**SCIENCE CHINA**
Information Sciences

• **Supplementary File** •

# Demystifying Graph Processing Frameworks and Benchmarks

Junyong Deng[1,2*], Qinzhe Wu[2], Xiaoyan Wu[3], Shuang Song[2], Joseph Dean[2] & Lizy Kurian John[2]

[1]*School of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an* 710121*, China;*
[2]*Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin* 78712*, USA;*
[3]*School of Software Engineering, Xi'an Jiaotong University, Xi'an* 710049*, China*

## Appendix A Methodology for Performance Characterization

This appendix describes the methodology and background for performance characterization of the selected graph frameworks/benchmark suites.

### Appendix A.1 Hardware Platform

This study is conducted on the servers of Texas Advanced Computing Center (TACC) equipped with Intel Xeon Platinum 8160 (Skylake) processors. The Skylake processor consists of 2 sockets (24 cores per socket). Each core has a 32KB L1 data cache, a 32KB instruction cache, a 1MB L2 cache, and all cores on a socket share a 33MB L3 cache. The Skylake processor can decode and retire up to 5 instructions per cycle, giving a theoretical maximum IPC of 5. Table S1 summarizes the features of the experimental platform.

**Table S1** The micro-architectural parameters of the Skylake processor and experimental system information

| Index | Description |
|---|---|
| Frequency | 2.1GHz nominal<br>(1.4-3.7GHz depending on instruction set and number of active cores) |
| Cores Per Socket | 24 cores (×2 sockets) |
| L1-icache | 32KB, 64B/line, 8-way, per core |
| L1-dcache | 32KB, 64B/line, 8-way, per core |
| L2 cache | 1MB, 64B/line, 16-way, per core |
| LLC | 33MB, 64B/line, 11-way, per socket |
| RAM | 192GB (2.67GHz) DDR4 |
| Kernel | 3.10.0-693.11.6.el7.x86_64 |
| Perf | 3.10.0-693.11.6.el7.x86_64.debug |

### Appendix A.2 Graph Datasets

The graph inputs for our analysis are summarized in Table S2, where *diameter* represents the longest shortest path between any two vertices in the graph, and *degree* denotes the ratio between number of edges and number of vertices. The seven graphs we select are real world graphs from The SuiteSparse Matrix Collection and Stanford Large Network Dataset Collection, and synthetic random graph, including the two broad categories of graphs: meshes and social networks. Flickr (FLI) is a graph built by forming links between images which shares common metadata. LiveJournal (LJ) and Orkut (OK) are free online social networks with small diameters. RoadCA (CA) for California and RoadTX (TX) for Texas are road networks, where intersections and endpoints are indicated by nodes and undirected edges represent the roads connecting these intersections or road endpoints. Cit-Patents(CIT) is a US patent citation network includes all citations granted between 1975 and 1999. Random(SYN) is a synthetic dataset generated by Graph500 Kronocker Graph Generator which

* Corresponding author (email: djy@xupt.edu.cn)

possesses a power-law degree distribution. It may be noted that FLI, LJ, OK, CIT and SYN are power-law distribution graphs and CA and TX are mesh topology graphs. Compared those graphs used in distributed graph processing systems studies, the graphs we chose are relatively small but they are big enough to stress a single node (the platform shown in Table S1).

**Table S2** Real world graph datasets used in this study

| dataset | #nodes | #edges | diameter | degree | type |
|---|---|---|---|---|---|
| Flickr(FLI) | 0.83M | 9.84M | 18 | 11.87 | Directed |
| LiveJournal(LJ) | 4.85M | 68.99M | 16 | 14.23 | Directed |
| Orkut(OK) | 3.07M | 117.19M | 9 | 76.28 | Undirected |
| RoadCA(CA) | 1.97M | 2.76M | 849 | 2.82 | Undirected |
| RoadTX(TX) | 1.38M | 1.92M | 1054 | 2.79 | Undirected |
| Cit-Patents(CIT) | 3.78M | 16.52M | 22 | 4.38 | Directed |
| Random(SYN) | 8.39M | 134.22M | 19 | 16 | Directed |

## Appendix A.3 Experimental Setup

Table S3 summarizes the compiler configurations of each of the graph platforms. The compiler configurations come together with the source code or are suggested by the documents of the graph frameworks/benchmarks. As those compiler flags suggest, all the frameworks/benchmarks enable OpenMP support, so we set the proper environmental variables (e.g., *OMP_NUM_THREADS*, *KMP_AFFINITY*, *GOMP_CPU_AFFINITY*), in order to reduce the effects from cross-socket communication and thread migration as much as possible. For example, in the *perf* experiments to collect the hardware metrics, we evenly spread 48 threads to 48 physical cores on two sockets. For the scalability study, we run each graph application from the four selected graph platforms with thread numbers of 1, 4, 8, 24, and 48 (later referred as 1-core setup, 4-core setup, 8-core setup, 24-core setup, and 48-core/thread setup, respectively) on seven graphs. In order to evaluate the cross-socket communication overhead, we conduct the experiments in two types of setup for 4-core, 8-core, 24-core, and 48-core/thread. One type is without the cross-socket communication overhead by assigning physical cores in one socket (for 48-thread setup, we spread 48 threads to 24 physical cores in one socket with 2 threads per core), one is with the overhead by evenly distributing the threads to physical cores cross two sockets (those cores having even virtual core id on socket0 and odd id on socket1, while for the 1 thread scenario, it is assigned to the physical core 1 from socket 1).

