Same Results: Boa: A Language and Infrastructure for Analyzing Ultra-Large-Scale Software Repositories

Summary: The authors developed a domain specific language called Boa to ease the process of querying information from many open-source projects (e.g., from SourceForge or GitHub). They evaluated Boa compared to Java by letting participants write scripts in each language. The scripts were analyzed regarding lines of code and execution times.

- Independent variable: Language (2 levels, operationalized as Boa and Java)
- Tasks: 21 typical mining tasks of four different areas:

  A Programming language (3 tasks, A1 to A3)

  B Project management (11 tasks, B1 to B11)

  C Legal (2 tasks, C1 and C2)

  D Platform/environment (5 tasks, D1 to D5)

  in three categories:

  - Metadata only (Tasks A1, A2, B1, B2, C1, C2, D1 to D5)
  - Data from one or few revision (Tasks A3, B3 to B5)
  - Data from most of the revisions (Tasks B6 to B11)

- Dependent variables:

  1. Lines of code per language and task [metric scale]
  2. Execution time per language (Java: repositories cached) and task [metric scale]
  3. Execution time per language (Java: repositories remotely accessed) and task [metric scale]

- Results:

  1. 8 to 18 fewer lines of Boa code compared to Java code
  2. 8 to 250 times of speed up for Boa code compared to Java code (repositories cached)
  3. 459 to 2364 times of speed up for Boa code compared to Java code (repositories remotely accessed)

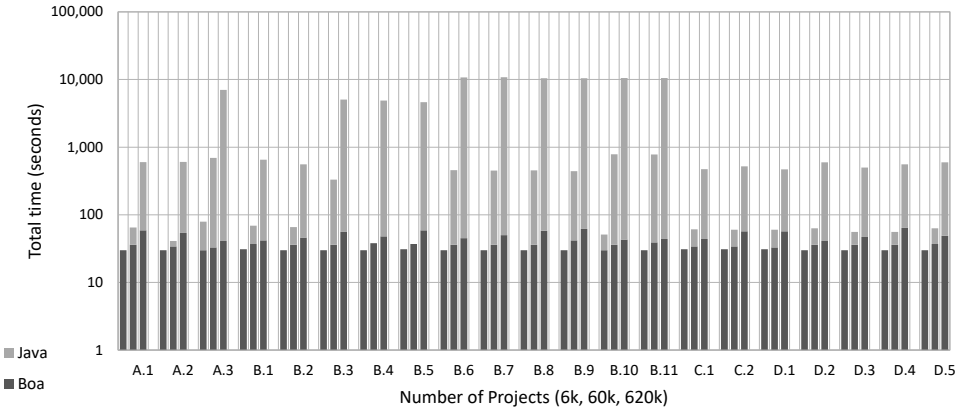
Participants created one script for each task, so for each dependent variable, there was one data point (e.g., 89 Lines of code for Task A3 for the Java script). Hence, it was not possible to conduct a significance test or state a null hypotheses. Instead, the authors reported the differences in terms of multiples (e.g., the Java script consist of 9x as much lines of code as the Boa script). In Table 1, we show the according data. In their discussion, the authors highlighted the maximum benefit, especially when comparing the remote access of Java scripts to the Boa scripts, they speak of an improvement of more than 2 000 times. Aggregating the data according to the three categories of the tasks defined by the authors, this number reduces to 1 800, and aggregating further, it becomes 1 300, which still can be seen as considerable improvement, without highlighting the maximum benefit. Thus, in this case, aggregation would not fundamentally alter the conclusions that can be drawn (i.e., that Boa scripts are shorter and faster), but the numbers reflect the average case, instead of the maximum case. Especially when it comes to evaluating a newly developed approach, it is important to be more objective and consider the aggregated data, instead of promoting the best values.

The authors also analyzed the scalability of the approaches, which also happened task-wise. In Figure 1, we show the original figure (Fig. 15 in the original paper), and additionally the aggregated version. Again, the information point to the same conclusion, that is, that for 60k projects and upwards, Boa is faster. The non-aggregated plot shows the additional information that this is true for all tasks, and not, for example, better for 80 % of the tasks and considerable worse for 20 % of the tasks. By contrast, the aggregated plot shows the data in a more concise way. Thus, depending on the underlying data, one could decide whether to aggregate or not to aggregate the data. In this case, the aggregated plot does not hide any irregularities in the data, so it might be a good choice.

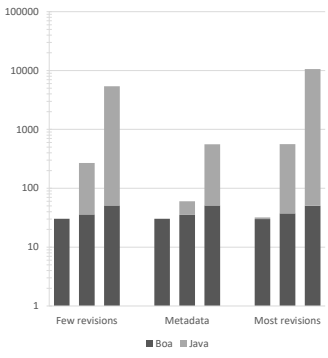
Table 1: Raw data for evaluation of Boa. Gray cells contain values computed by us.

Category	Task	Java-LOC	Boa-LOC	Diff-LOC	Java-Time (cached)	Boa-Time	Diff-Time (cached)	Java-Time (remote)	Diff-Time (remote)
Metadata	A1	61	4	15.25	602	59	10.20		
	A2	32	4	8.00	603	51	11.82		
	B1	43	3	14.33	651	42	15.50		
	B2	45	4	11.25	556	46	12.09		
	C1	63	4	15.75	474	44	10.77		
	C2	32	4	8.00	522	57	9.16		
	D1	61	4	15.25	469	57	8.23		
	D2	33	4	8.25	597	41	14.56		
	D3	61	4	15.2	498	47	10.60		
	D4	32	4	8.00	558	64	8.72		
	D5	71	5	14.20	598	49	12.20		
One/few revisions	A3	89	10	8.90	6998	41	170.68	45793	1116.90
	B3	66	6	11.00	5053	56	90.23	25690	458.75
	B4	107	6	17.83	4880	48	101.67	18700	389.58
	B5	60	5	12.00	4636	59	78.58	17888	303.19
Most revisions	B6	76	6	12.67	10750	45	238.89	95404	2120.09
	B7	69	6	11.50	10821	50	216.42	85265	1705.30
	B8	72	4	18.00	10435	58	179.91	95755	1650.95
	B9	68	5	13.60	10431	62	168.24	88440	1426.45
	B10	79	6	13.17	10489	43	243.93	100883	2346.12
	B11	82	6	13.67	10518	44	239.05	88279	2006.34
Metadata		48.55	4.00	12.14	557.09	50.64	11.00		
One/few revisions		80.5	6.75	11.93	5391.75	51.00	105.72	27017.75	529.76
Most revisions		74.33	5.5	13.52	10574.00	50.33	210.08	92337.67	1834.52
All		62	4.95	12.52	4339.95	50.62	85.74	66209.70*	1308.49*

*Only based on the categories *one/few revisions* and *most revisions*



(a) Unaggregated data from the original study.



(b) Aggregated data from our re-analysis

Figure 1: Task completion times for Boa, compared to Java, for different project sizes.