Errata for "Teseo and the Analysis of Structural Dynamic Graphs" (PVLDB 14(6): 1053 - 1066)

Dean De Leo
CWI
dleo@cwi.nl

Per Fuchs
TUM
per.fuchs@cs.tum.edu

Peter Boncz
CWI
boncz@cwi.nl

PVLDB Reference Format:
Dean De Leo, Per Fuchs, and Peter Boncz. Errata for "Teseo and the Analysis of Structural Dynamic Graphs" (PVLDB 14(6): 1053 - 1066). PVLDB, 14(11): 2271 - 2272, 2021.
doi:10.14778/3476249.3476278

In our paper [4], we experimentally evaluated our work, Teseo, together with five other systems under the LDBC Graphalytics benchmark [6]. We developed and publicly released [2] an ad-hoc driver for the purpose. Since the time the paper was published, a bug [1] in the driver has been found. Due to this bug, we discovered that the completion times for Graphalytics have been incorrectly measured by 1 second or slightly more than their actual values. This issue involves the results of Table 2, Figure 8 and Table 3 reported in our paper [4]. Still, because the bug equally affected all systems evaluated and it is only related to the measurements, most of the comparisons and the general conclusions in the paper still hold.

We also want to take advantage of this errata to further integrate some additional feedback, related to the driver, that we received after the publication of the paper and that we believe it can provide substantial value to the presented results. In detail:

(1) We correct the error [1] in the measurements of our Graphalytics driver [2]. As effect, all the absolute results of Table 3 in [4] become lower of about 1 second.
(2) We replace the output of the experiments from a hash table to a vector. The output of all algorithms in Graphalytics is a score associated to each vertex in the graph. With a hash table, all systems took several seconds to create it in Scale Factor SF 26. In some cases, this cost could be higher than the actual execution of a kernel. By replacing the hash table with a vector, the output is always created in the order of milliseconds for the CSR. This change allows to report a completion time of less than a second for the BFS execution of the CSR with SF 26.
(3) We stabilise the results for the CSR in the CDLP kernel for SF 22 and SF 24 of Table 3 in [4], by repeating the experiments a higher number of times. The previous results were caused by "bad runs", where the algorithm converged more slowly due to non deterministic choices made by the OpenMP scheduler.

We note that the changes 1-2 only involve the driver [3] and equally affect all systems evaluated. We request to amend Table 2, Figure 8 and Table 3 of [4] with the new measurements, attached to this errata. It follows:

- Table 2 shows the speed-up of our CSR implementation over GraphMat and GraphBlas. Compared to the results in paper [4], due to the improvements to our driver, the CSR becomes always faster than the other systems, based on their native external drivers, and the speed-up differences are larger.
- Figure 8 shows the speed-up of the kernel implementations used in our experiments over those shipped by the library authors. Compared to the same Figure 8 in [4], the speed-up differences are more pronounced, particularly for the BFS, as the completion times for the kernels are lower.
- Table 3 shows the speed-up of the CSR implementation over the systems evaluated. Compared to Table 3 in [4], the completion times of the CSR are generally lower. As consequence, the speed-up differences are more pronounced for the SF 22, SF 24 and the BFS kernel. There are only limited differences for SF 26, as the overall completion times were already dominated by the kernel executions. The speed-up of the CSR over GraphOne can now be significant for the faster kernels, due to the time spent by this system to set up a *static view*. The speed-up of the CSR over LiveGraph for the BFS kernel can also be particularly high, due to the preprocessing time required to compute the degrees of the vertices[1]. These overheads were previously hidden in the measurements of [4]. The results for the CDLP kernel for the CSR are always better or comparable to the other systems.

REFERENCES

[1] 2021. Bug fix for the Graphalytics driver. https://github.com/cwida/gfe_driver/commit/7f2b79830ad55a428ae3667d4117f753ed0cb341
[2] 2021. Graphalytics driver public repository. https://github.com/cwida/gfe_driver
[3] 2021. Proposed changes to the Graphalytics driver. https://github.com/cwida/gfe_driver/tree/feature/vector_materialization
[4] Dean De Leo and Peter Boncz. 2021. Teseo and the Analysis of Structural Dynamic Graphs. *Proc. VLDB Endow.* 14, 6 (Feb. 2021), 1053–1066. https://doi.org/10.14778/3447689.3447708
[5] Bálint Hegyi and Gábor Szárnyas. 2020. SuiteSparse:GraphBLAS driver for LDBC Graphalytics. https://github.com/ftsrg/ldbc-graphalytics-platforms-graphblas
[6] Alexandru Iosup, Tim Hegeman, Wing Lung Ngai, Stijn Heldens, Arnau Prat-Pérez, Thomas Manhardto, Hassan Chafio, Mihai Capotă, Narayanan Sundaram, Michael Anderson, Ilie Gabriel Tănase, Yinglong Xia, Lifeng Nai, and Peter Boncz. 2016. LDBC Graphalytics: A Benchmark for Large-scale Graph Analysis on Parallel

[1] In LiveGraph, the API does not provide the degree of a vertex and it needs to be computed explicitly.