**Table S3** Compilation configurations for various graph frameworks/benchmarks

| Graph Platform | Compilation Configuration |
|---|---|
| GraphMat | mpiicpc -cxx=icpc -qopenmp -std=c++11 -L/opt/apps/intel18/boost/1.65/lib/ -I./include -I./include/GMDP -I/opt/apps/intel18/boost/1.65/include -O3 -ipo -xHost |
| Graph500 | gcc -flto -fwhole-program -g -std=c99 -Wall -O3 -march=native -lm -lrt -fopenmp |
| GAP | g++ -std=c++11 -O3 -Wall -fopenmp |
| GraphBIG | g++ -std=c++0x -Wall -Wno-deprecated -O3 -DUSE_OMP -fopenmp -lpfm_cxx -lpfm |

## Appendix A.4 Performance Metrics

This subsection describes the metrics used for performance comparisons, including *Data Movement Per Edge*, *Instructions Per Cycle (IPC)*, *L1 data cache MPKI*, *L2 MPKI*, *L3 MPKI*, *Computation Per Edge*, *Execution Time*, and *Energy Consumption Per Edge*.

• **Data Movement Per Edge:** It has been observed that the cost of moving data is higher than the cost of computing operations. In the big data era, many applications are increasingly affected by the cost of data movement. In graph computing, this issue is a prominent problem. To show the intensity of the data movement issue, we measure the number of *Data Movement Per Edge* in each graph. Note that this is the count of the move operations rather the amount of data moved. As shown in Equation A1, we record the counts of load and store instructions (denoted as $\#load$ and $\#store$, respectively), then divide their sum with the number of edges (denoted as $\#edge$).

$$Data\ Movement\ Per\ Edge = \frac{\#load + \#store}{\#edge} \tag{A1}$$

• **IPC:** *IPC* shows the average number of retired instructions per cycle. It is a direct index of Instruction Level Parallelism (ILP). The Skylake architecture issues up to five instructions per cycle, however,in practice, *IPC* rarely reaches the upper bound of 5 due to various effects, such as long-latency communication with memory; floating point or Single Instruction Multiple Data (SIMD) operations; instruction starvation in the front-end etc. *IPC* is an excellent metric for judging an overall potential for application performance tuning, and micro-architecture efficiency. For multi-threaded executions, the *IPCs* are calculated based on the aggregated cycles and instructions of all threads.

• **MPKI:** *MPKI* indicates the average number of misses per kilo-instructions. *MPKI* is usually preferred over cache miss rates, because *MPKI* also conveys information on the fraction of memory access instructions in the overall instruction stream.

• **Computation Per Edge:** We use the number of *Computation Per Edge* to indicate the amount of computations needed to finish the processing of a graph. The *Computation Per Edge* is calculated by Equation A2, where $\#inst$ denotes the total number of instructions of the current execution and $\#br$ denotes the total number of branch instructions.

$$Computation\ Per\ Edge = \frac{\#inst - \#load - \#store - \#br}{\#edge} \tag{A2}$$

• **Execution Time Per Edge:** Execution time indicates the graph processing performance directly, which is recorded using the C++ system library. For example, in each application of GraphMat, it records the start time and end time of graph processing through system function *gettimeofday()*.

• **Energy Consumption Per Edge:** Energy efficiency is an essential consideration in graph analytics, especially for the mobile equipment. Since energy consumption is supposed to scale with the graph size, we use *Energy Consumption Per Edge* instead to indicate the energy efficiency, as shown in Equation A3, where *energy_pkg* indicates the energy consumption (in joules) during the execution duration.

$$Energy\ Consumption\ Per\ Edge = \frac{energy\_pkg}{\#edge} \tag{A3}$$

## Appendix A.5 Principal Components Analysis

As described in previous sections, various metrics are collected on four BFSs, three SSSPs, three PRs, and three TCs for five real-world graph inputs. Many metrics may be correlated to each other. In order to remove the correlation and make similarity analysis more meaningful, Principal Components Analysis (PCA) is leveraged to analyze the raw data. PCA is a statistical data analysis technique. It computes new variables called *principal components* that are linear combinations of the original variables, such that all principal components are uncorrelated. PCA transforms the $p$ variables $X_1$, $X_2$, ..., $X_p$ into $p$ principal components $Z_1$, $Z_2$, ..., $Z_p$ with Equation A4:

$$Z_i = \sum_{j=1}^{p} a_{ij} X_j \tag{A4}$$

This transformation has the properties: (1)$Var[Z_1] \geqslant Var[Z_2] \geqslant ... \geqslant Var[Z_p]$, which means that $Z_1$ contains the more information and $Z_p$ contains the least; and (2) $Cov[Z_i, Z_j]$=0, $\forall\ i \neq j$, which means that there is no information overlap between the principal components. With the first property, we can remove the components with the lowest variance from the analysis as long as the $q$ principal components hold 90% of the total variance. PCA not only reduces the dimensionality of the data set but also controls the data that is lost.

