# A Study of Partitioning Policies for Graph Analytics on Large-scale Distributed Platforms

Gurbinder Gill Roshan Dathathri Loc Hoang Keshav Pingali

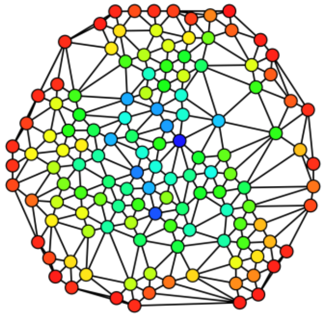
汇报人：陶明洄

51205901082

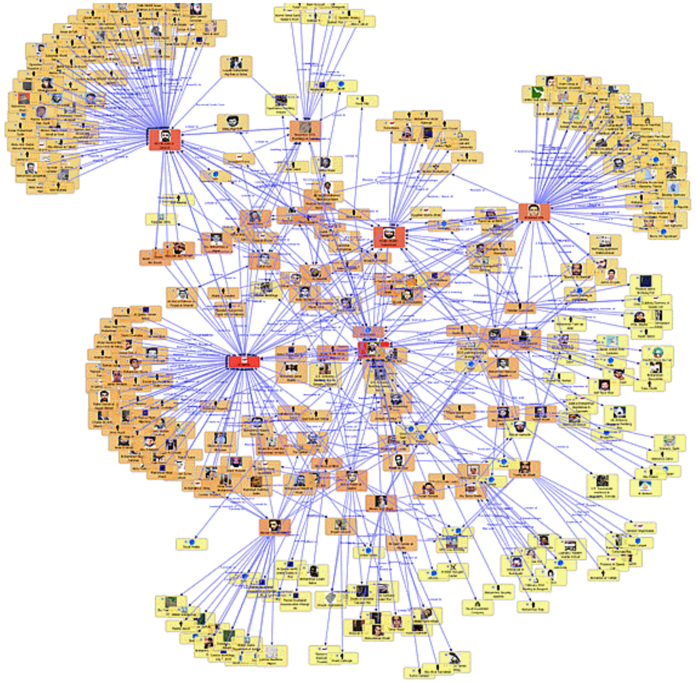
# Graph Analytics

**Applications:**
machine learning and network analysis





**Datasets:** unstructured graphs



Need TBs of memory

# Distributed Graph Analytics

- Distributed-memory clusters are used for in-memory processing of very large graphs
    - D-Galois [PLDI'18], Gemini [OSDI'16], PowerGraph [OSDI'12], ...
- Graph is partitioned among machines in the cluster
    - Many heuristics for graph partitioning (partitioning policies)
- Application performance is sensitive to policy
    - There is no clear way to choose policies

# Motivation

- No clear to choose policies for the user as well as to support for the systems
- Main objectives of a good partitioning policy:
    - Minimize the communication overhead
    - Balance computation load

# Existing Partitioning Studies

- Performed on small graphs and clusters
- Only considered metrics such as edges cut, avg. number of replicas, etc.
- Did not consider work-efficient data-driven algorithms
  - Only topology-driven algorithms evaluated
- Used framework that use similar communication pattern for all partitioning policies
  - Putting some partitioning policies at a disadvantage

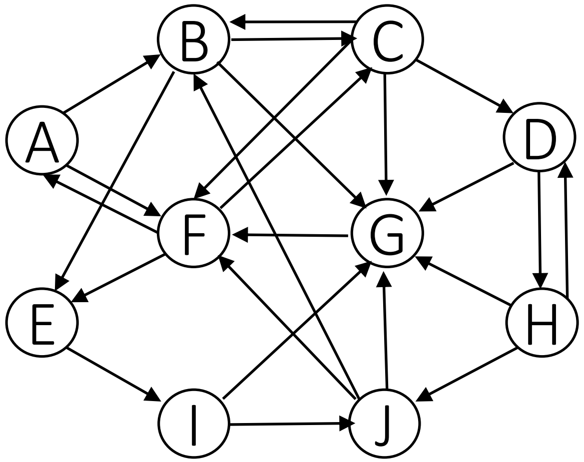
# Contributions

- Experimental study of partitioning strategies for **work-efficient** graph analytics applications:
  - Largest publicly available web-crawls, such as **wdc12** (~1TB)
  - Large **KNL** and **Skylake** clusters with up to 256 machines (~69K threads)
  - Uses the start-of-the-art graph analytics system, **D-Galois** [PLDI'19]
  - Evaluate various kinds of partitioning policies

- Analyze partitioning policies using an **analytical model** and **micro-benchmarking**

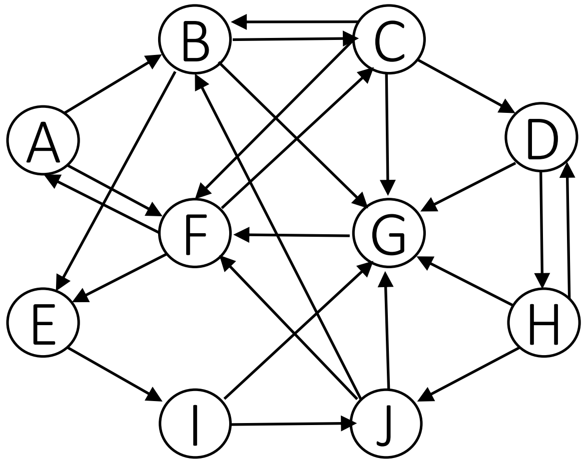
- Present **decision tree** for selecting the best partitioning policy

