Crayons – An Azure Cloud Based Parallel System for GIS Overlay Operations

SUSHIL K. PRASAD

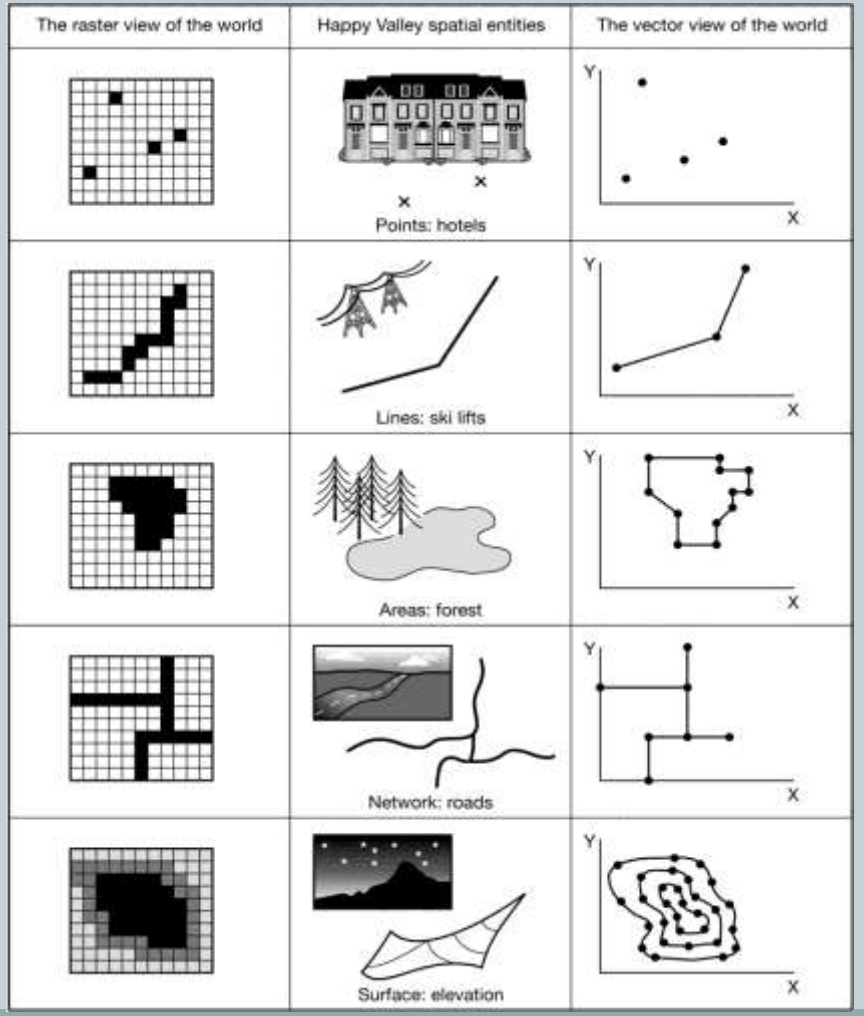
COMPUTER SCIENCE DEPARTMENT
GEORGIA STATE UNIVERSITY

Outline

- Geographic Information System (GIS) Data and Computation
- Computational Problems & Our Solution
- Crayons Architecture
  - Centralized Dynamic Load Balancing
  - Distributed Static Load Balancing
  - Distributed Dynamic Load Balancing
- Performance
- Current & Future Work

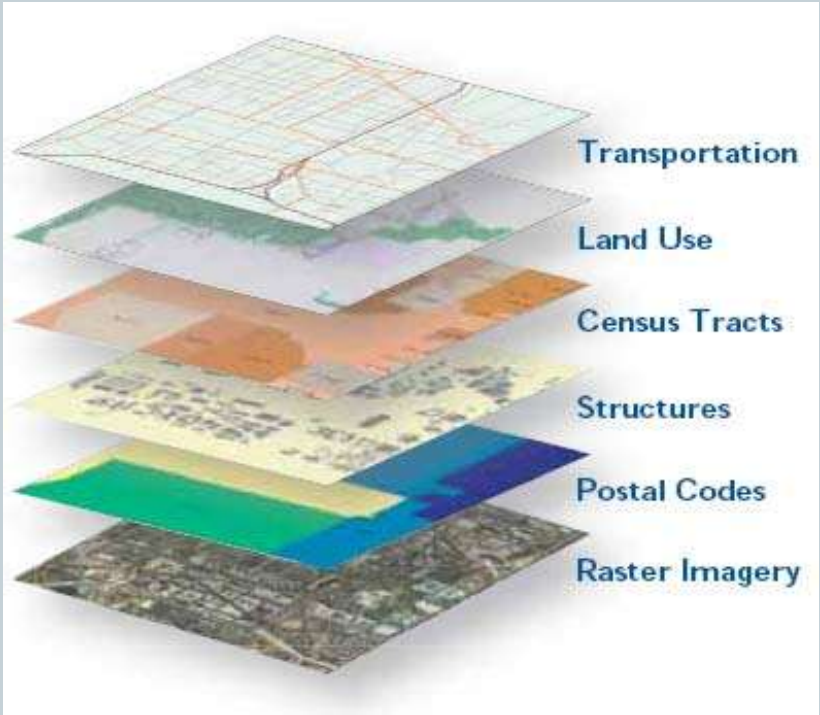
Raster Vs. Vector Data in GIS

- Raster Data – Akin to an Image
  - Divided into cells
  - Each cell represents dominating feature
- Vector Data – Collection of objects defined using coordinates
  - Polygons, Lines, Points
  - Better visualization



Spatial Overlay Operations

- Real world GIS data representation through thematic layers
- Layers are overlaid to collect aggregated information



Applications of Spatial Overlay

- Emergency Response
  - Hurricane swath's map overlaid with city's map to find out safe rescue shelters
- City Planning
  - Road network's map overlaid with public buildings' map
- Habitat Analysis
  - Effect of population density on wildlife habitat

Typical File Sizes

- GIS Data grows enormously over time
- Beyond the capability of a sequential system
- Can easily take hours to days for processing

Source	Example Type	Description	File Size
US Census (Census.gov 2011)	Block Centroids	Block centroids for entire US	705 MB
	Block Polygons	2000 Block polygons for the state of Georgia	108 MB
	Blockgroup Polygons	2000 Blockgroup polygons for the state of Georgia	14 MB
GADoT (GDOT 1916)	Roads	Road centerlines for 5-county Atlanta metro	130 MB
USGS (USGS 1879)	National Hydrography Data set	Hydrography features for entire US	13.1 GB
	National Landcover Data set	Landcover for entire US	3-28 GB
JPL (NASA 1936)	Landsat TM	pan-sharpened 15m resolution	4 TB
Open Topography (Open Topography Facility)	LIDAR	LIDAR point clouds 1-4 pts/sq. ft	0.1-1 TB