## Appendix B Data Movement Variation

Figure S1 summarizes the metric, *data movement per edge*, for all the experimental cases (combinations of four applications, seven graphs, and four frameworks). Due to the large difference in the order of the magnitude, the vertical axis is in logarithmetic scale.

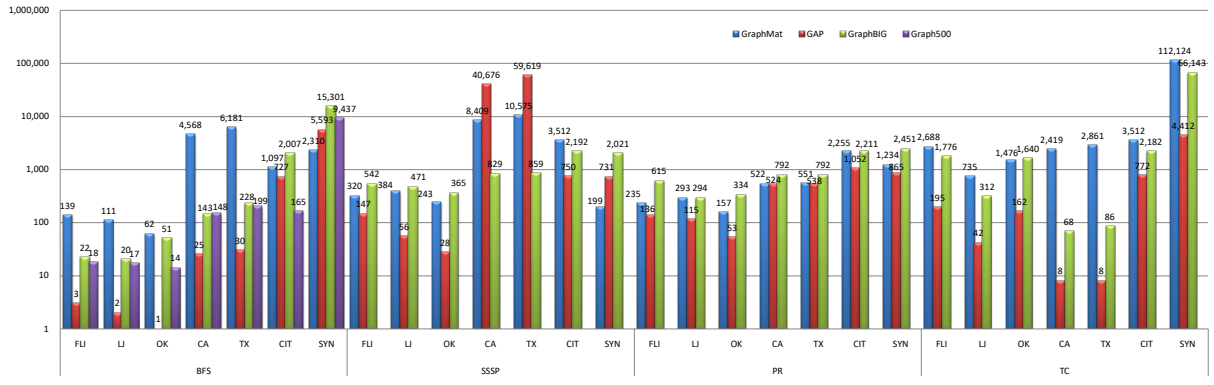



**Figure S1** *Data Movement Per Edge* of graph applications on different graphs. Data are collected on 48-core setup, while BFS is the only application in Graph500 that has multi-threading implementation.

## Appendix C Tables of Collected Data

**Table S4** Data Movement Per Edge of Graph Applications on Different Graphs.

| | BFS | | | | SSSP | | | PageRank | | | Triangle Counting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Graphs | GraphMat | Graph500 | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG |
| FLI | 139.15 | 18.18 | 2.73 | 21.57 | 319.75 | 147.32 | 542.19 | 235.32 | 135.96 | 615.17 | 2,688.39 | 195.37 | 1,775.98 |
| LJ | 110.98 | 16.65 | 2.41 | 20.15 | 383.62 | 55.97 | 471.44 | 293.21 | 114.54 | 293.63 | 735.40 | 41.92 | 312.39 |
| OK | 61.80 | 13.55 | 0.73 | 51.15 | 243.32 | 27.83 | 364.92 | 156.93 | 52.58 | 333.92 | 1,476.12 | 161.67 | 1,640.08 |
| CA | 4,567.69 | 147.60 | 24.83 | 142.78 | 8,409.26 | 40,675.98 | 828.99 | 521.50 | 524.09 | 791.52 | 2,418.86 | 8.24 | 68.18 |
| TX | 6,181.48 | 199.32 | 29.90 | 228.13 | 10,574.72 | 59,618.98 | 858.64 | 551.17 | 537.54 | 792.22 | 2,861.46 | 8.06 | 86.00 |
| CIT | 1097.14 | 165.04 | 727.53 | 2006.87 | 3512.00 | 749.92 | 2191.74 | 2255.09 | 1052.02 | 2210.72 | 3512.00 | 771.96 | 2182.31 |
| SYN | 2309.89 | 9436.85 | 5593.27 | 15301.04 | 198.52 | 731.10 | 2020.88 | 1234.22 | 864.70 | 2450.67 | 112124.40 | 4412.83 | 66143.71 |

**Table S5** IPC of Graph Applications on Different Graphs.

| | BFS | | | | SSSP | | | PageRank | | | Triangle Counting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Graphs | GraphMat | Graph500 | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG |
| FLI | 0.44 | 0.58 | 0.31 | 0.04 | 0.86 | 0.02 | 0.08 | 0.68 | 0.36 | 0.05 | 0.96 | 0.99 | 0.67 |
| LJ | 0.57 | 0.54 | 0.25 | 0.04 | 1.26 | 0.03 | 0.09 | 1.24 | 0.35 | 0.07 | 0.73 | 0.82 | 0.30 |
| OK | 0.6 | 0.71 | 0.2 | 0.08 | 1.54 | 0.03 | 0.09 | 1.31 | 0.43 | 0.05 | 1.08 | 0.62 | 0.75 |
| CA | 1.38 | 0.33 | 0.05 | 0.04 | 0.88 | 0.03 | 0.07 | 1.10 | 0.23 | 0.07 | 0.39 | 0.40 | 0.02 |
| TX | 1.3 | 0.27 | 0.05 | 0.05 | 0.96 | 0.03 | 0.07 | 0.89 | 0.28 | 0.07 | 0.37 | 0.38 | 0.03 |
| CIT | 0.32 | 0.28 | 0.50 | 0.68 | 0.43 | 0.53 | 0.72 | 0.63 | 0.40 | 0.32 | 0.43 | 0.45 | 0.26 |
| SYN | 0.50 | 0.51 | 0.93 | 0.45 | 1.22 | 0.79 | 0.49 | 0.72 | 0.70 | 0.24 | 1.43 | 1.20 | 1.73 |