# Partitioning

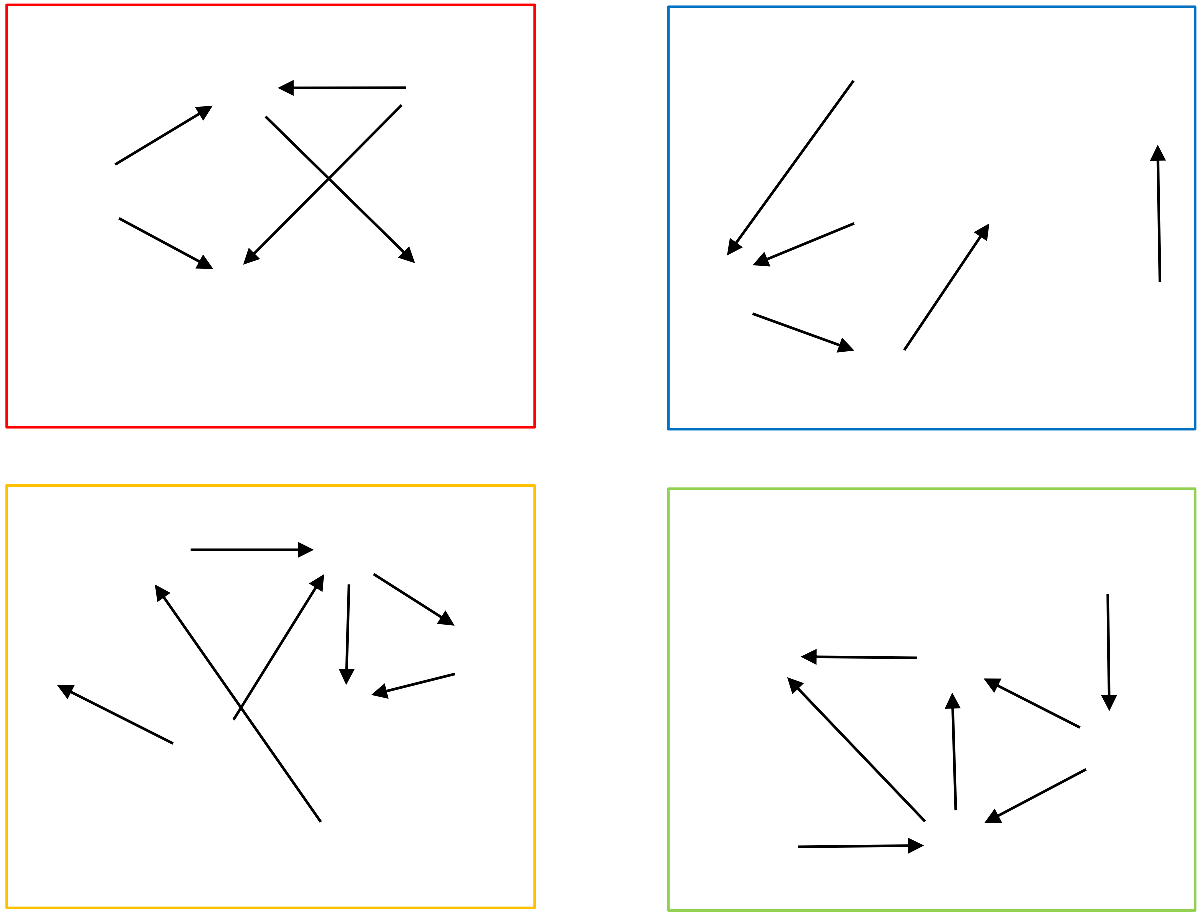



Original graph

# Partitioning

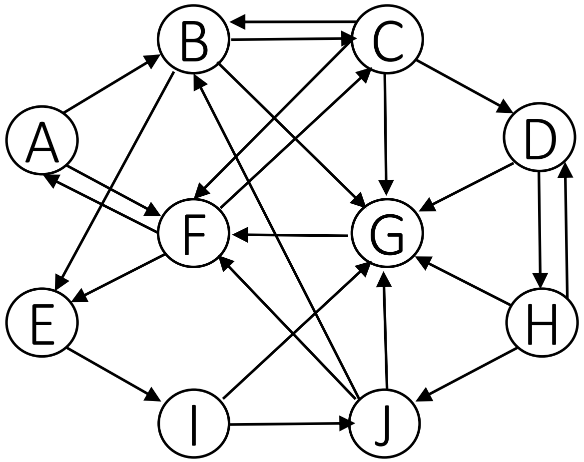



Original graph

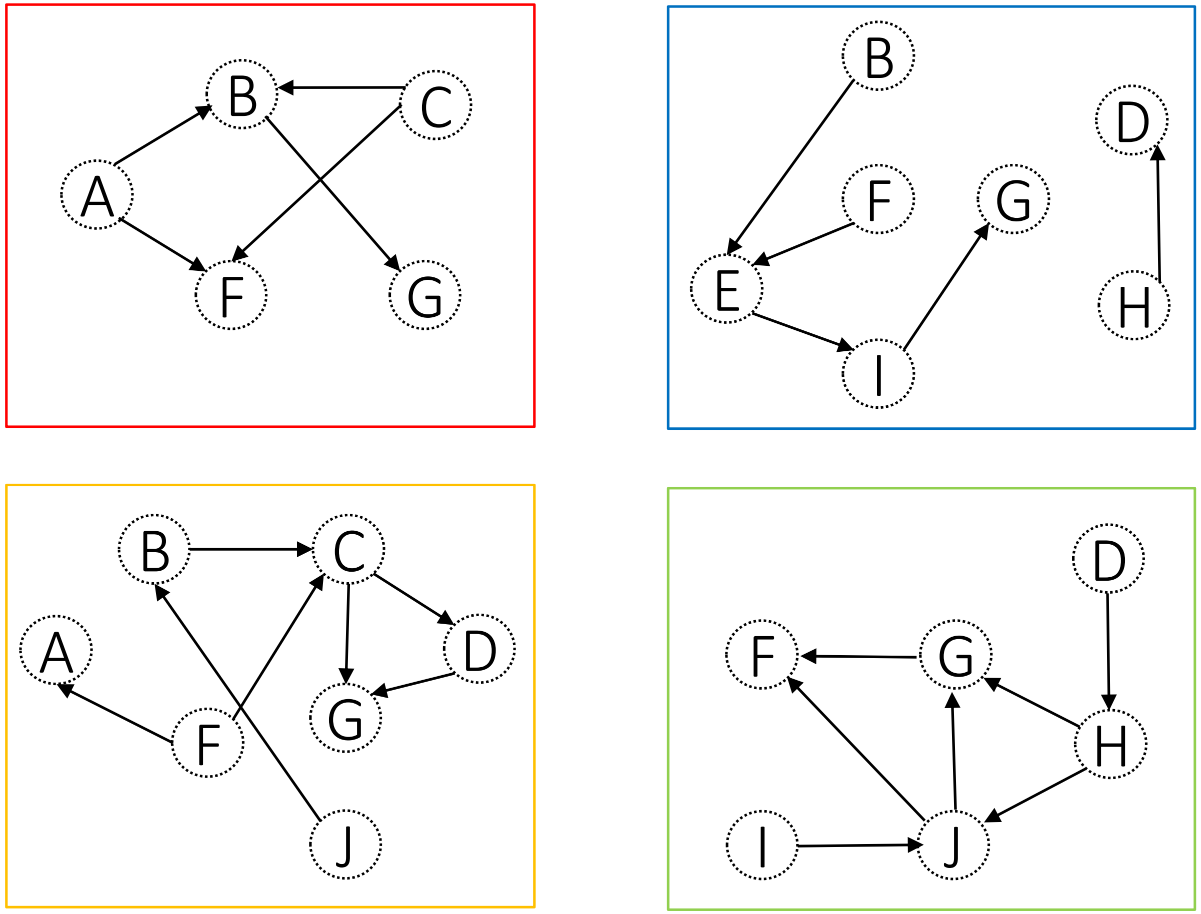


- Each edge is assigned to a unique host

# Partitioning

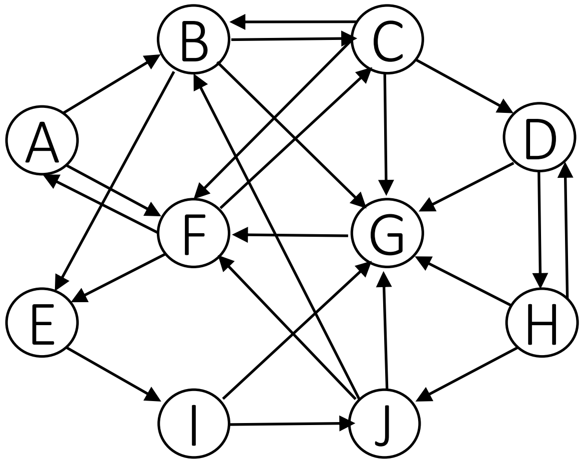



Original graph

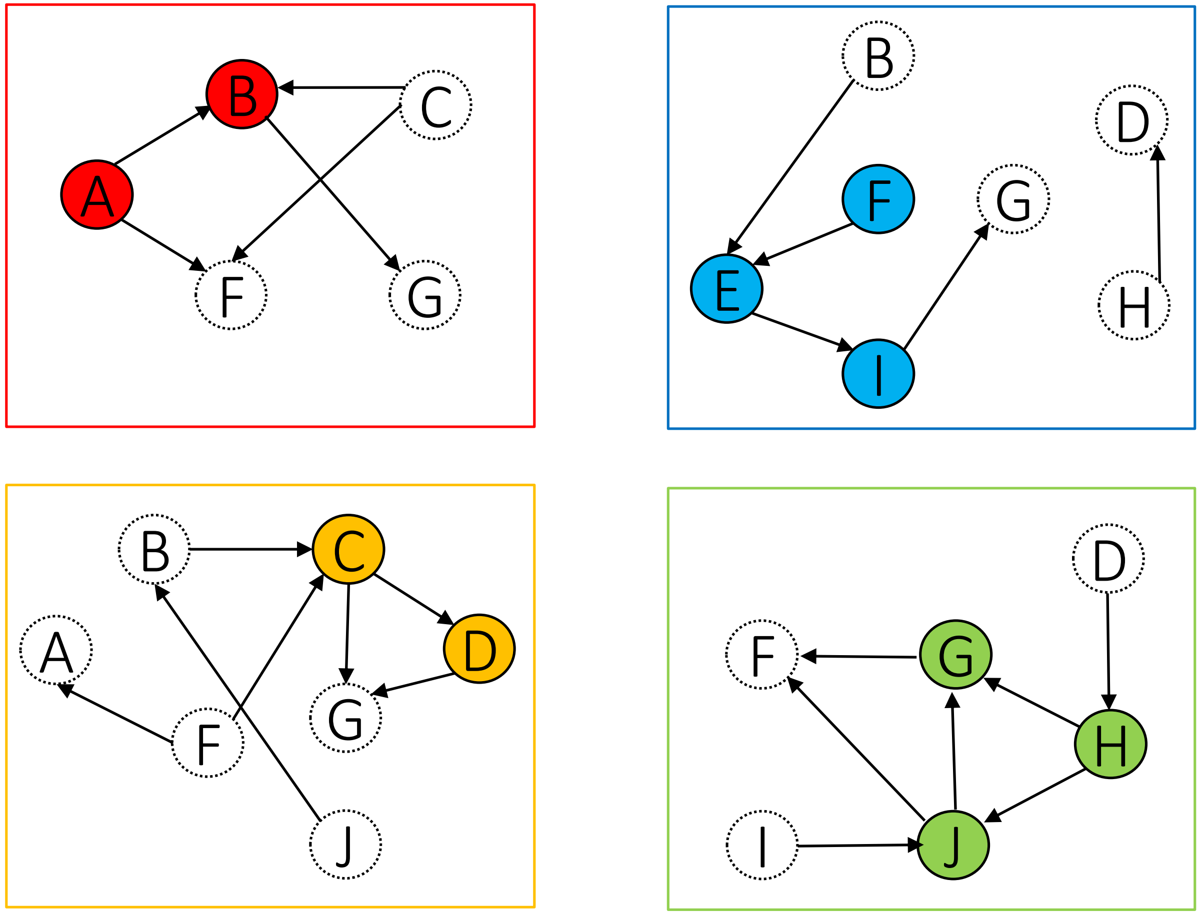