Computation Problem & Solution

- GIS Data-Intensive and Irregular Computation
- Desktop Sequential Processing is the state-of-the-art
- Our Solution: First end-to-end cloud system
  - Speedup of 30x, skewed small data 10x
  - Scaling to 100 Azure workers
- Open Problems: File i/o, Multicore/GPU algorithms for overlay and R-tree

Why Cloud & Azure

- On-demand Cloud Computing
- Azure Cloud over others
  - Opportunity to research on emerging platform
    - NSF/Microsoft funds
  - Problems
    - Virtual Machine Configuration, reconfiguration, file upload,
    - fast synchronization and messaging among processors,
    - Lack of MPI/Map-reduce libraries
    - Unstable API, Low fidelity of Azure simulator
  - In-built resilience of storage infrastructure

Windows Azure Platform

- Computation
  - Web Role
  - Worker Role
- Storage
  - Queue Storage – communication
  - Blob Storage – Large data stores
  - Table Storage – Organized data storage

Clipper Library

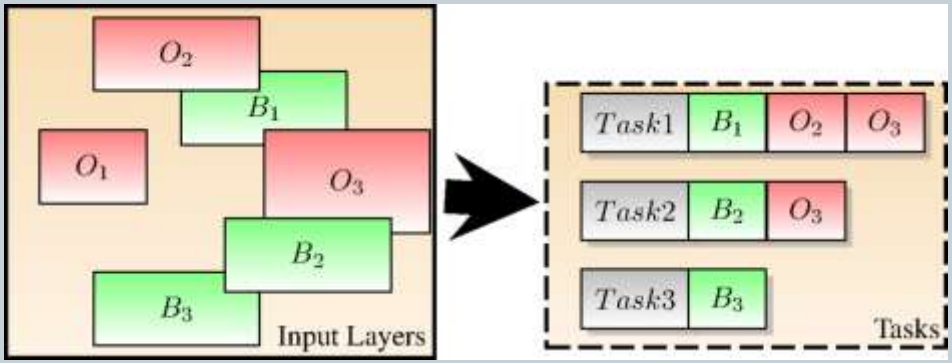
- Created by Alan Murta
- Supports Union, Intersection, XOR, and Difference operations
- Usable due to open-architecture of Crayons
  - Library is packed as a dll file
- Not Multithreaded – open research problem

Crayons' Framework

- Three different flavors
  - Centralized Dynamic Load Balancing
  - Distributed Static Load Balancing
  - Distributed Dynamic Load Balancing

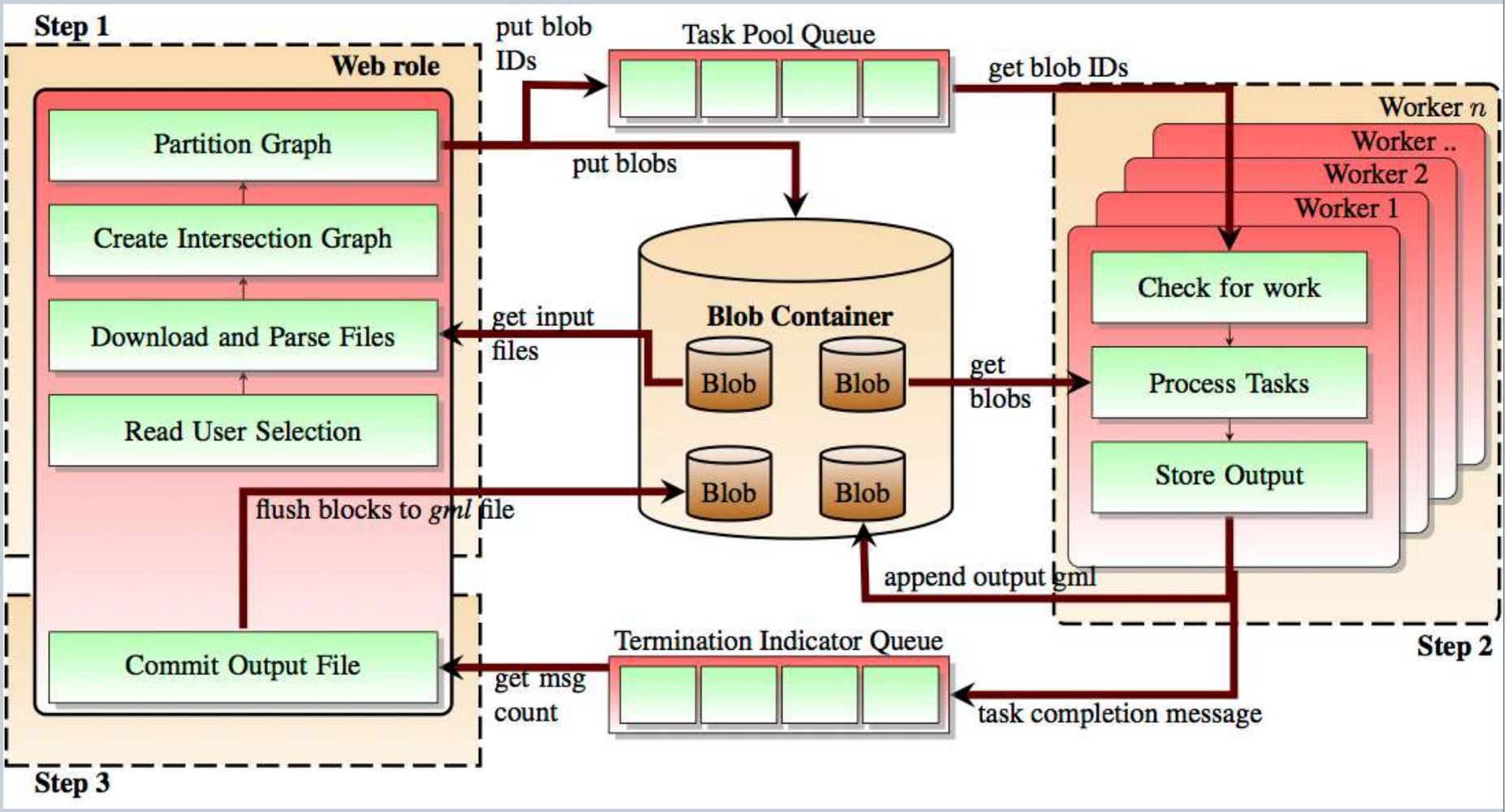
Centralized Dynamic Load Balancing

- Web role does most of the tasks
  - Download and Parse Files, Create Intersection Graph, and Partition and create tasks