**Table S6** L1D MPKI of Graph Applications on Different Graphs.

| | BFS | | | | SSSP | | | PageRank | | | Triangle Counting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Graphs | GraphMat | Graph500 | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG |
| FLI | 10.87 | 33.83 | 15.19 | 116.23 | 21.46 | 19.55 | 147.87 | 18.99 | 104.32 | 91.08 | 3.37 | 5.58 | 10.43 |
| LJ | 12.77 | 29.91 | 35.46 | 132.45 | 22.52 | 24.49 | 130.04 | 23.22 | 94.80 | 97.43 | 8.15 | 8.69 | 16.20 |
| OK | 13.89 | 34.95 | 30.14 | 69.61 | 19.06 | 27.39 | 146.32 | 18.79 | 141.46 | 126.12 | 0.13 | 9.34 | 13.27 |
| CA | 18.44 | 55.63 | 51.56 | 53.76 | 24.22 | 15.16 | 122.03 | 17.82 | 31.09 | 59.13 | 5.14 | 18.69 | 57.69 |
| TX | 18.65 | 79.22 | 47.56 | 36.34 | 24.84 | 19.27 | 126.74 | 16.66 | 32.59 | 62.51 | 4.54 | 24.08 | 45.20 |
| CIT | 2.99 | 18.23 | 5.32 | 9.25 | 0.21 | 5.57 | 8.67 | 15.57 | 31.55 | 18.42 | 9.07 | 12.50 | 12.35 |
| SYN | 9.03 | 36.65 | 3.86 | 18.40 | 10.60 | 4.46 | 16.90 | 19.69 | 38.64 | 33.05 | 5.08 | 5.56 | 14.96 |

**Table S7** L2 MPKI of Graph Applications on Different Graphs.

| | BFS | | | | SSSP | | | PageRank | | | Triangle Counting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Graphs | GraphMat | Graph500 | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG |
| FLI | 0.84 | 6.49 | 1.91 | 47.08 | 1.76 | 8.42 | 72.79 | 1.25 | 14.58 | 47.86 | 0.14 | 0.48 | 0.92 |
| LJ | 0.98 | 12.67 | 6.43 | 50.25 | 1.80 | 9.95 | 49.49 | 1.09 | 56.15 | 45.27 | 0.87 | 1.72 | 3.01 |
| OK | 0.82 | 20.24 | 7.64 | 23.16 | 1.24 | 11.62 | 55.41 | 0.61 | 69.83 | 61.39 | 0.36 | 2.34 | 1.16 |
| CA | 1.07 | 1.24 | 7.26 | 21.38 | 3.42 | 7.10 | 53.86 | 1.01 | 8.94 | 39.32 | 0.63 | 6.36 | 14.95 |
| TX | 1.05 | 1.25 | 7.30 | 13.44 | 3.38 | 7.35 | 54.53 | 0.89 | 8.36 | 42.24 | 0.56 | 6.65 | 18.07 |
| CIT | 0.29 | 0.44 | 0.15 | 2.54 | 1.29 | 0.11 | 2.34 | 0.86 | 17.06 | 6.61 | 1.29 | 2.64 | 3.53 |
| SYN | 0.17 | 23.14 | 0.15 | 5.17 | 0.23 | 0.16 | 4.73 | 0.69 | 22.79 | 12.88 | 0.03 | 0.49 | 0.31 |