- Each edge is assigned to a unique host
- All edges connect proxy nodes on the same host

# Partitioning

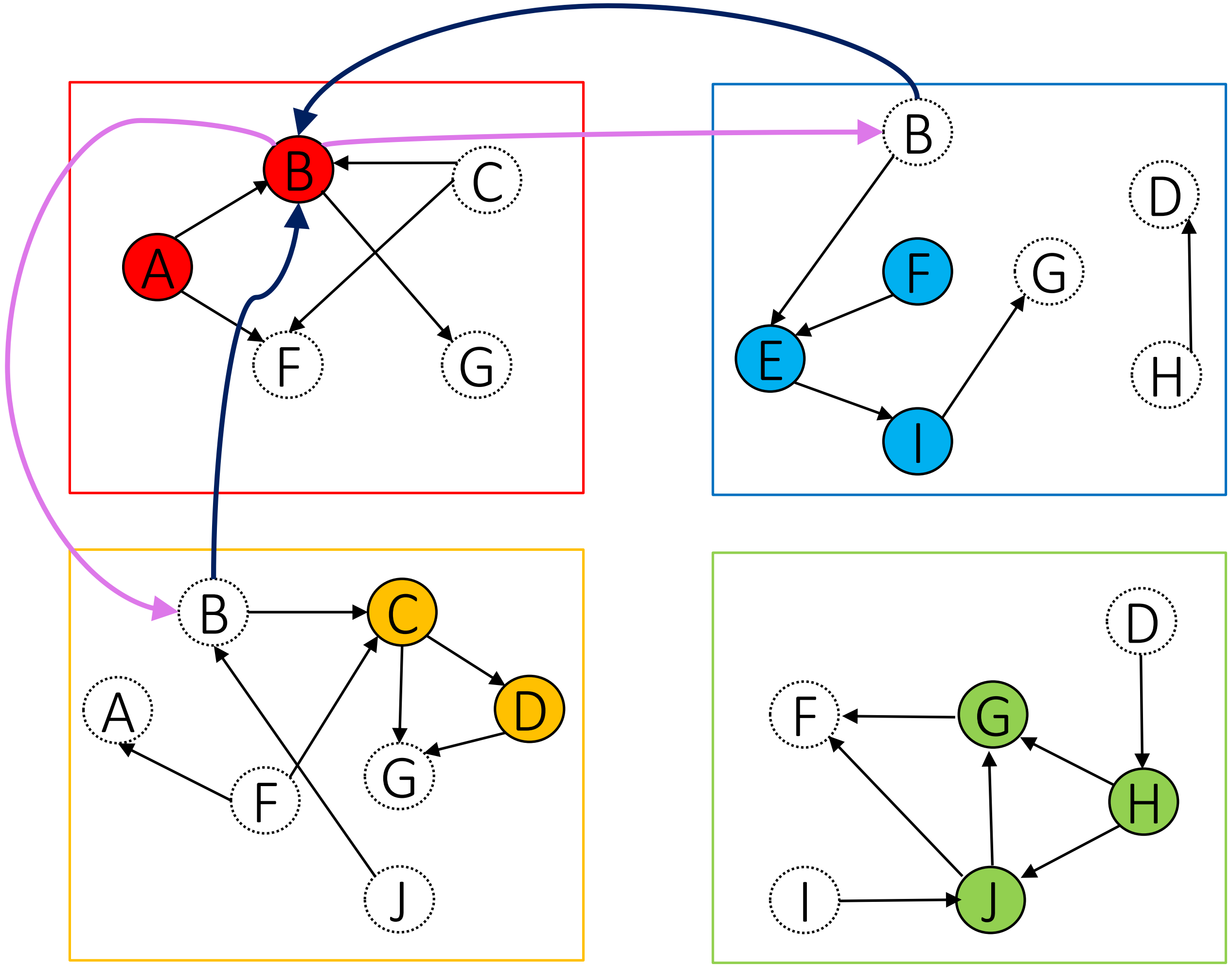



Original graph

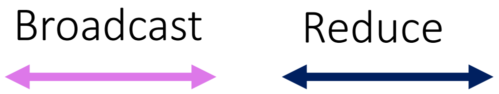



- Each edge is assigned to a unique host
- All edges connect proxy nodes on the same host
- A node can have multiple proxies: one is **master** proxy; rest are **mirror** proxies