- Shared task pool for all workers
- Demand-supply imbalance

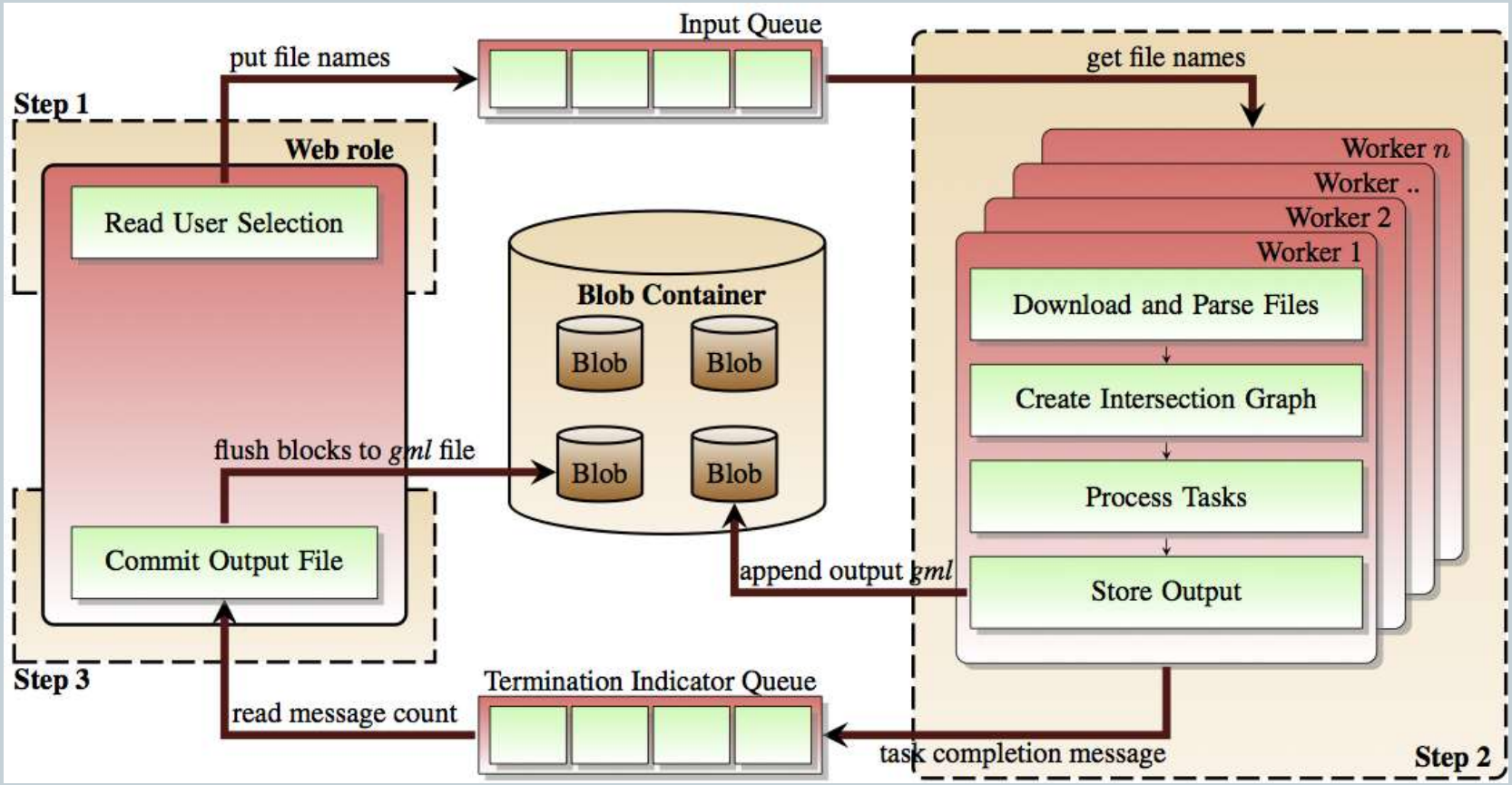
Centralized Dynamic Load Balancing



Distributed Static Load Balancing

- Web role is relieved of computation
- Workers create work individually
- No sharing of work
- No demand-supply imbalance
- Skewed load distribution can affect performance