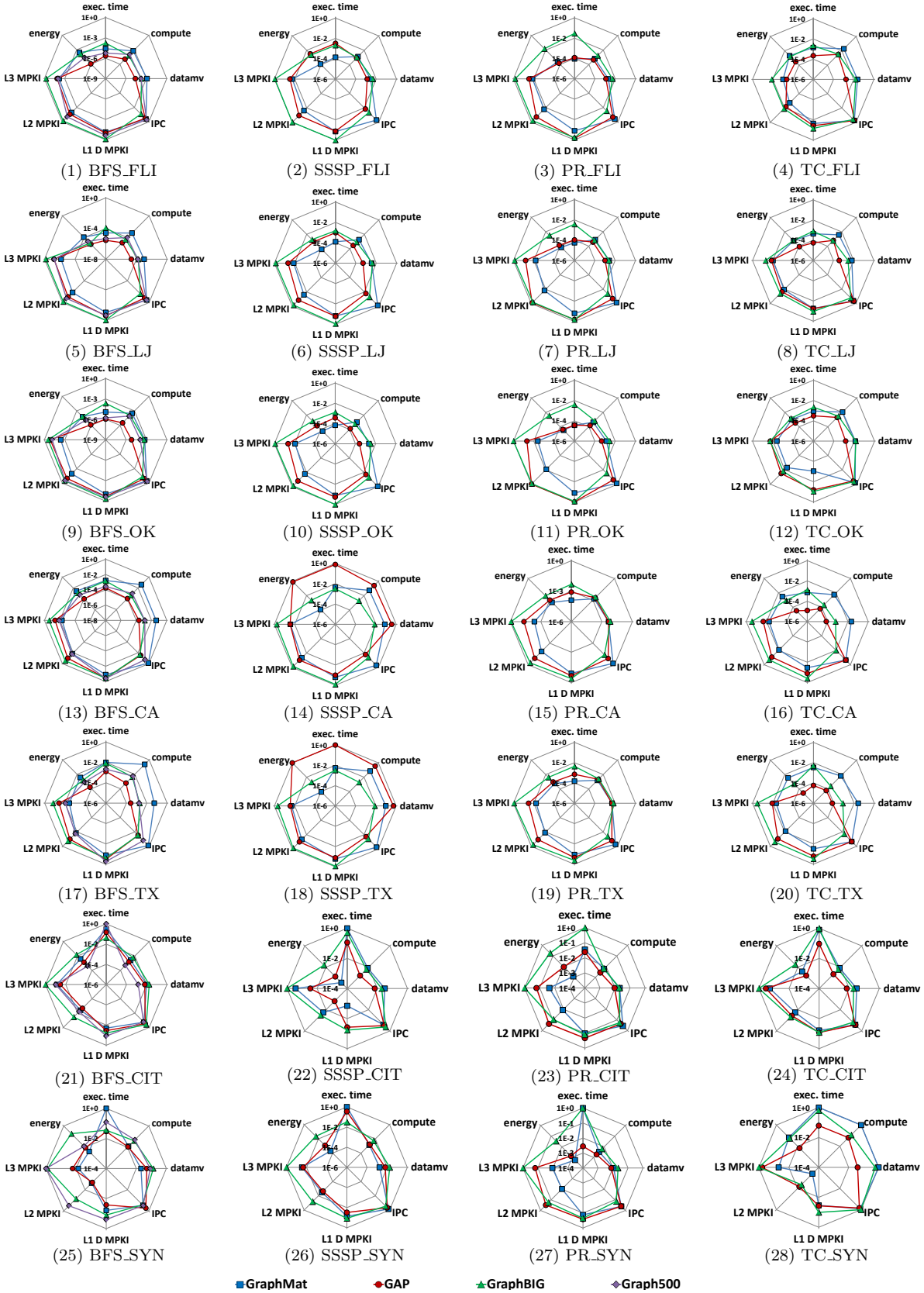
**Table S8** Computation Per Edge of Graph Applications on Different Graphs.

| | BFS | | | | SSSP | | | PageRank | | | Triangle Counting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Graphs | GraphMat | Graph500 | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG |
| FLI | 222.54 | 40.03 | 4.40 | 25.55 | 512.40 | 331.22 | 391.41 | 249.40 | 170.64 | 658.27 | 7,404.83 | 1,217.74 | 1,274.86 |
| LJ | 260.12 | 37.56 | 4.26 | 16.81 | 696.50 | 110.92 | 323.73 | 305.13 | 140.60 | 254.30 | 1,486.62 | 249.40 | 232.60 |
| OK | 120.43 | 31.80 | 1.26 | 45.69 | 415.63 | 49.28 | 230.36 | 160.71 | 57.95 | 259.47 | 4,212.12 | 755.10 | 1,126.68 |
| CA | 15768.00 | 346.97 | 42.00 | 184.40 | 21,163.55 | 98,523.89 | 740.63 | 580.80 | 809.61 | 967.69 | 2,134.86 | 23.83 | 78.66 |
| TX | 20701.32 | 475.85 | 51.80 | 388.38 | 27,873.43 | 138,704.65 | 771.28 | 606.82 | 830.77 | 952.81 | 2,516.21 | 23.20 | 94.42 |
| CIT | 998.99 | 187.39 | 572.56 | 2238.51 | 3231.17 | 624.77 | 2443.06 | 2352.04 | 1078.85 | 2450.20 | 3231.17 | 850.40 | 2423.74 |
| SYN | 4836.35 | 19191.86 | 4254.16 | 16091.23 | 551.89 | 644.60 | 2155.60 | 1281.25 | 736.73 | 2447.85 | 386240.89 | 24687.84 | 45269.04 |