Proceedings of the VLDB Endowment, Vol. 14, No. 11 ISSN 2150-8097.
doi:10.14778/3476249.3476278

Graph	System	BFS	CDLP	LCC	LCC (opt)	PageRank	SSSP	WCC	Graph	System	BFS	CDLP	LCC	LCC (opt)	PageRank	SSSP	WCC
DOTA League	CSR (baseline)	< 1*ms*	1.48*s*	472*s*	69*s*	0.16*s*	0.37*s*	0.04*s*	Graph500 SF 22	CSR (baseline)	0.03*s*	4.25*s*	436*s*	18.05*s*	0.43*s*	1.26*s*	0.19*s*
	Stinger	127.21*x*	1.43*x*	3.16*x*	N/A	6.22*x*	3.32*x*	5.78*x*		Stinger	4.86*x*	1.30*x*	2.97*x*	N/A	4.01*x*	1.92*x*	4.26*x*
	LLAMA	**2.31x**	2.67*x*	DNF	N/A	**1.05x**	5.99*x*	1.30*x*		LLAMA	**1.44x**	2.51*x*	DNF	N/A	**1.44x**	4.00*x*	**1.07x**
	GraphOne	4386.22*x*	3.66*x*	1.29*x*	N/A	25.50*x*	11.72*x*	135.48*x*		GraphOne	112.18*x*	2.91*x*	1.34*x*	N/A	11.55*x*	4.91*x*	65.52*x*
	LiveGraph	84.57*x*	1.20*x*	1.43*x*	N/A	3.02*x*	2.09*x*	2.59*x*		LiveGraph	17.85*x*	2.16*x*	1.46*x*	N/A	4.75*x*	2.04*x*	2.51*x*
	Teseo, log. vtx	11.05*x*	1.54*x*	1.60*x*	1.49*x*	2.59*x*	2.23*x*	2.76*x*		Teseo, log. vtx	1.58*x*	1.76*x*	2.20*x*	3.72*x*	4.29*x*	2.98*x*	4.39*x*
	Teseo, real vtx	9.24*x*	**0.99x**	**1.08x**	**1.27x**	1.14*x*	**1.38x**	**0.98x**		Teseo, real vtx	1.87*x*	**1.13x**	**1.18x**	**1.90x**	1.91*x*	**1.66x**	1.56*x*
Uniform SF 24	CSR (baseline)	0.15*s*	34*s*	32*s*	4.88*s*	4.08*s*	8.26*s*	1.45*s*	Graph500 SF 24	CSR (baseline)	0.14*s*	21*s*	3195*s*	105*s*	2.63*s*	6.71*s*	1.04*s*
	Stinger	2.23*x*	1.18*x*	2.36*x*	N/A	2.49*x*	**1.89x**	1.99*x*		Stinger	4.70*x*	1.20*x*	DNF	N/A	3.17*x*	**1.78x**	3.49*x*
	LLAMA	1.63*x*	2.19*x*	DNF	N/A	2.42*x*	8.47*x*	1.51*x*		LLAMA	**1.41x**	1.98*x*	DNF	N/A	**1.68x**	5.25*x*	**1.18x**
	GraphOne	26.81*x*	2.03*x*	3.13*x*	N/A	3.33*x*	3.15*x*	61.32*x*		GraphOne	31.68*x*	2.00*x*	DNF	N/A	4.05*x*	2.79*x*	57.49*x*
	LiveGraph	6.33*x*	1.90*x*	1.65*x*	N/A	3.07*x*	2.99*x*	2.07*x*		LiveGraph	19.40*x*	1.77*x*	DNF	N/A	3.74*x*	1.94*x*	2.07*x*
	Teseo, log. vtx	1.10*x*	1.69*x*	3.18*x*	3.60*x*	3.20*x*	3.46*x*	2.95*x*		Teseo, log. vtx	1.52*x*	1.77*x*	DNF	6.14*x*	3.96*x*	3.23*x*	4.46*x*
	Teseo, real vtx	**0.76x**	**1.05x**	**1.38x**	**1.89x**	**1.53x**	2.11*x*	**1.48x**		Teseo, real vtx	1.68*x*	**1.04x**	DNF	**1.93x**	1.86*x*	1.98*x*	1.59*x*
Uniform SF 26	CSR (baseline)	0.55*s*	170*s*	137*s*	20*s*	20*s*	40*s*	6.87*s*	Graph500 SF 26	CSR (baseline)	0.50*s*	108*s*	DNF	683*s*	15.05*s*	36*s*	5.19*s*
	Stinger	4.24*x*	1.16*x*	2.46*x*	N/A	2.15*x*	**1.81x**	1.82*x*		Stinger	2.55*x*	1.13*x*	DNF	N/A	2.49*x*	**1.63x**	2.91*x*
	GraphOne	12.24*x*	1.68*x*	3.33*x*	N/A	2.30*x*	2.76*x*	59.94*x*		GraphOne	10.26*x*	1.66*x*	DNF	N/A	2.64*x*	2.22*x*	62.01*x*
	LiveGraph	7.19*x*	1.59*x*	1.64*x*	N/A	2.75*x*	3.16*x*	1.95*x*		LiveGraph	21.72*x*	1.53*x*	DNF	N/A	3.14*x*	2.02*x*	1.89*x*
	Teseo, log. vtx	1.30*x*	1.62*x*	3.46*x*	4.22*x*	2.93*x*	3.48*x*	2.01*x*		Teseo, log. vtx	1.34*x*	1.71*x*	DNF	DNF	3.45*x*	2.97*x*	4.24*x*
	Teseo, real vtx	**0.89x**	**1.02x**	**1.39x**	**2.03x**	**1.39x**	2.16*x*	**1.46x**		Teseo, real vtx	**1.03x**	**1.00x**	DNF	**2.08x**	**1.62x**	1.76*x*	**1.49x**