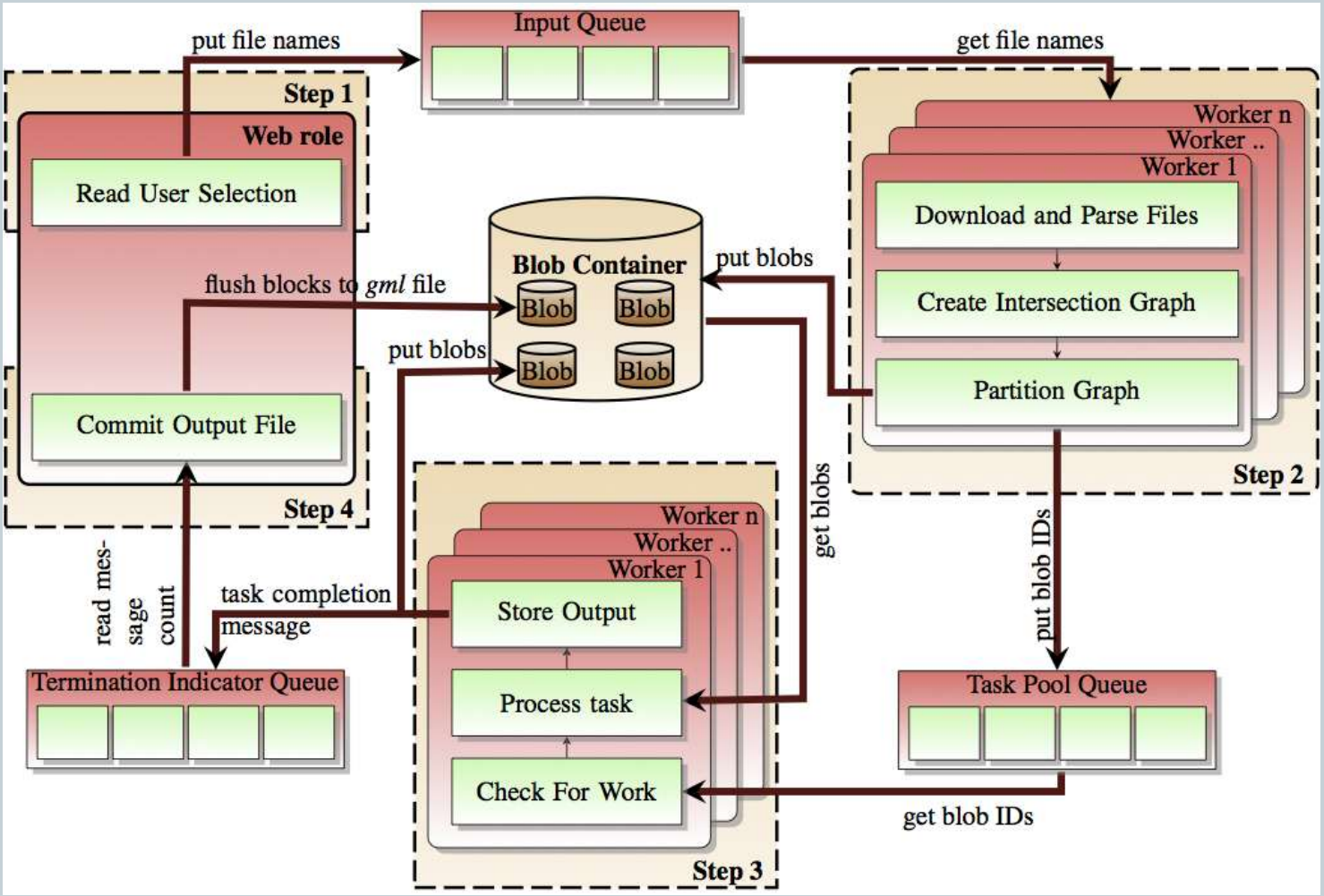
Distributed Static Load Balancing



Distributed Dynamic Load Balancing

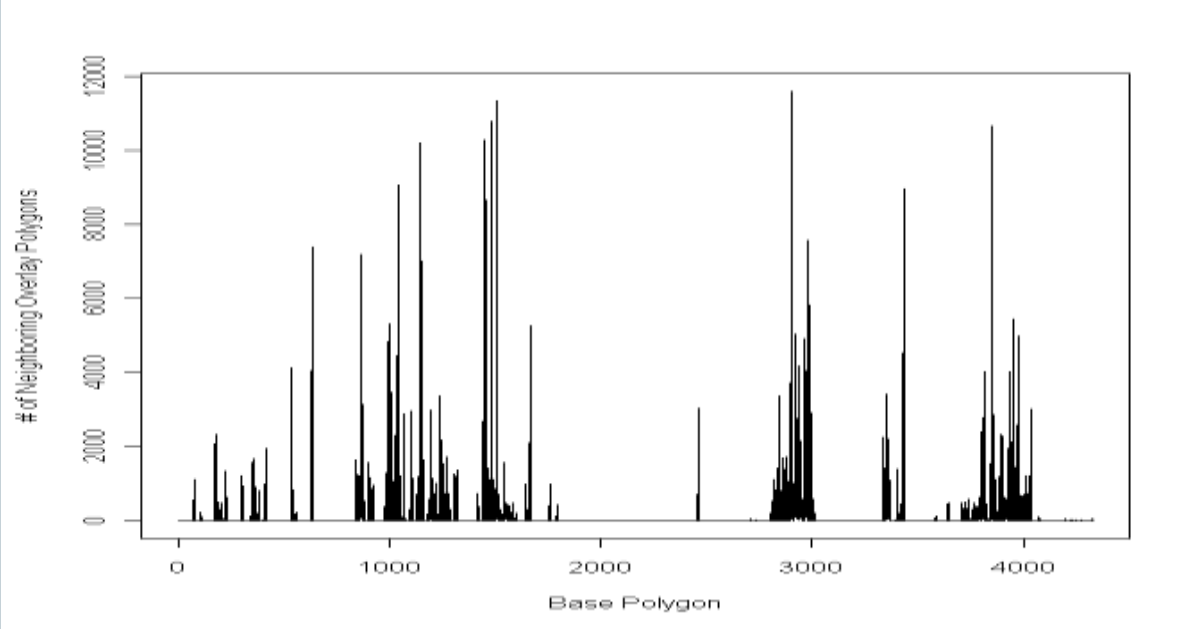
- Web role still doesn’t do much
- Workers create work
- Work is shared among all workers

Distributed Dynamic Load Balancing



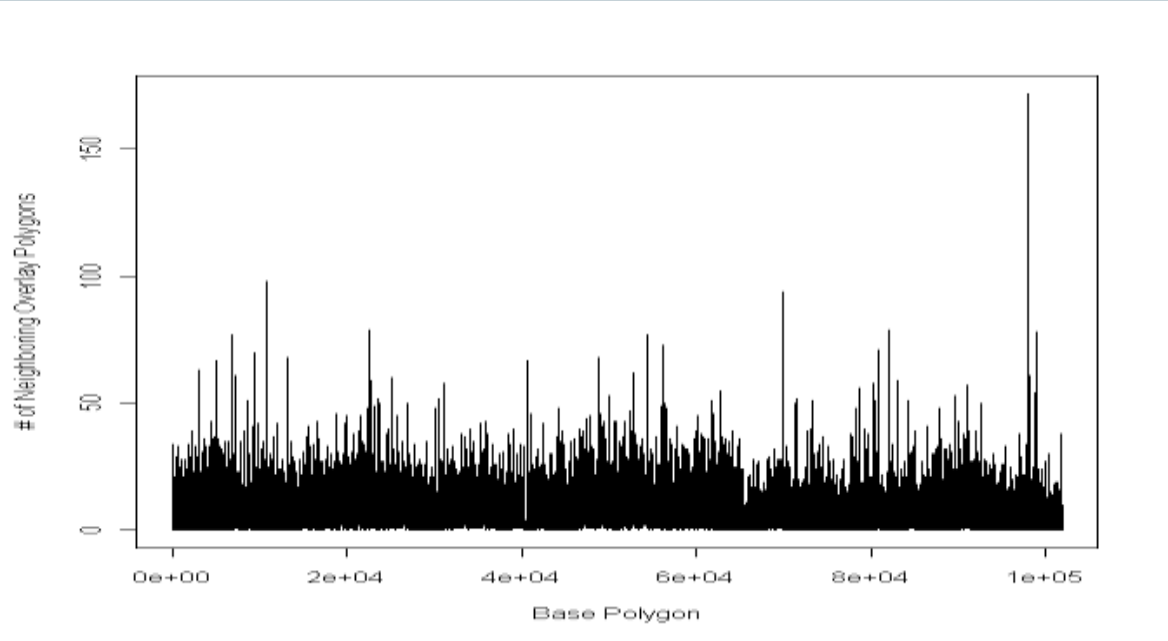
Input Data Sets

- Smaller input data set
  - File Sizes = 16 MB, 770 MB
  - Polygons: Base Layer = 4332, Overlay Layer = 502,674
  - Highly skewed load distribution