**Table S9** Energy Consumption Per Edge of Graph Applications on Different Graphs (in Joules).

| | BFS | | | | SSSP | | | PageRank | | | Triangle Counting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Graphs | GraphMat | Graph500 | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG | GraphMat | GAP | GraphBIG |
| FLI | 3.58E-06 | 2.96E-07 | 1.52E-08 | 1.74E-06 | 1.60E-06 | 4.18E-05 | 2.97E-05 | 2.96E-06 | 1.92E-06 | 2.01E-04 | 2.64E-05 | 3.80E-06 | 1.98E-05 |
| LJ | 1.52E-06 | 2.54E-07 | 7.52E-08 | 1.11E-07 | 1.11E-06 | 1.28E-05 | 2.47E-05 | 8.96E-07 | 1.72E-06 | 4.12E-05 | 7.44E-06 | 1.14E-06 | 8.09E-06 |
| OK | 8.04E-07 | 1.62E-07 | 1.64E-08 | 7.58E-07 | 7.43E-07 | 4.38E-06 | 1.93E-05 | 4.24E-07 | 5.41E-07 | 4.56E-05 | 1.37E-05 | 4.44E-06 | 1.89E-05 |
| CA | 3.44E-05 | 7.26E-06 | 1.25E-06 | 1.59E-05 | 1.53E-06 | 1.07E-02 | 2.92E-05 | 6.43E-06 | 1.32E-05 | 5.36E-05 | 3.86E-05 | 4.17E-07 | 1.15E-05 |
| TX | 4.73E-05 | 1.13E-05 | 2.15E-06 | 1.96E-05 | 1.18E-06 | 1.31E-02 | 2.68E-05 | 7.46E-06 | 1.21E-05 | 5.37E-05 | 4.54E-05 | 3.50E-07 | 6.36E-06 |
| CIT | 5.07E-05 | 4.91E-06 | 1.68E-05 | 1.96E-04 | 4.66E-06 | 1.73E-05 | 2.03E-04 | 1.62E-05 | 1.23E-04 | 2.44E-03 | 5.13E-05 | 2.12E-05 | 2.33E-04 |
| SYN | 5.21E-05 | 1.62E-04 | 1.16E-04 | 2.50E-03 | 2.61E-06 | 1.52E-05 | 3.00E-04 | 7.35E-06 | 1.77E-05 | 4.52E-04 | 8.64E-04 | 8.60E-05 | 7.28E-04 |

## Appendix D Kiviat Diagrams




(1) BFS_FLI (2) SSSP_FLI (3) PR_FLI (4) TC_FLI

(5) BFS_LJ (6) SSSP_LJ (7) PR_LJ (8) TC_LJ

(9) BFS_OK (10) SSSP_OK (11) PR_OK (12) TC_OK

(13) BFS_CA (14) SSSP_CA (15) PR_CA (16) TC_CA

(17) BFS_TX (18) SSSP_TX (19) PR_TX (20) TC_TX

(21) BFS_CIT (22) SSSP_CIT (23) PR_CIT (24) TC_CIT

(25) BFS_SYN (26) SSSP_SYN (27) PR_SYN (28) TC_SYN

**Figure S2** Kiviat diagrams of graph applications with different graphs. Data are collected on 48-core setup. BFS is the only application that has OpenMP implementation in Graph500.

