Go Manatee Go Why we want to drop our core module

Pavel Rychlý



pavel.rychly@sketchengine.co.uk, pary@fi.muni.cz

7th Sketch Engine Workshop Portoroz, May 23, 2016

Sketch Engine Structure

- Bonito Python
 - user interface
 - remote API
 - high level command line tools
- Manatee C++
 - corpus encoding
 - query evaluation
 - computing statistics
 - finlib disk access

Manatee is hard to maintain

- several modules 18 years old
- most of the code 15 years old
- C++ with templates
- only a few developers can add new features

Manatee is slow for huge corpora

- Manatee is very fast even for huge corpora
- asynchronous query processing (display the first page before computing the whole result)
- complex queries could take minutes or more
- huge corpora mean huge concordance
- whole-concordance operations are slow (sort, frequency distribution)

Can it be better? (on one processor)

Not much

- use C instead of C++
- better (more) compression/indexes
- it is hard to process 100,000 parts of 100 GB file
- users don't want to wait 10 minutes

Can it be better? (on one processor)

Not much

- use C instead of C++
- better (more) compression/indexes
- it is hard to process 100,000 parts of 100 GB file
- users don't want to wait 10 minutes5 minutes is not better
- go parallel (more processors, more computers)

Go (golang)

- new programming language from Google
- to make programmers more productive
- Go is expressive, concise, clean, and efficient
- provides easy concurrent (parallel) programming

Go (golang)

- new programming language from Google
- to make programmers more productive
- Go is expressive, concise, clean, and efficient
- provides easy concurrent (parallel) programming
- manatee-go whole manatee reimplemented in Go
- joint project with NLP Centre, Masaryk University (Radoslav Rábara)

Vertical parallelization (queries)

- query translated into tree
- each part of a query on separate processor

Vertical parallelization (queries)

- query translated into tree
- each part of a query on separate processor
- benchmark: set of very hard queries
- one processor:
 - C++: 4h 29m
 - Go: 2h 28m (1.8× faster)

Vertical parallelization (queries)

more procesors:

Num of CPU	total time	speedup	
1	2:27:39	1.00	
2	1:23:46	1.76	
3	1:03:17	2.33	
4	0:51:35	2.86	
5	0:45:57	3.21	
6	0:43:39	3.38	
7	0:42:25	3.48	
8	0:45:06	3.27	

- corpus divided into small parts
- each part on separate computer

- corpus divided into small parts
- each part on separate computer
- benchmark: 13 billion corpus (enTenTen12)
- experiments with up to 130 parts, up to 70 computers
- query, concordance sort, frequency distribution

Performance of the sort operation (query, sort, and retrieve the first page, in seconds) cluster of 65 computers, 130 processors

Query	C++	Go on cluster	Go on 1
[word="Gauss"]	26.8	3.0	26.8
[word="recurrence"]	180.1	1.6	52.0
[word="enjoyment"]	410.0	1.3	123.9
[word="test"]	492.7	2.4	158.3
[word="said"]	266.6	3.4	100.7
[word="a"]	>1 hour	27.2	> 1 hour
[word="the"]	> 1 hour	56.5	> 1 hour

Performance of frequency distribution (following token), in seconds cluster of 65 computers, 130 processors

Query	C++	Go on cluster	Go on 1
[word="Gauss"]	17.0	0.35	12.7
<pre>[word="recurrence"]</pre>	159.3	0.32	31.8
[word="enjoyment"]	361.9	0.58	101.5
[word="test"]	482.9	3.66	138.3
[word="said"]	147.4	5.28	67.2
[word="a"]	576.3	15.41	136.8
[word="the"]	1273.0	28.85	621.9

Scalability of sort operation (query, sort, and retrieve the first page)
Fixed size on each computer (200 million tokens),
10 computers = 2 billion corpus, 70 computers = 14 billion corpus

0	Number of servers						
Query	10	20	30	40	50	60	70
[word="Gauss"]	0.293	0.570	0.828	0.813	1.020	3.239	4.089
[word="recurrence"]	0.652	0.644	0.750	0.869	1.073	1.702	2.177
[word="enjoyment"]	0.927	0.938	0.947	0.971	1.372	1.297	1.637
[word="test"]	2.105	1.929	2.054	2.058	2.657	2.366	2.483
[word="said"]	2.546	2.300	2.843	2.988	3.223	3.060	3.131
[word="a"]	22.841	23.263	23.772	24.049	24.056	26.086	30.881
[word="the"]	51.606	52.394	52.795	54.826	53.282	57.549	55.280

Conclusions

- manatee-go is easy to maintain
- runs faster on single computer
- scalable to a cluster of many computers