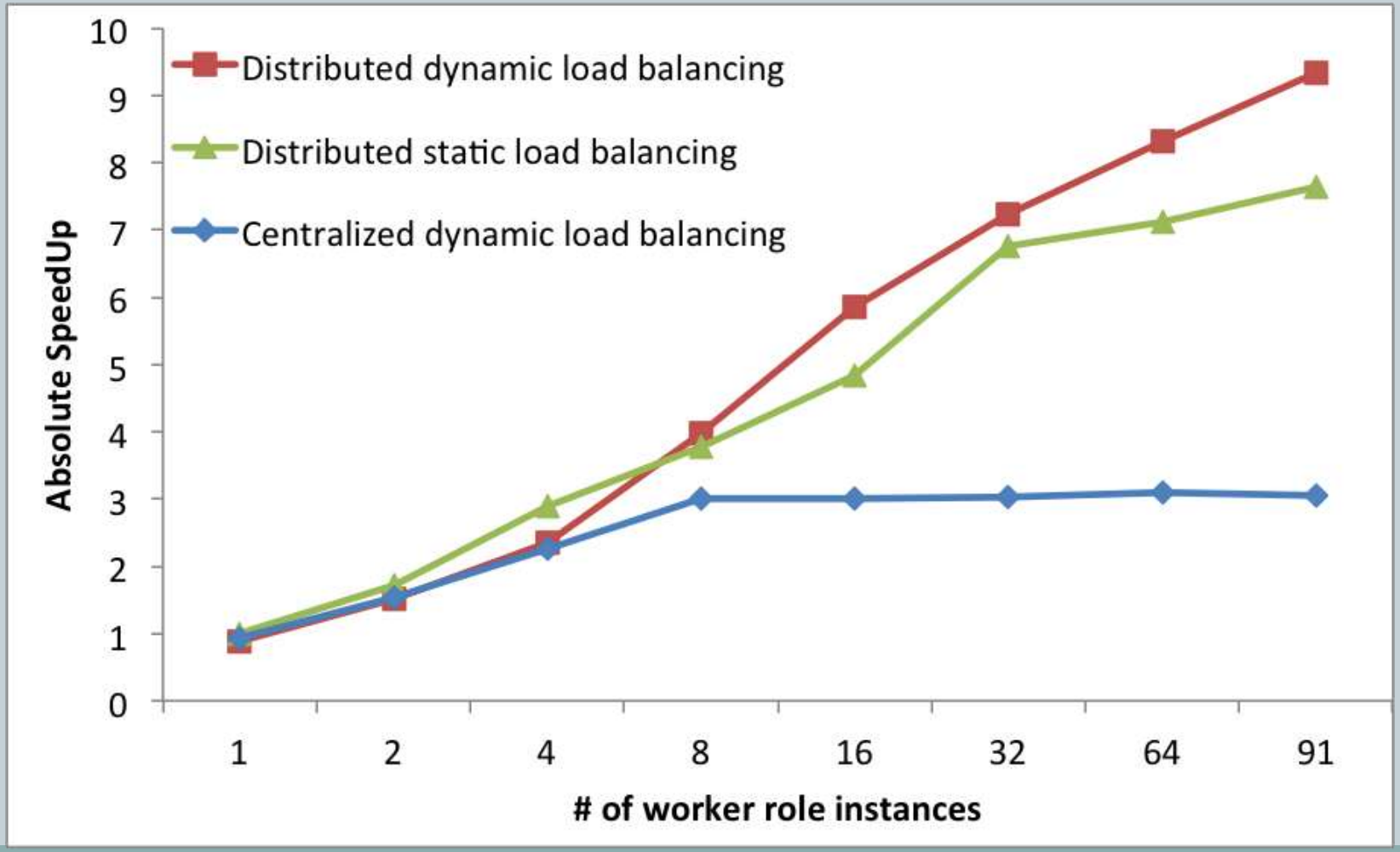


Input Data Sets

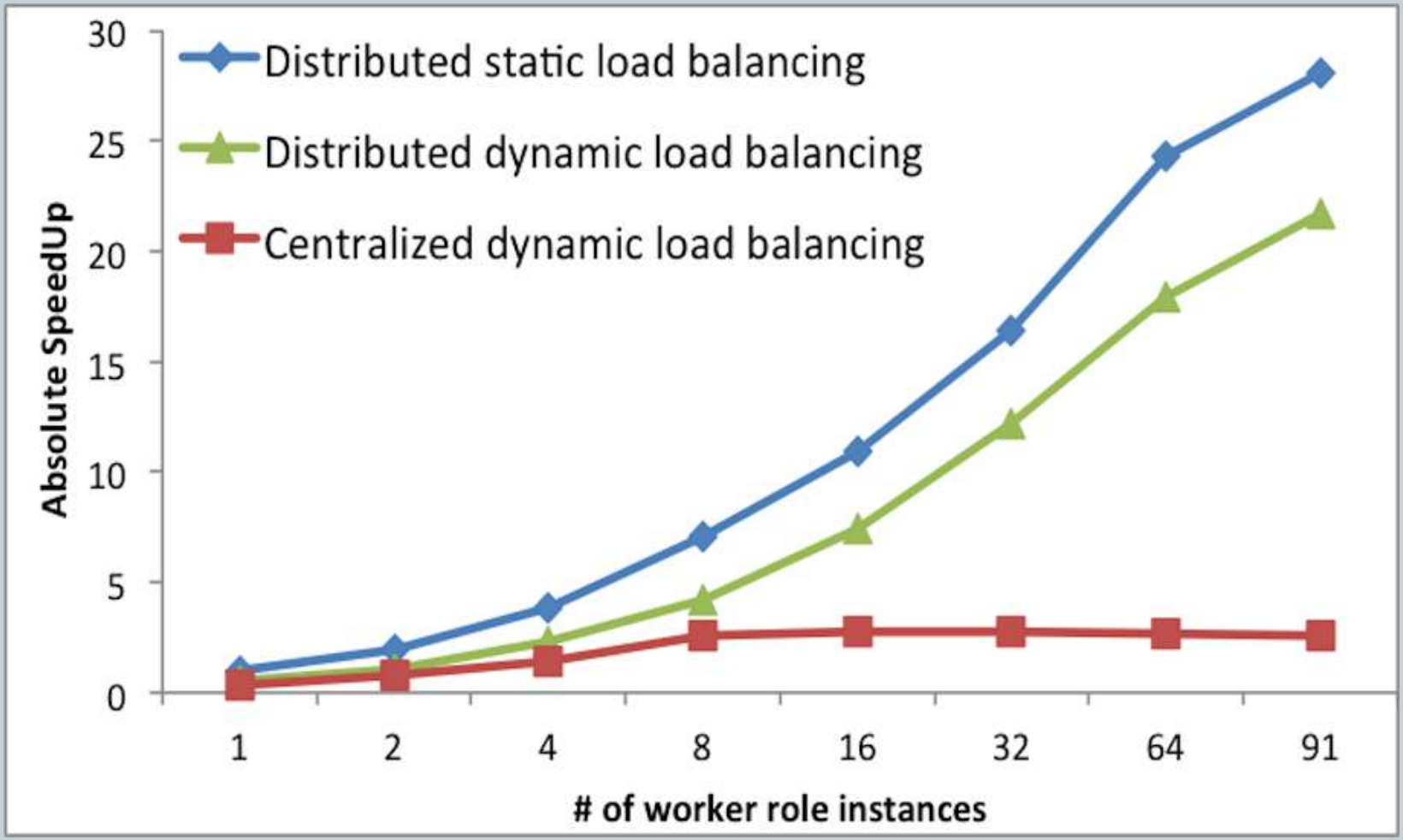
- Larger input data set
  - File Sizes = 242 MB, 318 MB
  - Polygons: Base Layer = 101,860 Overlay Layer = 128,682
  - Comparatively uniform load distribution



