaml
0.16	0.16	0.16	0.17
Non-Source	Documentation		Bug Reports
0.22	0.21		0.23

Revised Program Length Estimation

- Halstead: Software Science equation
  - Links vocabulary size with program size

- Zhang: Saw inaccuracy with Halstead on Java code
  - Saw Zipf-Mandelbrot work for Java tokens
  - Devised new length estimation equation from Zipf

$$N\hat{} = C\frac{(n+\beta)^{1-\alpha}-(1+\beta)^{1-\alpha}}{1-\alpha}$$

Where:

N^ = program length

n = vocabulary length

α, β, C = Zipf constants

- Estimations were only sometimes accurate
  - Good estimation for 84 out of 141 projects (60%)

Conclusions

Zipf's Law is able to describe token distributions well for:

- Project Source, Languages, Paradigms
- Documentation, Bug Reports

Program size estimation equation needs further investigation

Questions left to explore:

- Meaning behind Zipf goodness of fit or generated constants
- Where types of tokens appear in distribution

Online Appendix:

http://www.cs.wm.edu/~dpierret/zipf-appendix.html