## Appendix E Scalability

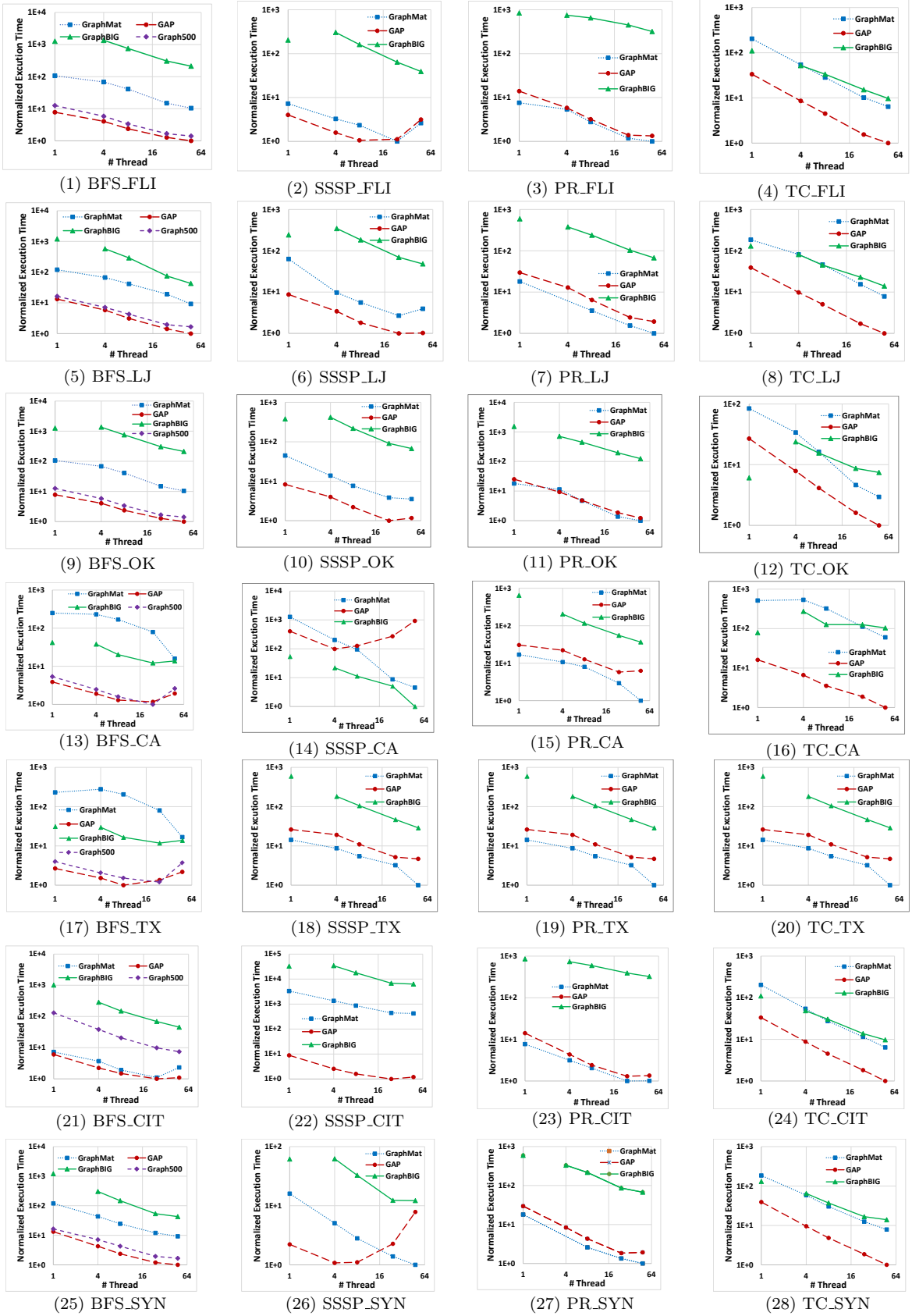



**Figure S3** Performance comparison and scalability of graph applications with different graphs. BFS is the only application that has OpenMP implementation in Graph500. Normalized with respect to lowest execution time in each chart.

## Appendix F Correlation Analysis

Based on the abundant experimental data collected (91 data points per metric), we also perform a thorough correlation analysis, and the results are reported in Table S10. The range for possible correlation coefficients is from -1 to 1, where 1 means two metrics are "perfectly correlated" on every data point (self-correlation for example), while -1 indicates an opposite trend. The top half of Table S10 presents correlation against counts of various events (labeled value), and the bottom half presents correlation against events per edge or per instruction (labeled ratio). As we can see, in the value section, the total number of L1 Data cache misses, data movement, and number of computations show the strongest correlation. In the ratio section, *Data Movement Per Edge* and *Compuations Per Edge* demonstrates a nearly perfect correlation to both performance and energy. *CPI* is correlated to performance or energy to some extent, however, all the *MPKIs* seem to be nearly useless in indicating either performance or energy (correlation coefficients are around zeros). Actually, cache techniques have progressed so much in the past several decades that caches perform very well for almost all of the graph applications, as we observe low cache *MPKIs* in many cases (in about two-thirds of cases *L1D MPKI*$\leqslant$40, *L2 MPKI*$\leqslant$10, *L3 MPKI*$\leqslant$2). The analysis also imply that graph analytics applications would benefit from hardware techniques which target on reducing data movement, for example, computing in situ and in memory.

**Table S10** Correlation of Performance and Energy to Various Metrics

| | Metrics | Correlation to Performance | Correlation to Energy |
|---|---|---|---|
| Value (total count) | Wall clock time | 1.000 | 0.166 |
| | **# L1D Miss** | **0.674** | **0.659** |
| | # L2 Miss | 0.614 | 0.609 |
| | # L3 Miss | 0.256 | 0.247 |
| | # Branch Prediction Miss | 0.125 | 0.107 |
| | **Data movement** | 0.548 | **0.834** |
| | **Computation** | 0.578 | **0.708** |
| | Energy consumption | 0.998 | 1.000 |
| Ratio (per edge/instruction) | Execution Time Per Edge | 1.000 | 0.977 |
| | CPI | 0.333 | 0.332 |
| | L1D MPKI | -0.103 | -0.106 |
| | L2 MPKI | -0.066 | -0.069 |
| | L3 MPKI | -0.082 | -0.085 |
| | Branch Prediction MPKI | -0.127 | -0.127 |
| | Speculation Ratio | -0.128 | -0.128 |
| | **Data Movement Per Edge** | **0.975** | **0.974** |
| | Computation Per Edge | 0.968 | 0.967 |
| | Energy Consumption Per Edge | 0.977 | 1.000 |

## Appendix G Benchmark Options

Based on the analysis done in the previous three sections, this section explores the workload space and offers some suggestions on choosing graph workloads as benchmarks.

We utilize Principal Component Analysis (PCA) to reduce the dimensions, and Figure S4 presents the distribution of the workloads in a 2D space of the two most dominant principal components covering more than 80% of the variance. Comparing the frameworks (indicated in different colors), GraphMat is the one that has it workloads most closely clustered together around the left bottom corner. GAP has two workloads (i.e., SSSP on CA and TX) extending the space on the dimension of PC2 dramatically. In contrast, GraphBIG has a very small span on PC2, but spreads its data points along PC1 from -0.38 (TC on OK) to 1.28 (SSSP on FLI). If categorized application by application (distinguished by marker types), SSSP is the one worth attention, because SSSP has the most diversity and two SSSP implementations are the outliers of the space. On the other hand, the workloads of TC gather more tightly than the workloads of other three applications.