End-to-end Speedups over Small Dataset

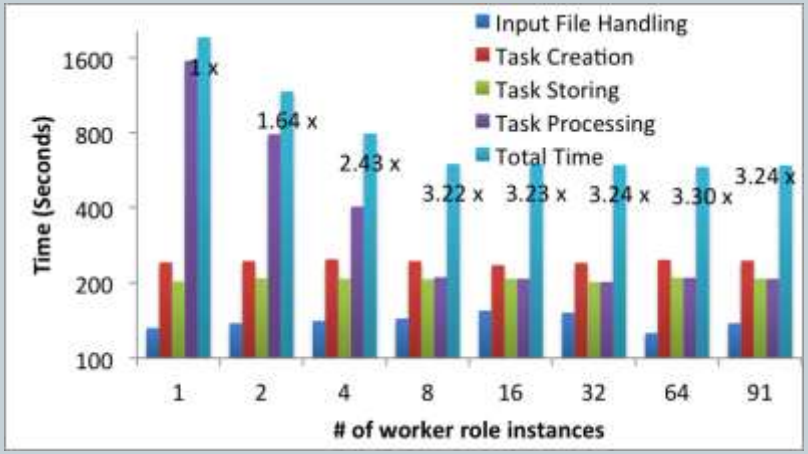


End-to-end Speedups over Large Dataset

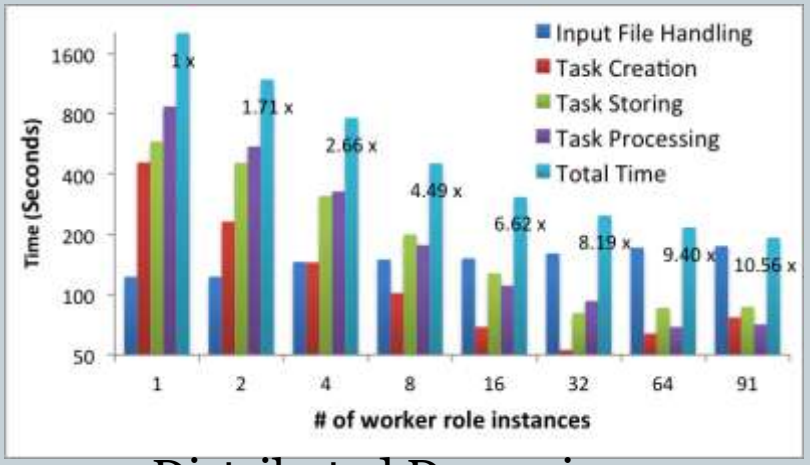


Individual process timings

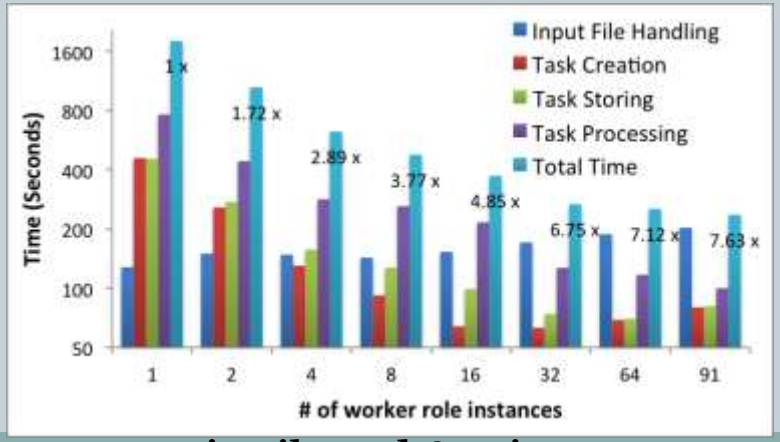
- Execution times for small dataset



Centralized



Distributed Dynamic

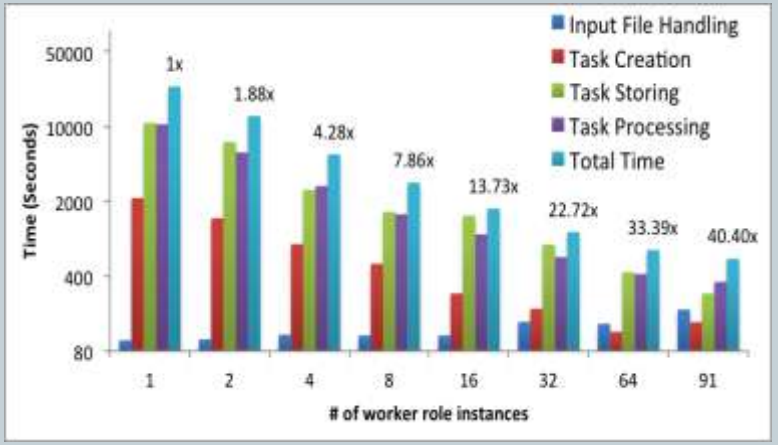


Distributed Static

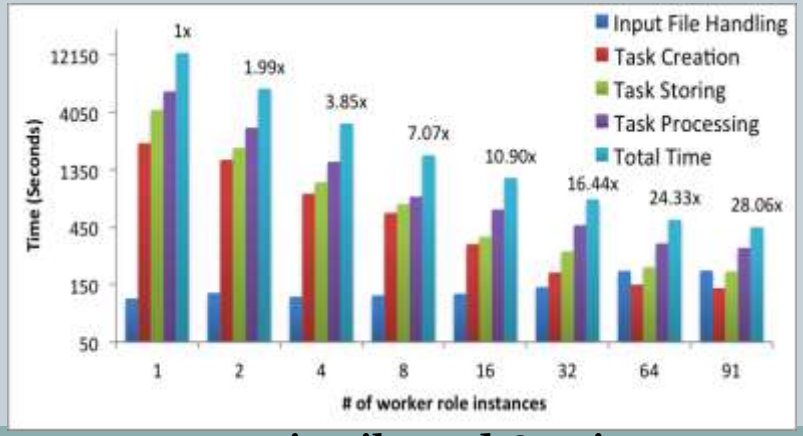
Individual process timings

- Execution times for large dataset

Centralized



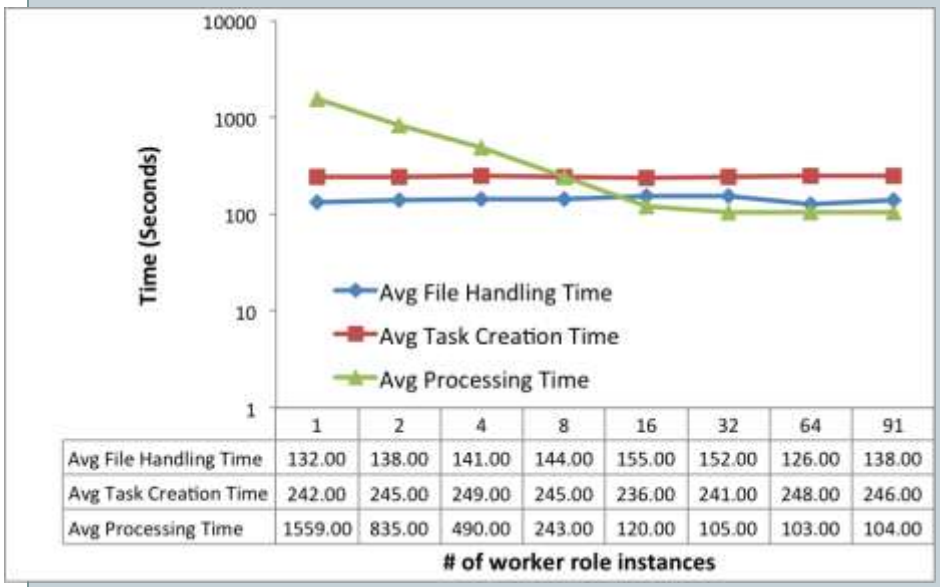
Distributed Dynamic



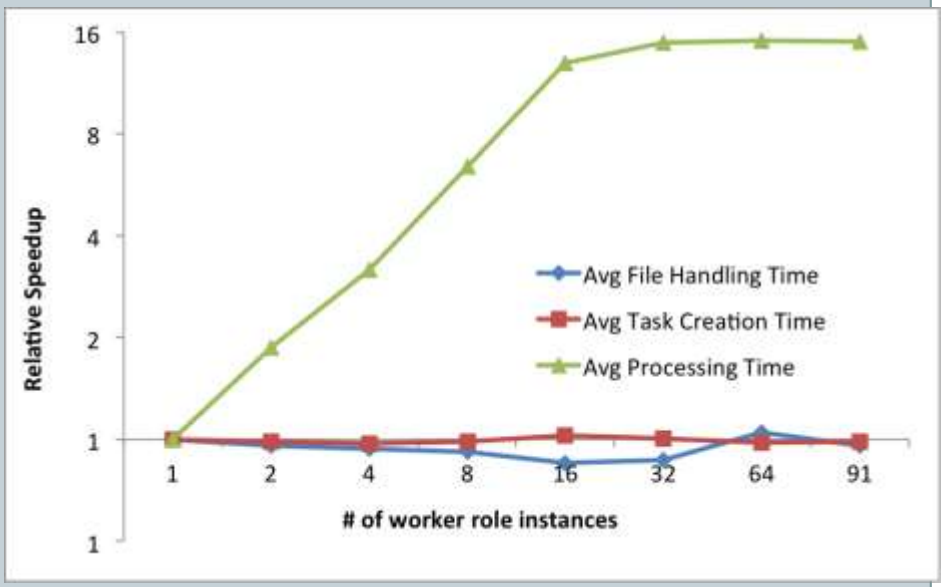
Distributed Static

Average timings across all workers

- Supply demand imbalance starts after 16 processors



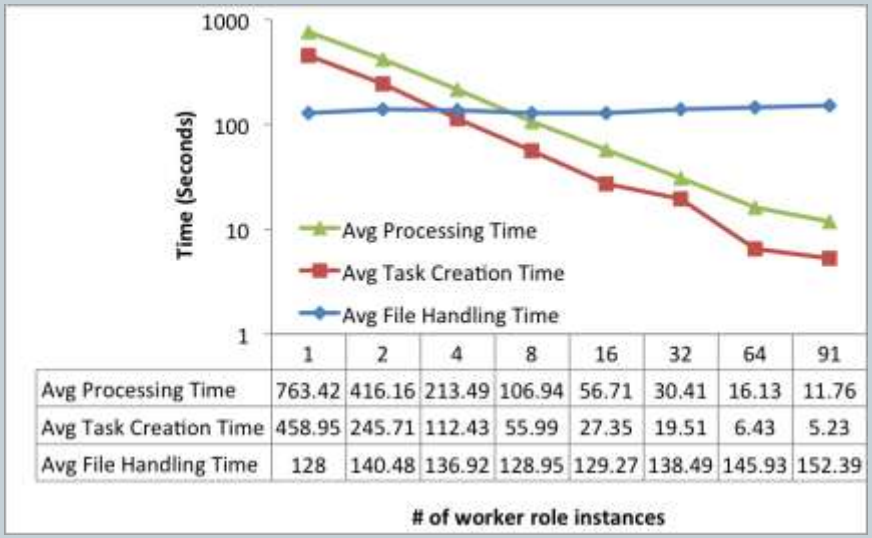
