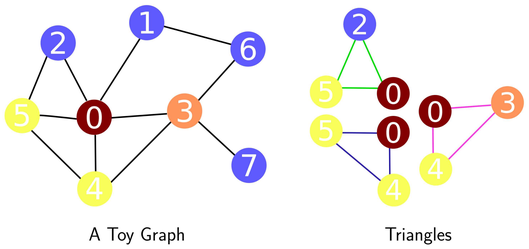
SAND2018-12369C

# Fast Linear Algebra-Based Triangle Analytics with Kokkos Kernels

Abdurrahman Yaşar[1], Sivasankaran Rajamanickam[2], Michael Wolf[2], Jon Berry[2], Ümit V. Çatalyürek[1]

[1]Georgia Institute of Technology, [2]Sandia National Labaratories

## Introduction




A Toy Graph  Triangles

Triangle-based analytics are important in many larger data-analytics based applications. Previously, a highly efficient linear algebra-based algorithm has been developped in Kokkos-Kernels

Primary contributions:

- We improve upon that work by developing an SpGEMM implementation that relies on a highly efficient, work-stealing, multithreaded runtime.
- We demonstrate that our implementation results in improving the runtime up to $5\times$ to $12\times$ on different architectures

## Triangle Counting

A triangle can be defined as a set of three mutually adjacent vertices in a graph.

*Triangle Counting Problem*

Given a graphs $G = \{V, E\}$, the triangle counting problem is to find the number $(T)$ of all set of three vertices, $u, v, w \in V$, such that:

$$T = |\ \{u, v, w \mid (u, v), (v, w), (w, u) \in E\}\ | .$$

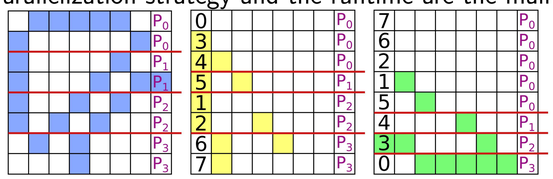
## Linear Algebra Formulations

Two linear-algebra based formulations of triangle counting that are based on the adjacency matrix of the graph: $L$ and $U$ represent lower and upper parts

- LU algorithm; $D = (L \cdot U) . * L$
  - (Pro): Low operation count,
  - (Con): Poor scalability
- LL algorithm; $D = (L \cdot L) . * L$
  - (Pro): Good scalability,
  - (Con): More operations than LU

## kkTri-Cilk Algorithm

Parallelization strategy and the runtime are the main differences between KKTri-Cilk and KKTri.

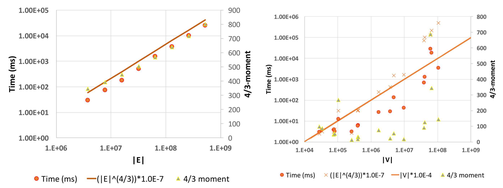



Matrix Form  LL Alg: L Matrix  LU Alg: L Matrix

- Rows are ordered for avoiding computation:
  - $LL$; in decreasing degree,
  - $LU$; in increasing degree
- Balance number of non-zeros within each partition.
- Each partition is spawned (in parallel) as a task.
  - A task runs matrix matrix multiplication within a partition.

## Triangle Counting Scalability

Letting $d_v$ be the degree of vertex $v$, the $4/3$-moment is defined as: $E[d_v^{4/3}] = 1/n \sum_v (d_v^{4/3})$.




Scale Graphs  Non Scale Graphs

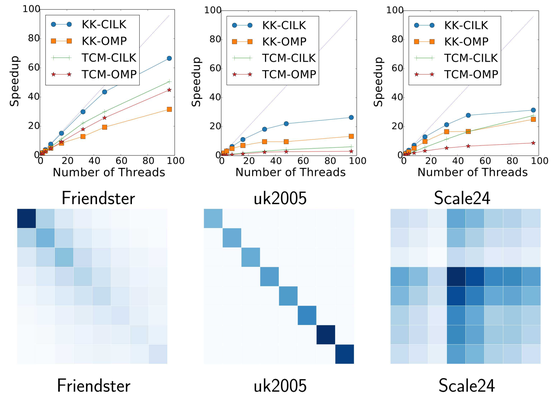
- $m^{4/3}$ (Graph Challenge 2017 regression line) predicts the trends effectively in the Scale graphs.
- However, $m^{4/3}$ does not model the runtime on real graphs as effectively as the $4/3$-moment does.

| Graph Type | Time-per-vertex | | Time-per-edge | |
|---|---|---|---|---|
| | $E^{\frac{4}{3}}$ | $\frac{4}{3}$-moment | $E^{\frac{4}{3}}$ | $\frac{4}{3}$-moment |
| Scale graphs | 0.90 | 0.98 | 0.89 | 0.98 |
| Non-scale graphs | 0.26 | 0.95 | 0.02 | 0.76 |

Per-vertex/edge runtimes are highly correlated with the $4/3$-moment, as predicted

## Experiments: Strong Scaling

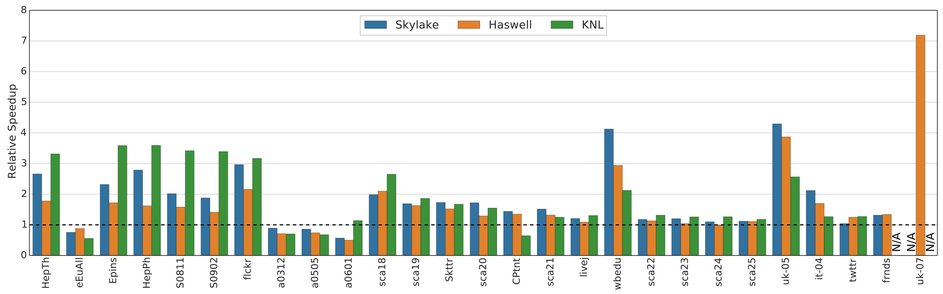
Following figures show strong scaling experiments for OpenMP and Cilk implementations of two algorithms.