Table 3: Results measured in the Graphalytics benchmark. For the CSR, we report the absolute completion time. For all the other systems, for ease of comparison, we show the speed-up of the CSR over the given system. The best result, outside the CSR, is marked in bold. The results refer to the median out of 15 runs.

System	Graph	BFS	CDLP	LCC	PageRank	SSSP	WCC
GraphMat	graph500-24	94.55*x*	40.15*x*	DNF	49.39*x*	7.23*x*	177.13*x*
	uniform-24	268*x*	74.12*x*	198*x*	87.06*x*	10.74*x*	179.12*x*
SuiteSparse	graph500-24	13.82*x*	3.40*x*	3.04*x*	10.44*x*	2.06*x*	4.15*x*
	uniform-24	7.92*x*	2.19*x*	4.62*x*	10.64*x*	2.87*x*	5.22*x*

Table 2: Speed-up achieved by our CSR baseline over GraphMat [8] and SuiteSparse:GraphBLAS [5]. The results refer to the *processing time* [7], that is, the completion time without the preprocessing and loading time, from the median out of 5 runs.

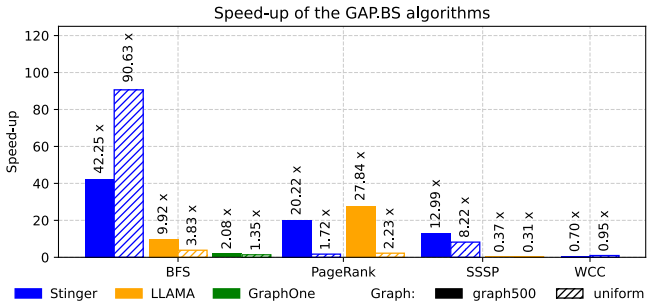

Figure 8: Speed-up of the graph algorithms from the GAP BS over the same algorithms provided by the library authors, for the graph500 and uniform graphs at SF 24.

and Distributed Platforms. *Proc. VLDB Endow.* 9, 13 (Sept. 2016), 1317–1328. https://doi.org/10.14778/3007263.3007270

[7] Alexandru Iosup, Tim Hegeman, Wing Lung Ngai, Stijn Heldens, Arnau Prat-Pérez, Thomas Manhardto, Hassan Chafio, Mihai Capotă, Narayanan Sundaram, Michael Anderson, Ilie Gabriel Tănase, Yinglong Xia, Lifeng Nai, and Peter Boncz. 2017. LDBC Graphalytics Benchmark specification, v0.9.0. http://github.com/ldbc/ldbc_graphalytics_docs

[8] Wing Lung Ngai, Stijn Heldens, Mihai Capotă, Ahmed Musaafir, Tim Hegeman, Narayanan Sundaram, and Michael J. Anderson. 2017. Graphalytics GraphMat platform driver. https://github.com/atlarge-research/graphalytics-platforms-graphmat