	1	2	4	8	16	32	64	91
Avg File Handling Time	132.00	138.00	141.00	144.00	155.00	152.00	126.00	138.00
Avg Task Creation Time	242.00	245.00	249.00	245.00	236.00	241.00	248.00	246.00
Avg Processing Time	1559.00	835.00	490.00	243.00	120.00	105.00	103.00	104.00



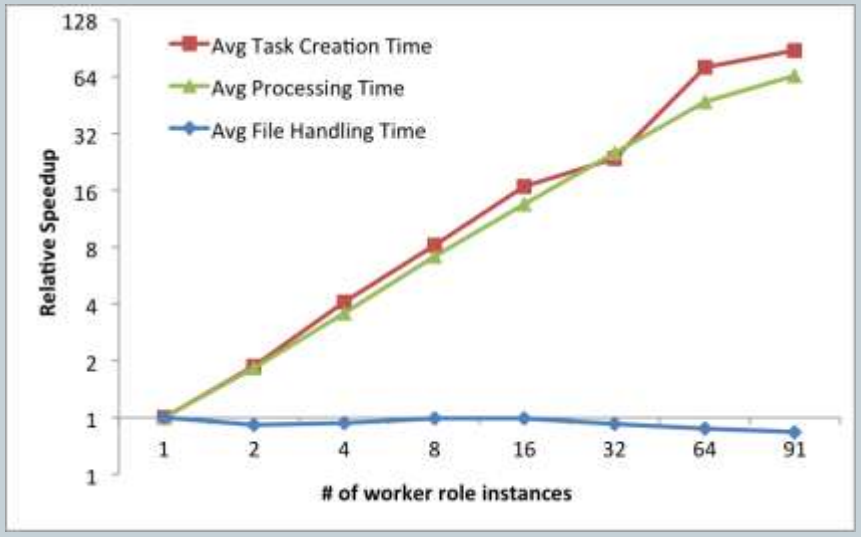
Centralized Dynamic Load Balancing

Average timings across all workers

- Task creation takes longer as workers are small virtual machines



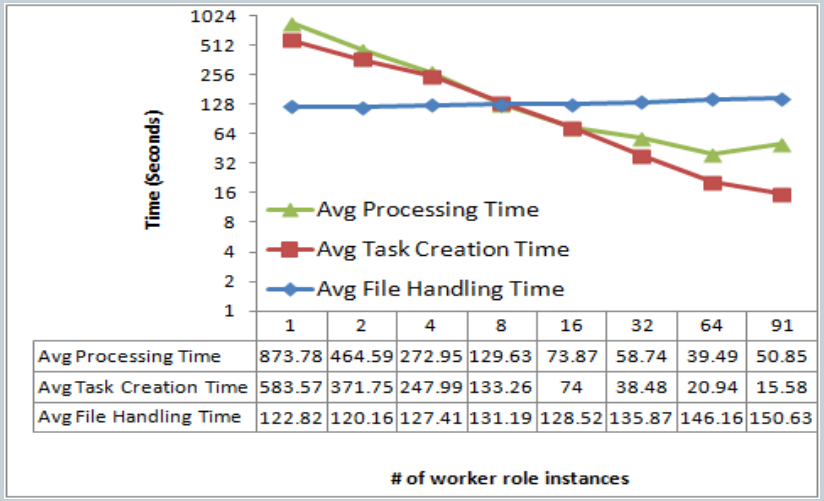
	1	2	4	8	16	32	64	91
Avg Processing Time	763.42	416.16	213.49	106.94	56.71	30.41	16.13	11.76
Avg Task Creation Time	458.95	245.71	112.43	55.99	27.35	19.51	6.43	5.23
Avg File Handling Time	128	140.48	136.92	128.95	129.27	138.49	145.93	152.39