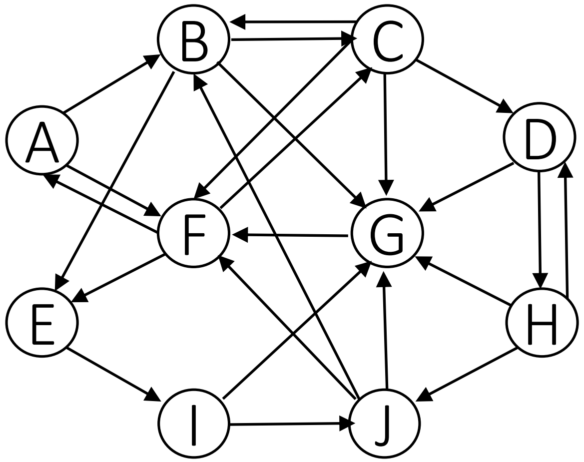
# Synchronization



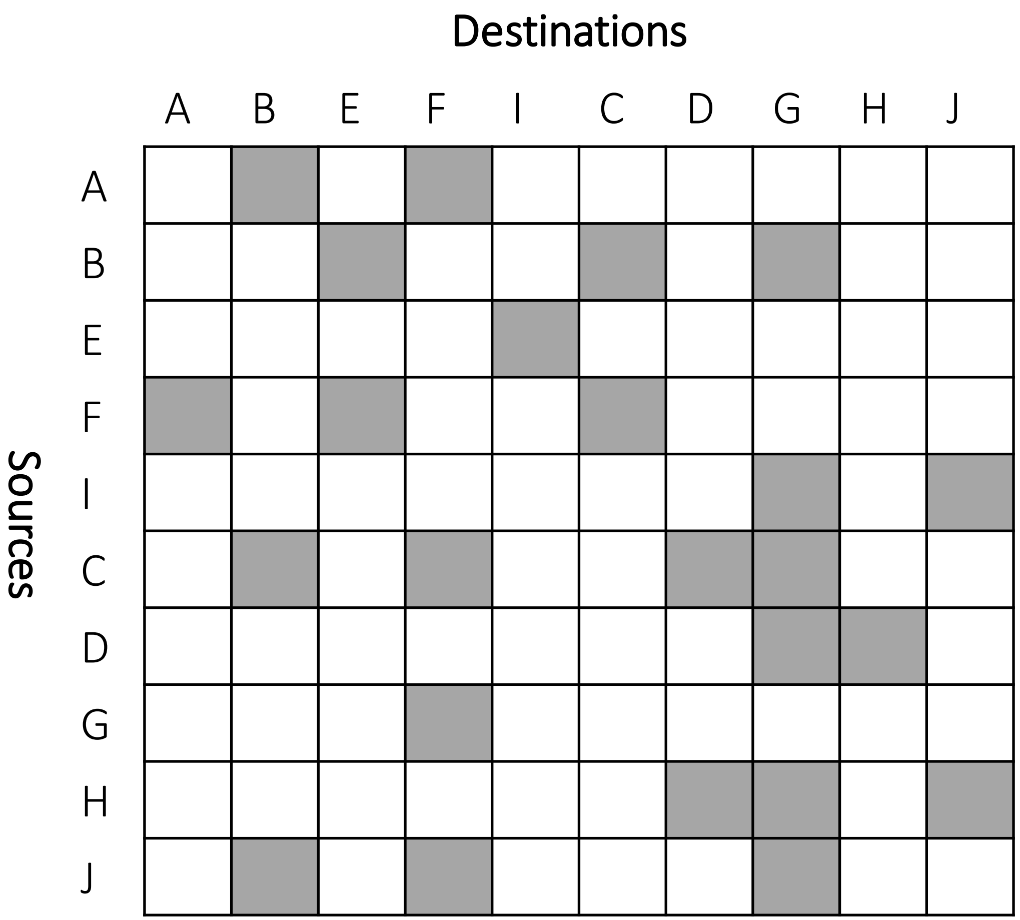

- **Reduce**: Mirror proxies send updates to master
- **Broadcast**: Master sends canonical value to mirrors



# Matrix View of a Graph



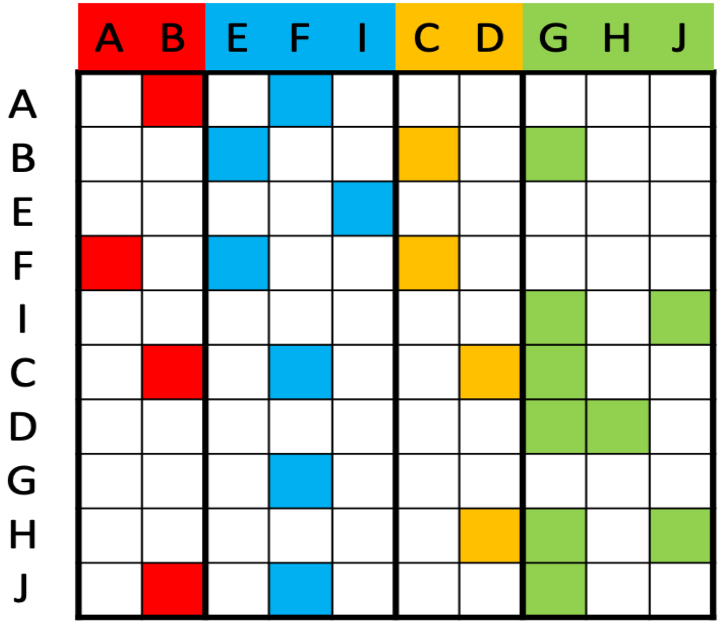

Original graph



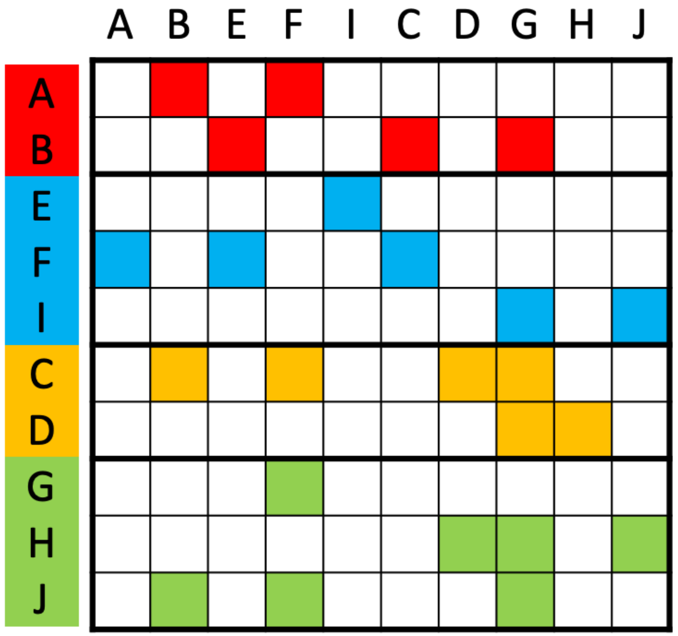

Matrix representation

# Partitioning Policies & Comm. Pattern

1D Partitioning



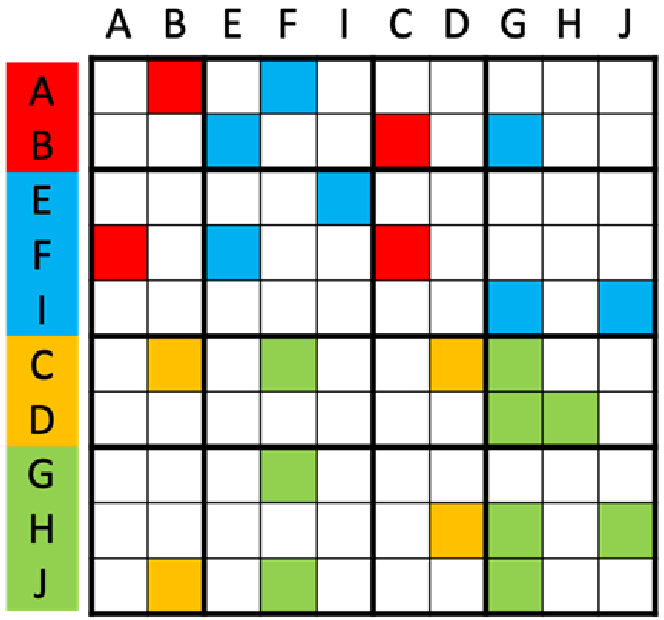

Incoming Edge Cut
(IEC)