Table S11 exhibits lists of graph workloads for several architectural evaluations. New cache designs may show their improvement on GraphBIG SSSP, since there are high cache *MPKIs*; The matrix structure and the low diversity across applications and graphs make GraphMat a good candidate for accelerators; Architects could justify the idea of PIM with GAP SSSP on some mesh graphs, because those are the test cases having a lot of data movement; Similarly, GAP SSSP on the mesh graphs could be used as the litmus test for multi-core processor designs where communication overhead is a big concern; To evaluate heterogeneous memory systems, diverse data access patterns are necessary, while OK and TX can be selected as the representatives for power-law graphs and mesh graphs, both of which should be covered.

As a suggestion in general, architects or computing system designers should pay attention to different implementations, trying several graph applications on both power-law graphs (social network for instance) and mesh graphs (such as road networks).

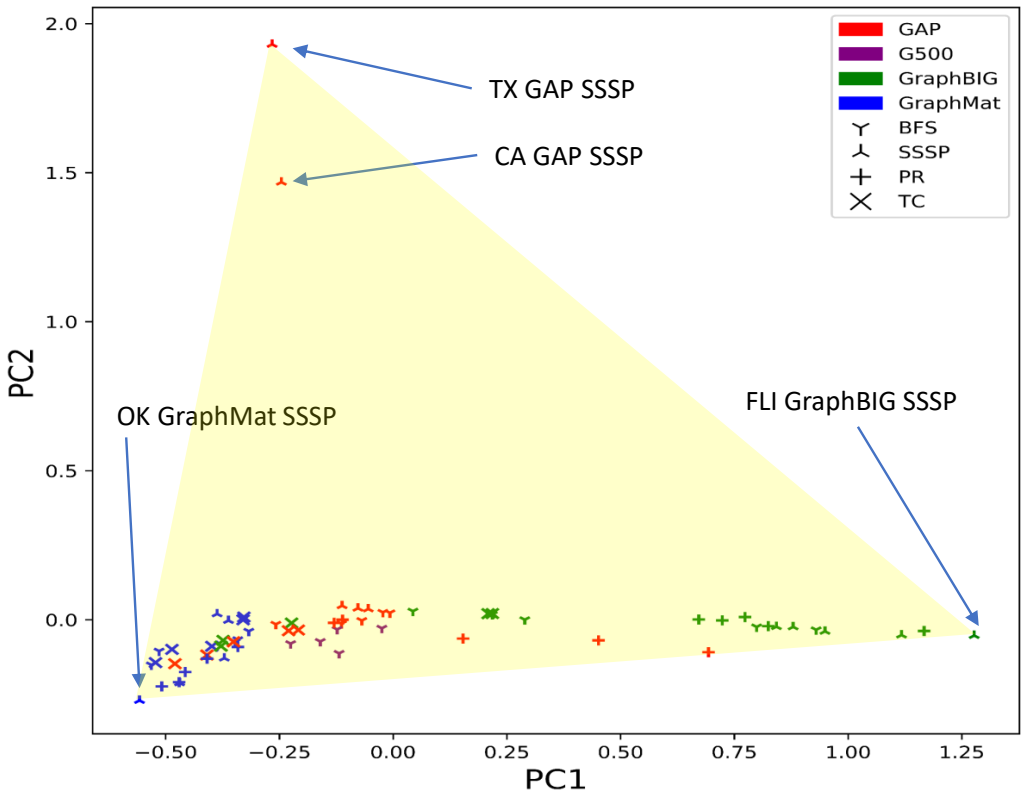



**Figure S4** Graph workloads in the Principal Component space. Principal Component 1 (PC1) together with Principal Component 2 (PC2) achieve fairly high coverage of the information from the eight aforementioned metrics (i.e., *Data Movement Per Edge*, *Instructions Per Cycle (IPC)*, *L1 data cache MPKI*, *L2 MPKI*, *L3 MPKI*, *Computation Per Edge*, *Execution Time Per Edge*, and *Energy Consumption Per Edge*). PC1 is dominated by *MPKIs* of three levels of cache (the higher the value is, the higher cache MPKIs are), while PC2 groups mainly the *Data Movement Per Edge*, *Computation Per Edge*, *Execution Time Per Edge*, and *Energy Consumption Per Edge* (the value increases as the amount of data movement, computation increase).

**Table S11** Suggested Graph Workloads with the Desired Features

| Purpose | Feature | Graph Workloads |
|---|---|---|
| Cache Evaluation | Challenging Cache | GraphBIG SSSP |
| Accelerator Design | Regularity and Scalability | GraphMat |
| Processing in Memory | Plenty of data movement | GAP SSSP on CA/TX |
| Multicore Processor or Network-on-Chip | Sensitivity to On-chip Communication | GAP SSSP on CA/TX |
| Heterogeneous Memory System | Diverse Data Access Pattern | Power-law and mesh graphs |