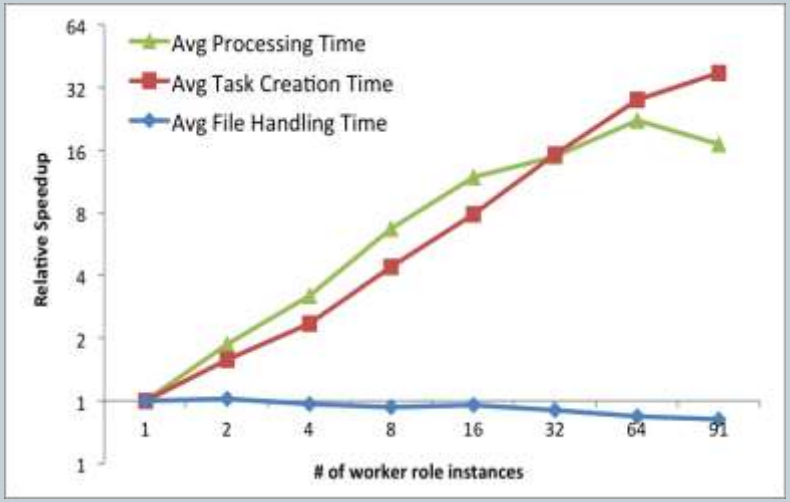
Distributed Static Load Balancing

Average timings across all workers

- Additional reading and writing (from/to Queue) overheads for large number of workers with small data
- Not enough work to process



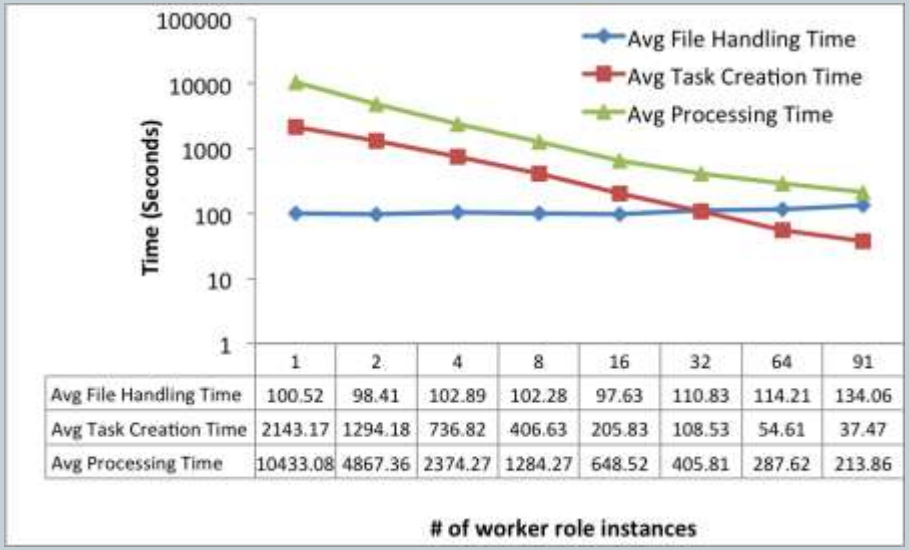
	1	2	4	8	16	32	64	91
Avg Processing Time	873.78	464.59	272.95	129.63	73.87	58.74	39.49	50.85
Avg Task Creation Time	583.57	371.75	247.99	133.26	74	38.48	20.94	15.58
Avg File Handling Time	122.82	120.16	127.41	131.19	128.52	135.87	146.16	150.63



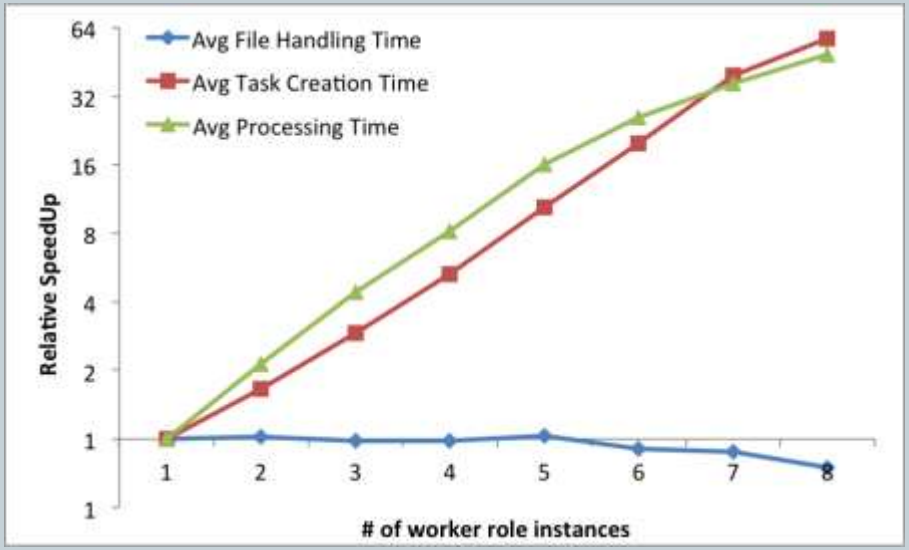
Distributed Dynamic Load Balancing

Average timings across all workers

- Larger data set



	1	2	4	8	16	32	64	91
Avg File Handling Time	100.52	98.41	102.89	102.28	97.63	110.83	114.21	134.06
Avg Task Creation Time	2143.17	1294.18	736.82	406.63	205.83	108.53	54.61	37.47
Avg Processing Time	10433.08	4867.36	2374.27	1284.27	648.52	405.81	287.62	213.86



Distributed Dynamic Load Balancing

Engineering Issues

- Azure-specific [Clouds 2012]
  - Table vs. Blob
  - Queues – FIFO Behavior
  - Serialization vs. GML Representation
  - Simulator vs. Cloud Environment
- Concurrency Control in .Net with Azure
  - Parallel constructs create lots of threads
  - Throttling the cloud storage
- Clipper library specific
  - Limited Supported Operations
  - Polygons with Holes can Cause Trouble

Current & Future Work

- Improved task creation using R-Tree
- Avoiding task storage in blobs
- Check pointing
- Parallel polygon overlay algorithm on multicores
- Shape file vs GML format, topologically distributed pages of GIS files

  -> Strongly scalable cloud system for skewed data

- MPI & Hadoop based implementation [IPDPS'12]
- Azure benchmark [IPDPS'12]
- GPU based parallel R tree construction

  -> GPU based GIS system

Questions? Contact: sprasad@gsu.edu