Outgoing Edge Cut
(OEC)

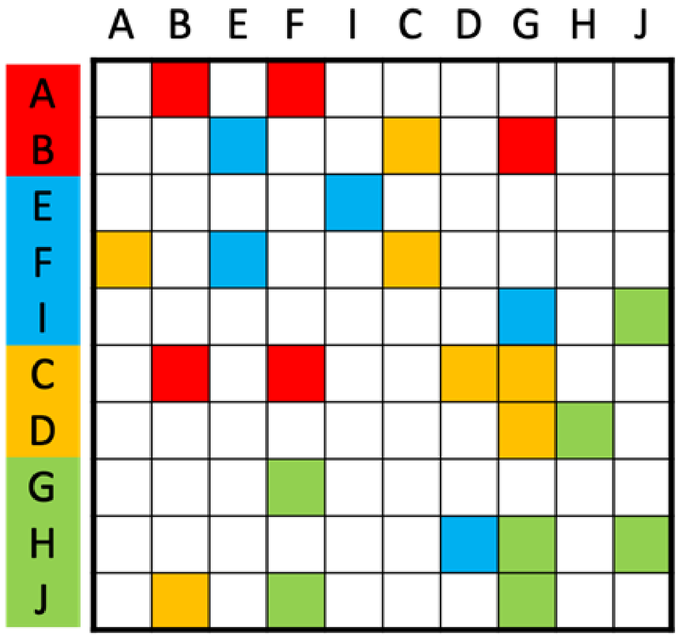
# Partitioning Policies & Comm. Pattern

2D Partitioning




Cartesian Vertex Cut
(CVC)

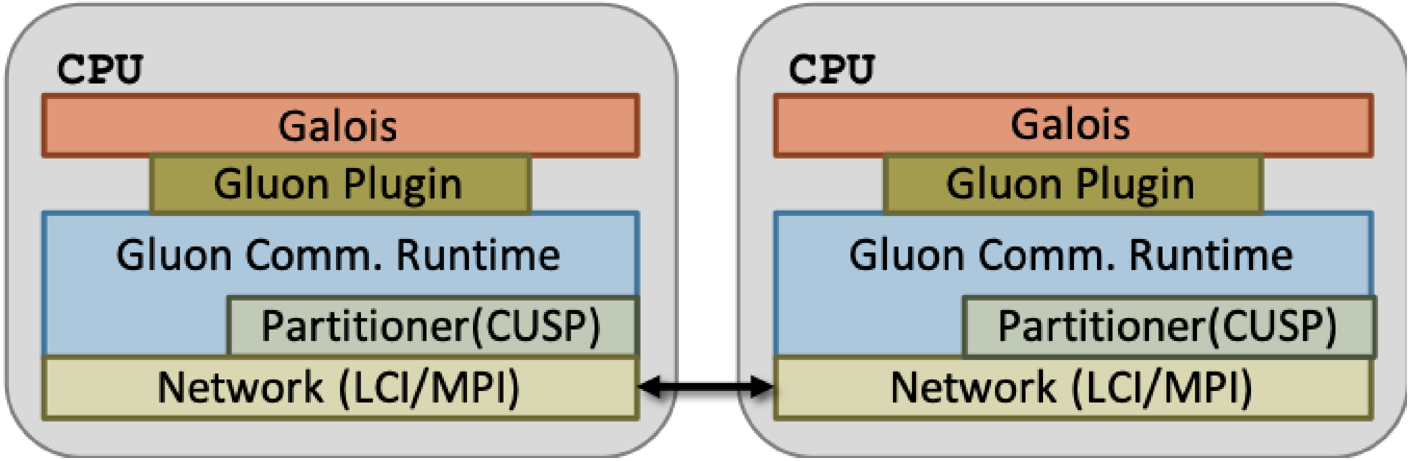
General
Vertex Cut




Hybrid Vertex Cut
(HVC)

# Implementation

- **Partitioning:** State-of-the-art graph partitioner, **CuSP** [IPDSP'19]
- **Processing:** State-of-the-art distributed graph analytics system, **D-Galois** [PLDI'18]
  - Bulk synchronous parallel (BSP) execution
  - Uses **Gluon** [PLDI'19] as communication substrate:
    - *Uses partition-specific communication pattern*



# Experimental Setup

- **Stampede2 at TACC: Up to 256 hosts**
  - KNL hosts:
    - *68 cores (without hyper threading)*
  - Skylake hosts
    - *24 cores*
- **Application:**
  - bc (betweenness centrality)
  - bfs (breadth-first search)
  - cc (connected components)
  - pr (pagerank)
  - sssp (single-source shortest path)

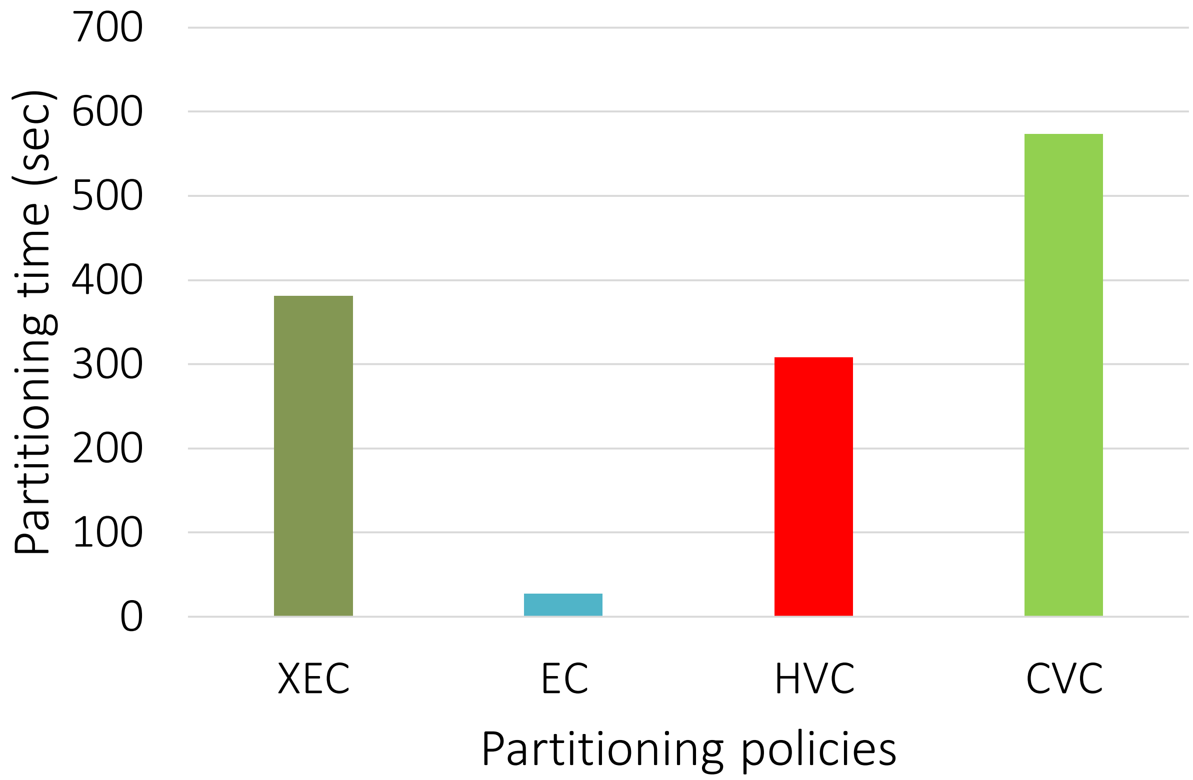
- **Partitioning Policies:**
  - XtraPulp (XEC)
  - Edge Cut (EC)
  - Cartesian Vertex Cut (CVC)
  - Hybrid Vertex Cut (HVC)

| | kron30 | clueweb12 | wdc12 |
|---|---|---|---|
| \|V\| | 1073M | 978M | 3,563M |
| \|E\| | 10,791M | 42,574M | 128,736M |
| Size on Disk | 136GB | 325GB | 986GB |