- KKTri-Cilk scales the best in all three problems
- uk-2005 achieves best rate: highly local computations.
- scale24 achieves worst rate: Poor cache usage.
- Friendster graph's distribution is in between (best scalability).

## Experiments: Relative Speedup

Comparisons of KKTri-Cilk with TCM, a state-of-the-art graph library.




- KKTri achieves up to $7\times$ speedup on graphs that have a good natural ordering such as wb-edu, uk-2005, and uk-2007. KKTri outperforms TCM in 23 of 27 cases.

## Experiments: Dataset and Peak Rate

Times highlighted in green when KKTri-Cilk is the fastest.

- $10^9$ barrier is passed for the uk-2005 matrix and wb-edu graph.
- A high correlation $(0.91)$ between the conductance and the rate.

| Data Set | $1 - C^d$ | Time (s) | Rates Skylake | Rates Haswell | Rates KNL |
|---|---|---|---|---|---|
| cit-HepTh | 0.141 | 0.003 | 1.20E+08 | 8.24E+07 | 1.54E+07 |
| email-EuAll | 0.112 | 0.003 | 1.16E+08 | 1.10E+08 | 2.16E+07 |
| soc-Epinions1 | 0.086 | 0.004 | 1.06E+08 | 6.72E+07 | 2.44E+07 |
| cit-HepPh | 0.091 | 0.004 | 1.11E+08 | 8.77E+07 | 2.47E+07 |
| soc-Slashdot0811 | 0.067 | 0.004 | 1.18E+08 | 7.97E+07 | 2.71E+07 |
| soc-Slashdot0902 | 0.069 | 0.003 | 1.57E+08 | 8.64E+07 | 2.77E+07 |
| flickrEdges | 0.098 | 0.013 | 1.85E+08 | 1.15E+08 | 2.99E+07 |
| amazon0312 | 0.229 | 0.006 | 3.87E+08 | 2.51E+08 | 9.34E+07 |
| amazon0505 | 0.233 | 0.006 | 3.79E+08 | 2.75E+08 | 9.36E+07 |
| amazon0601 | 0.276 | 0.006 | 4.17E+08 | 2.87E+08 | 9.81E+07 |
| scale18 | 0.059 | 0.031 | 1.24E+08 | 1.07E+08 | 2.88E+07 |
| scale19 | 0.058 | 0.075 | 1.04E+08 | 8.06E+07 | 2.79E+07 |
| as-Skitter | 0.17 | 0.026 | 4.23E+08 | 3.23E+08 | 1.23E+08 |
| scale20 | 0.059 | 0.184 | 8.53E+07 | 5.63E+07 | 2.50E+07 |
| cit-Patents | 0.027 | 0.028 | 5.82E+08 | 4.21E+08 | 1.22E+08 |
| scale21 | 0.059 | 0.511 | 6.21E+07 | 4.78E+07 | 2.01E+07 |
| soc-LiveJournal1 | 0.242 | 0.137 | 3.14E+08 | 2.28E+08 | 1.07E+08 |
| wb-edu | 0.938 | 0.042 | 1.10E+09 | 6.55E+08 | 1.48E+08 |
| scale22 | 0.058 | 1.581 | 4.05E+07 | 3.50E+07 | 1.71E+07 |
| scale23 | 0.059 | 3.786 | 3.41E+07 | 2.62E+07 | 1.45E+07 |
| scale24 | 0.059 | 10.282 | 2.53E+07 | 2.04E+07 | 1.21E+07 |
| scale25 | 0.059 | 25.652 | 2.04E+07 | 1.88E+07 | 9.11E+06 |
| uk-2005 | 0.925 | 0.684 | 1.14E+09 | 9.35E+08 | 2.59E+08 |
| it-2004 | 0.942 | 1.293 | 7.95E+08 | 5.86E+08 | 1.47E+08 |
| twitter | 0.126 | 28.359 | 4.24E+07 | 4.46E+07 | N/A |
| friendster | 0.182 | 18.552 | 9.74E+07 | 7.93E+07 | N/A |
| uk-2007 | 0.968 | 3.545 | 9.31E+08 | 7.49E+08 | N/A |

## Conclusion

- KKTri-Cilk surpasses $10^9$ for the rate measure.
- KKTri-Cilk is faster on 63 of 78 instances
- KKTri-Cilk is faster than state-of-the-art graph based implementation (up to $7\times$)
- We corroborate that the scalability of the triangle counting is bounded by $O(n)$ when the $4/3$-moment is bounded
- We show correlation between the high rates achieved and the conductance of the graph

## References

[1] J. W. Berry, L. K. Fostvedt, D. J. Nordman, C. A. Phillips, C. Seshadhri, and A. G. Wilson, "Why do simple algorithms for triangle enumeration work in the real world?" in *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science*, ser. ITCS '14. New York, NY, USA: ACM, 2014, pp. 225–234. [Online]. Available: http://doi.acm.org/10.1145/2554797.2554819

[2] J. Shun and K. Tangwongsan, "Multicore triangle computations without tuning," in *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE, 2015, pp. 149–160.

[3] M. M. Wolf, M. Deveci, J. W. Berry, S. D. Hammond, and S. Rajamanickam, "Fast linear algebra-based triangle counting with kokkoskernels," in *High Performance Extreme Computing Conference (HPEC), 2017 IEEE*. IEEE, 2017, pp. 1–7.