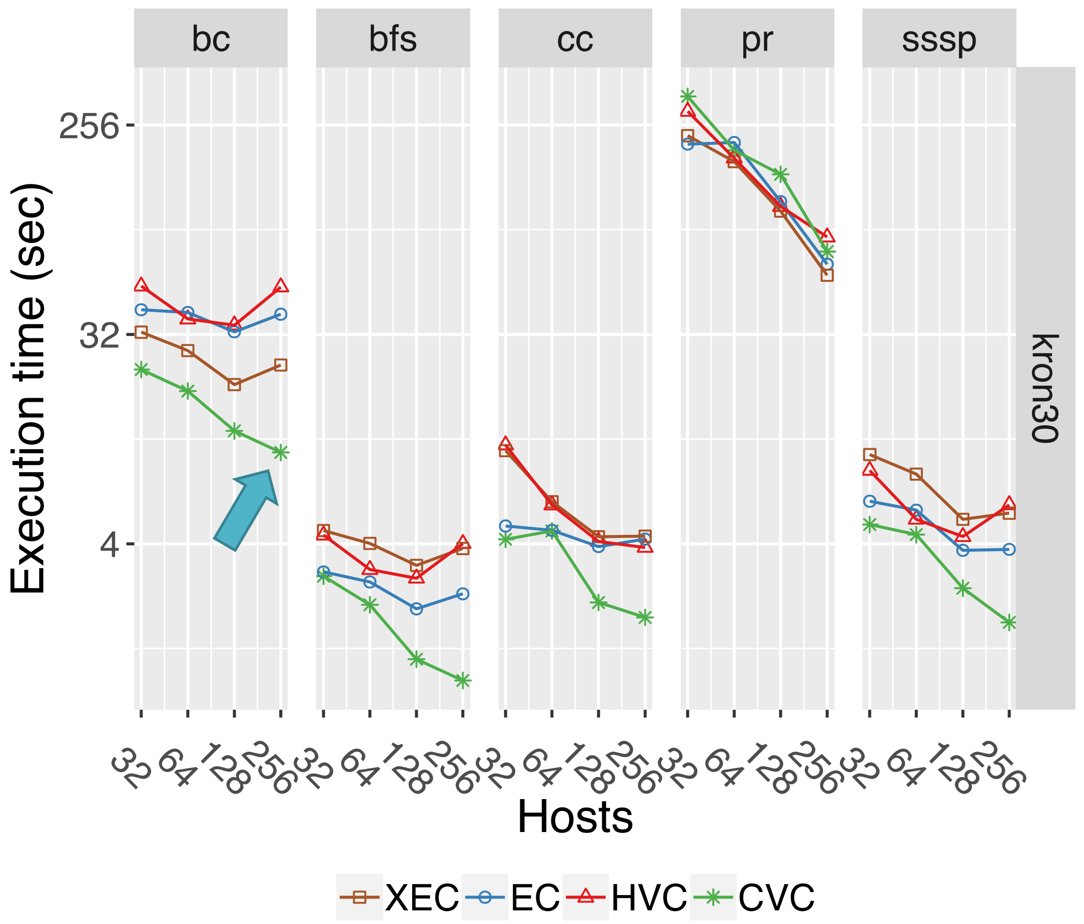
Inputs and their properties

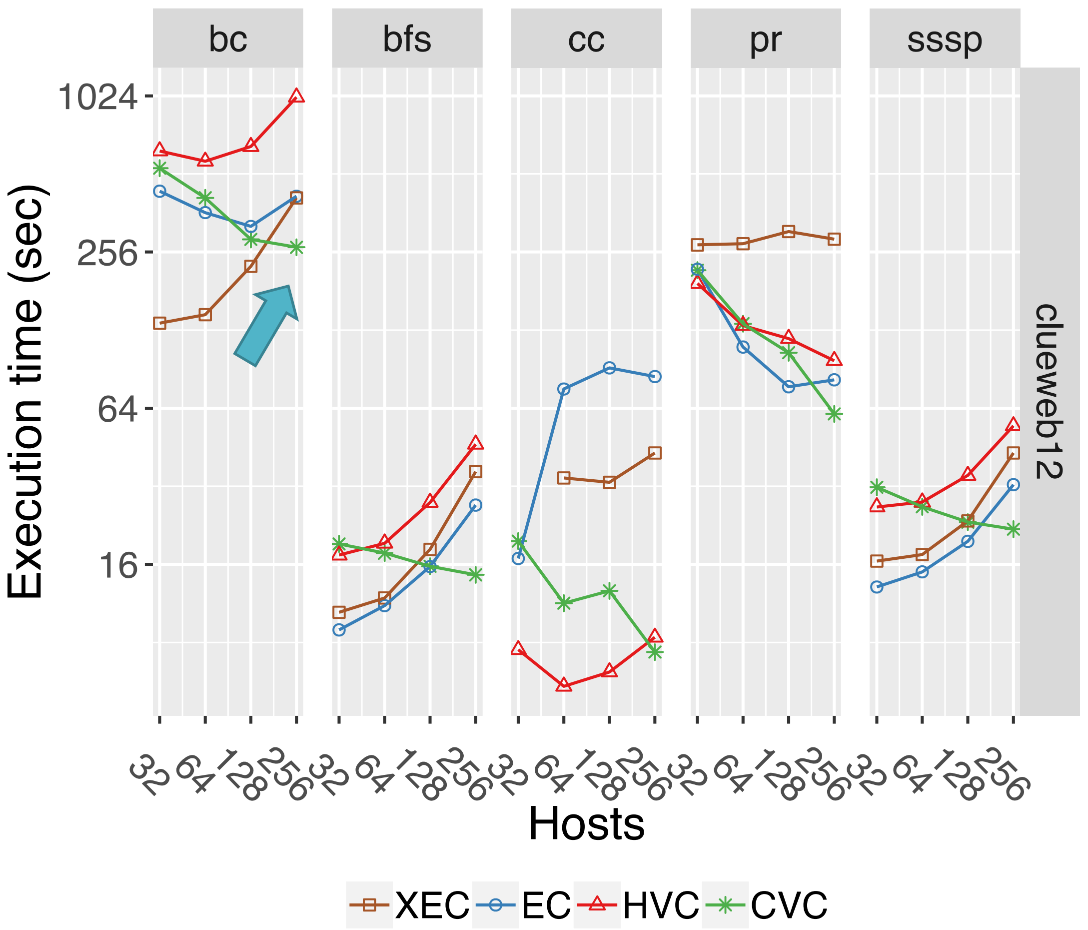
# Partitioning Time (clueweb12)

Partitioning time (sec) on 256 hosts:



# Execution Time



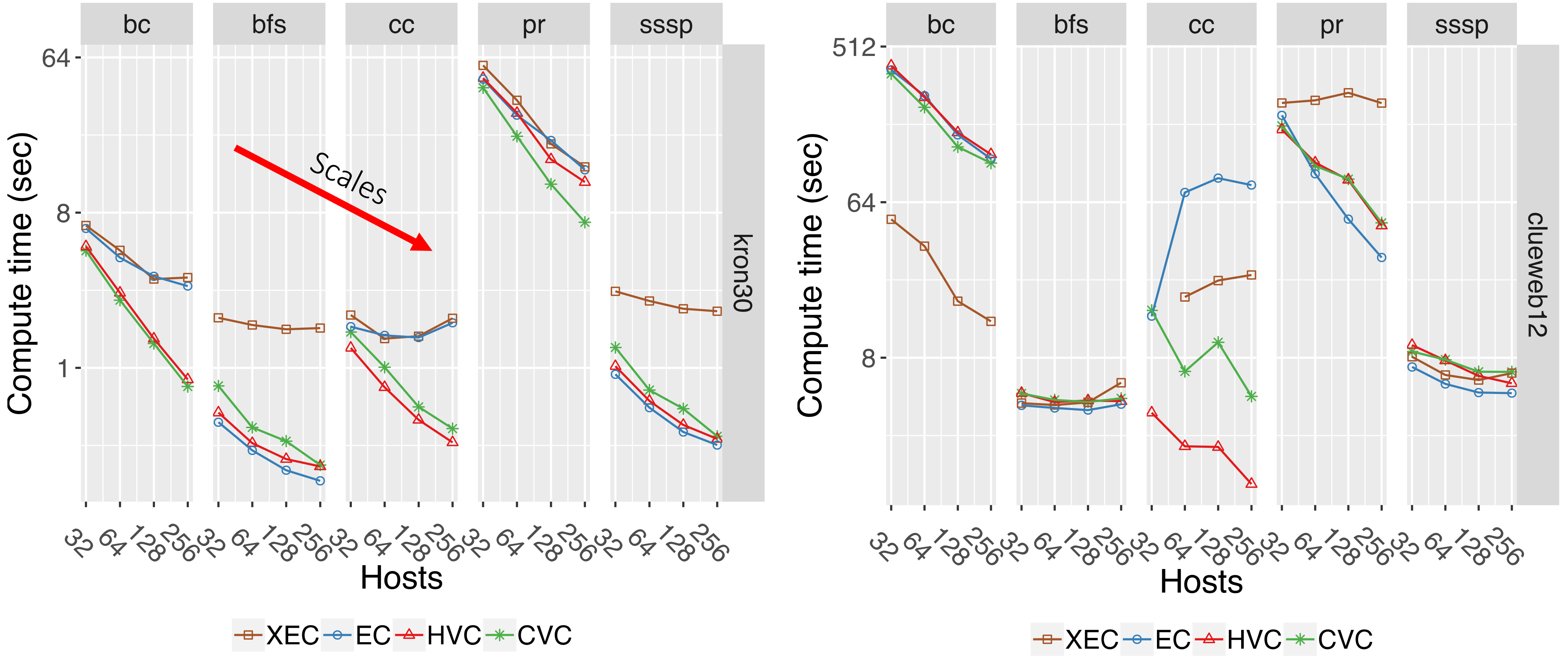



# Load Balance

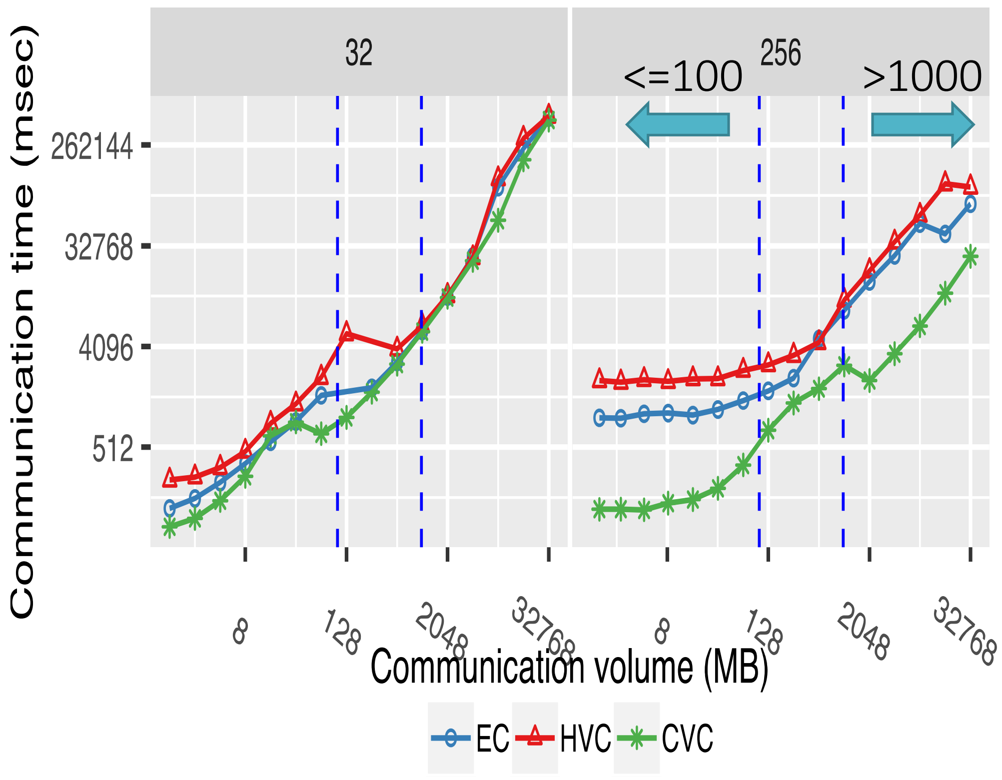
| | | Partitioning time (sec) | | | Max-by-mean edges | | |
|---|---|---|---|---|---|---|---|
| | | **bc bfs sssp** | **cc** | **pr** | **bc bfs sssp** | **cc** | **pr** |
| **kron30** | **XEC** | 304 | 448 | 312 | 1.05 | 1.08 | 1.06 |
| | **EC** | 51 | 76 | 51 | 1.01 | 1.01 | 1.01 |
| | **HVC** | 102 | 130 | 101 | 1.02 | 1.02 | 1.02 |
| | **BVC** | 345 | 379 | 365 | 1.02 | 1.02 | 1.02 |
| | **JVC** | 1006 | 1006 | 1016 | 1.00 | 1.00 | 1.00 |
| | **CVC** | 261 | 288 | 241 | 1.00 | 1.00 | 1.00 |
| **clueweb12** | **XEC** | 381 | 647 | 373 | 3.18 | 8.93 | 14.69 |
| | **EC** | 27 | 152 | 38 | 1.00 | 1.11 | 1.00 |
| | **HVC** | 308 | 374 | 308 | 3.39 | 1.64 | 3.39 |
| | **BVC** | 1179 | 12907 | 12843 | 20.24 | 20.24 | 20.24 |
| | **JVC** | 1904 | 1924 | 1960 | 1.82 | 1.53 | 1.01 |
| | **CVC** | 573 | 1239 | 1119 | 9.16 | 2.03 | 3.26 |
| **wdc12** | **XEC** | OOM | OOM | OOM | OOM | OOM | OOM |
| | **EC** | 109 | 251 | 236 | 1.00 | 1.03 | 1.00 |
| | **HVC** | 3080 | 2952 | 3068 | 1.18 | 1.13 | 1.18 |
| | **BVC** | 8039 | OOM | OOM | 15.44 | OOM | OOM |
| | **JVC** | 5263 | 6570 | 8890 | 1.09 | 1.05 | 1.01 |
| | **CVC** | 2487 | 4276 | 3221 | 1.79 | 1.17 | 1.27 |

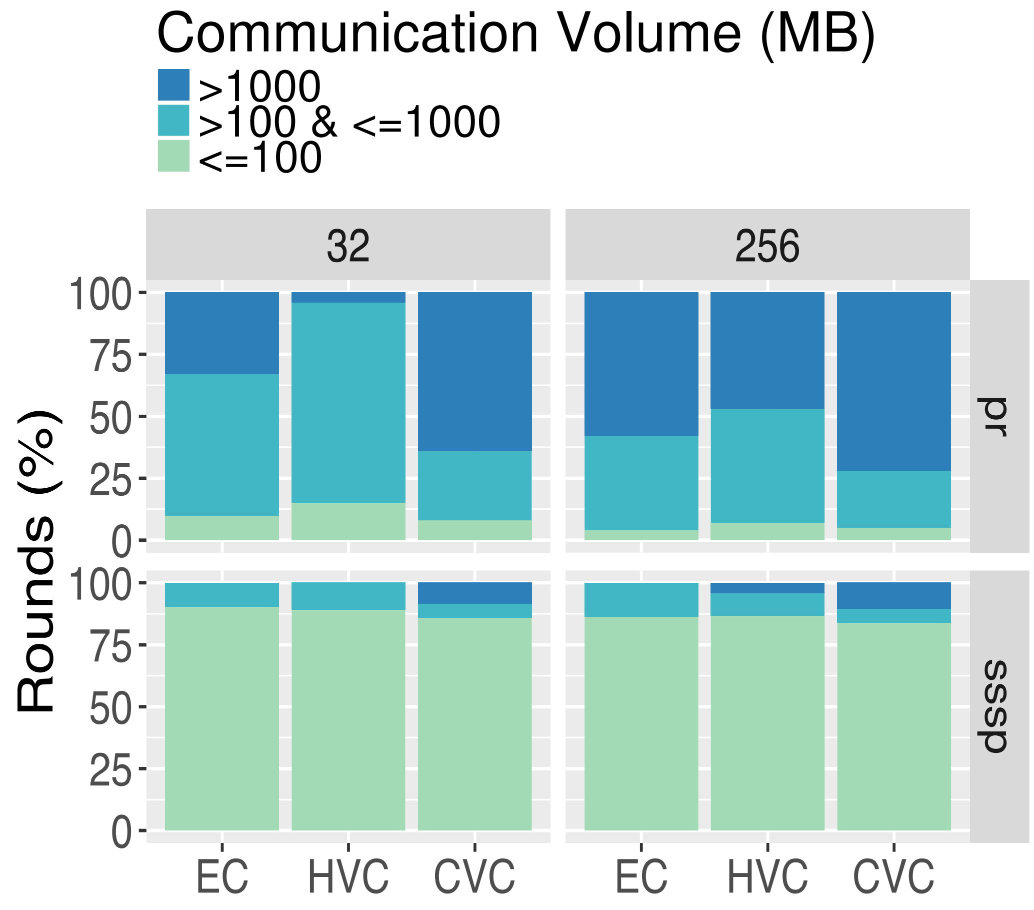
Table 4: Graph partitioning time (includes time to load and construct graph) and static load balance of edges assigned to hosts on 256 KNL hosts.

# Compute Time



# Communication Volume and Time



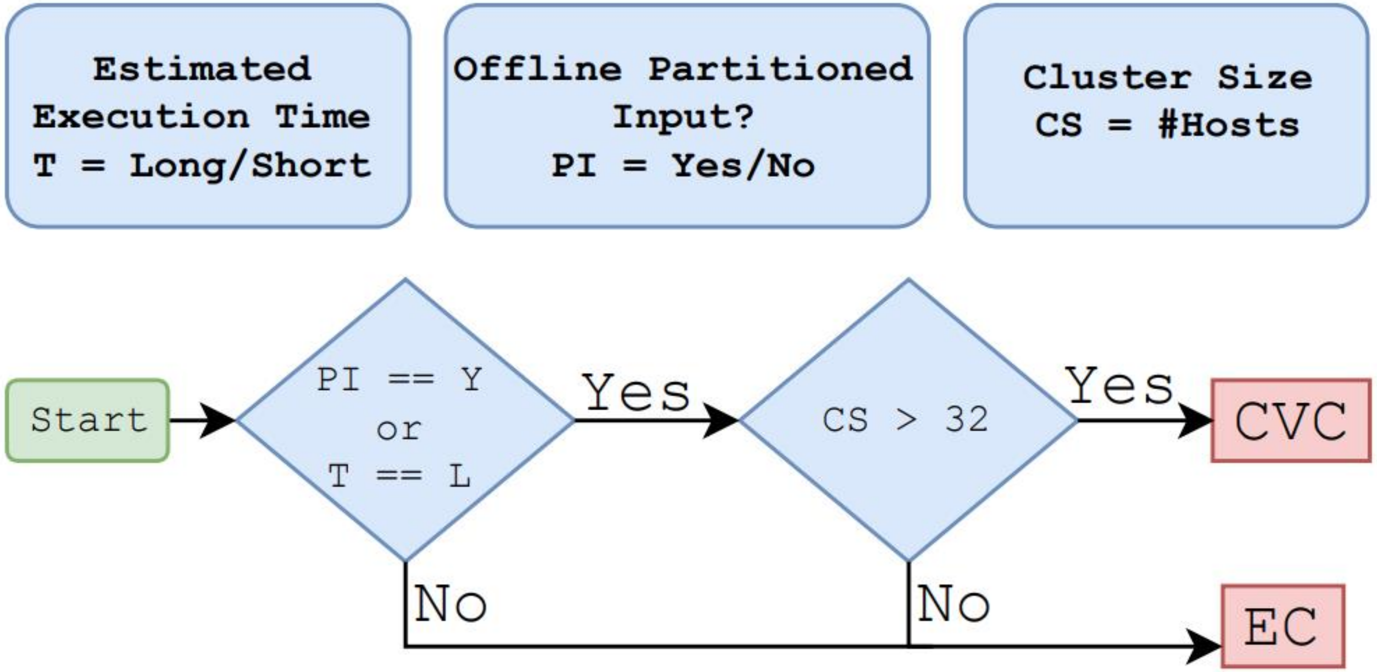



# Best Partitioning Policy (clueweb12)

Execution time (sec):

| | 32 hosts | | | | 256 hosts | | | |
|---|---|---|---|---|---|---|---|---|
| | XEC | EC | HVC | CVC | XEC | EC | HVC | CVC |
| bc | **136.2** | 439.1 | 627.9 | 539.1 | 413.7 | 420 | 1012.1 | **266.9** |
| bfs | 10.4 | **8.9** | 17.4 | 19.2 | 36.4 | 27.1 | 46.5 | **14.6** |
| cc | OOM | 16.9 | **7.5** | 19.6 | 43.0 | 84.7 | 8.4 | **7.3** |
| pr | 272.6 | 219.6 | **193.5** | 217.9 | 286.7 | 82.3 | 97.5 | **60.8** |
| sssp | 16.5 | **13.1** | 26.6 | 31.7 | 43.0 | 32.5 | 54.7 | **21.8** |

# Decision Tree




| | | 8 | 256 |
|---|---|---|---|
| kron30 | bc | 21.79% | 0% |
| | bfs | 0% | 0% |
| | cc | 0% | 0% |
| | pr | 0% | 0% |
| | sssp | 0% | 0% |
| clueweb12 | bc | 0% | 0% |
| | bfs | 0% | 0% |
| | cc | 12.84% | 0% |
| | pr | 0% | 0% |
| | sssp | 0% | 11.34% |

% difference in execution time between policy chosen by decision tree vs. optimal

# Key Lessons

- Best performing policy depends on:
  - Application
  - Input
  - Number of hosts (scale)

- EC performs well at small scale but CVC wins at large scale

- Graph analytics systems must support:
  - Various partitioning policies
  - Partition-specific communication pattern

Thank you