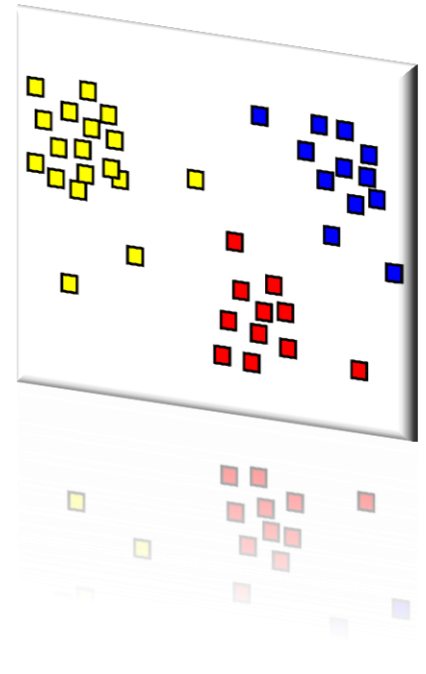


# CloudClustering

Toward a scalable machine  
learning toolkit for Windows Azure



Ankur Dave

XCG Intern

Microsoft Research

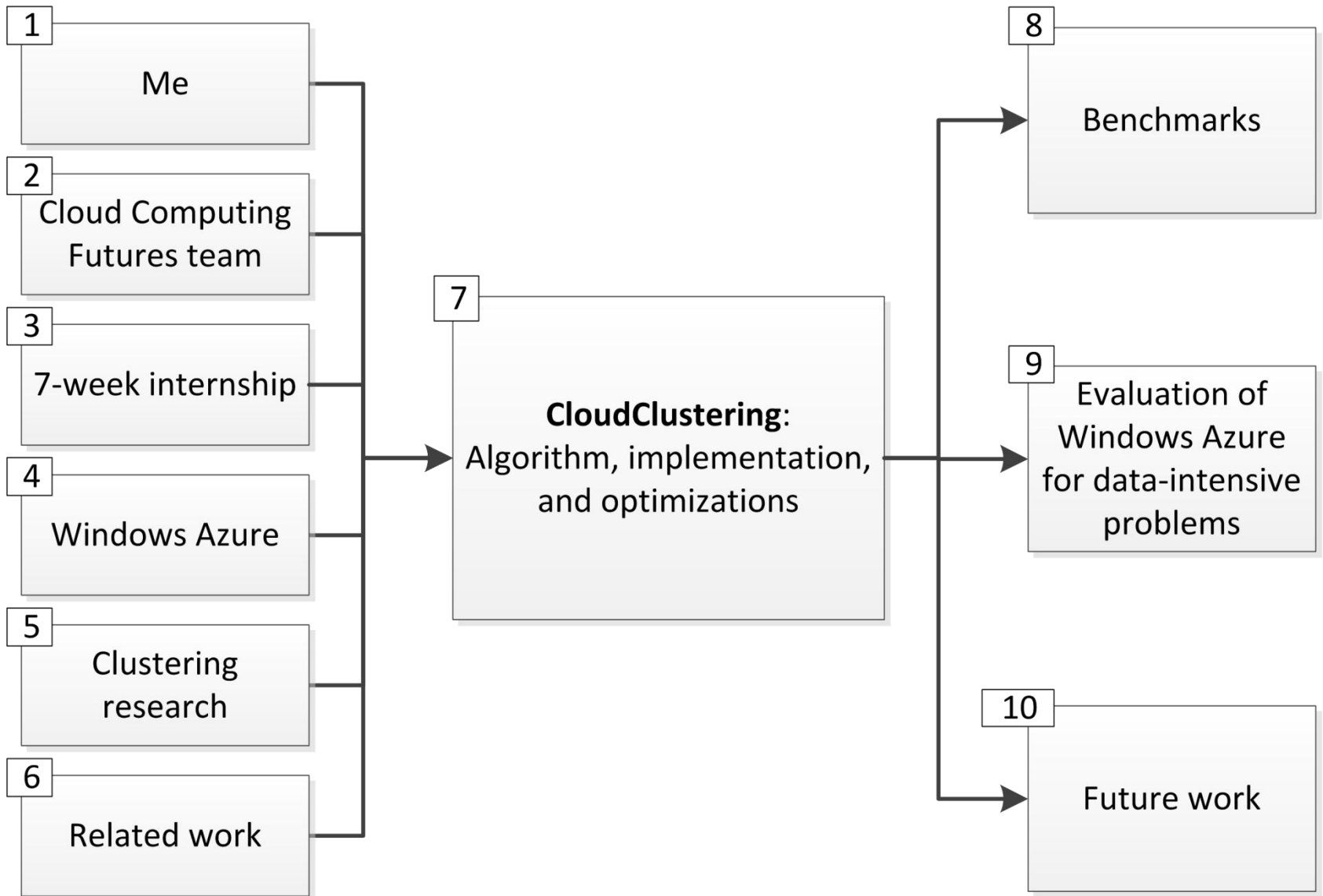
Mentors:

Roger Barga

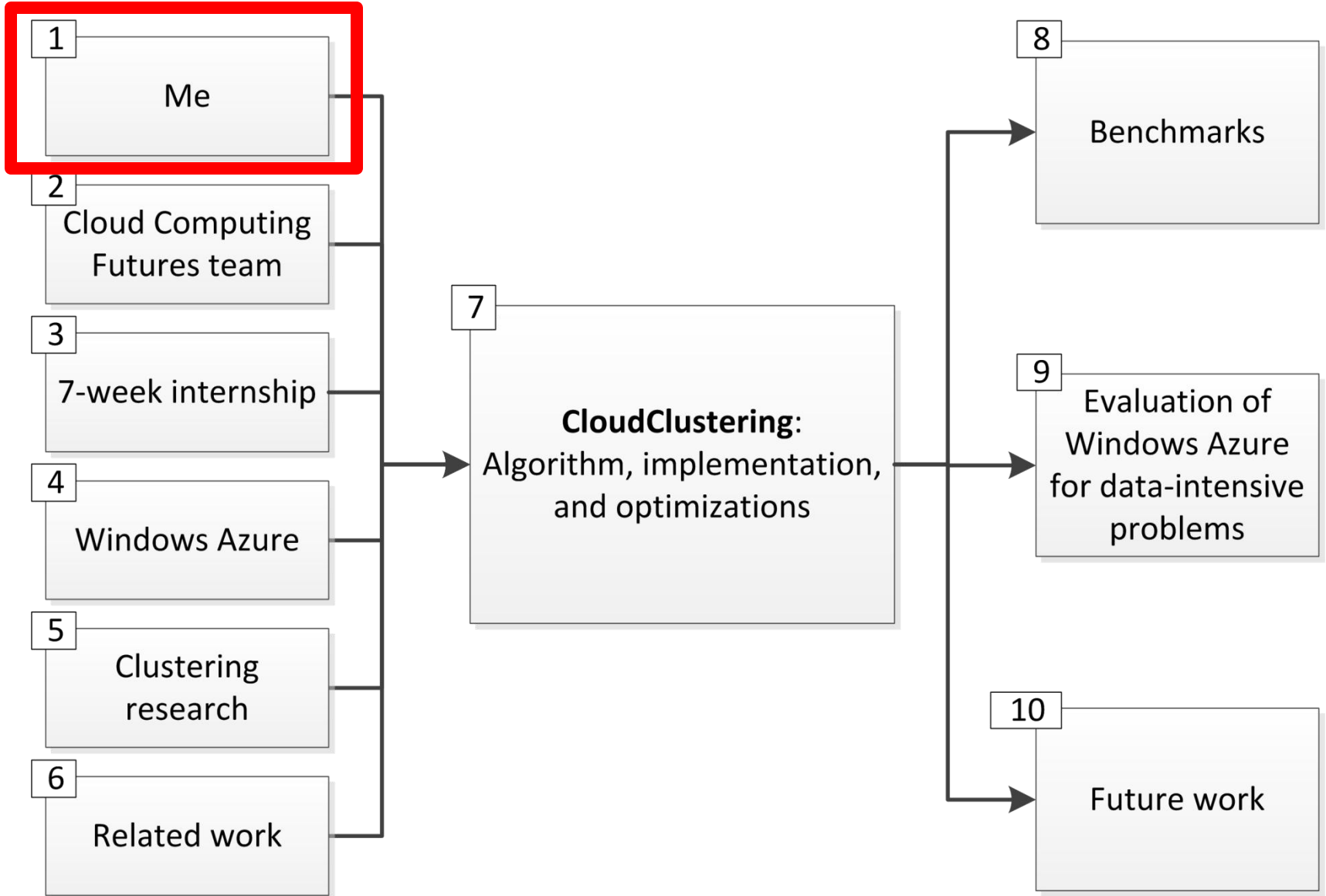
Wei Lu

August 12, 2010

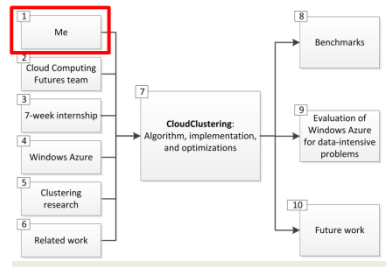
# Agenda



# Agenda



# About me



## EDUCATION

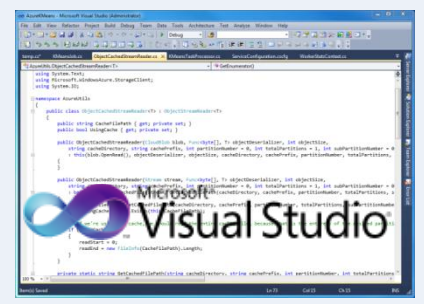
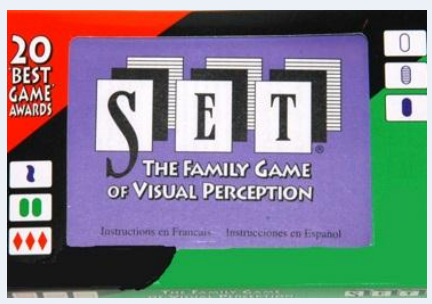


UNIVERSITY OF CALIFORNIA, BERKELEY

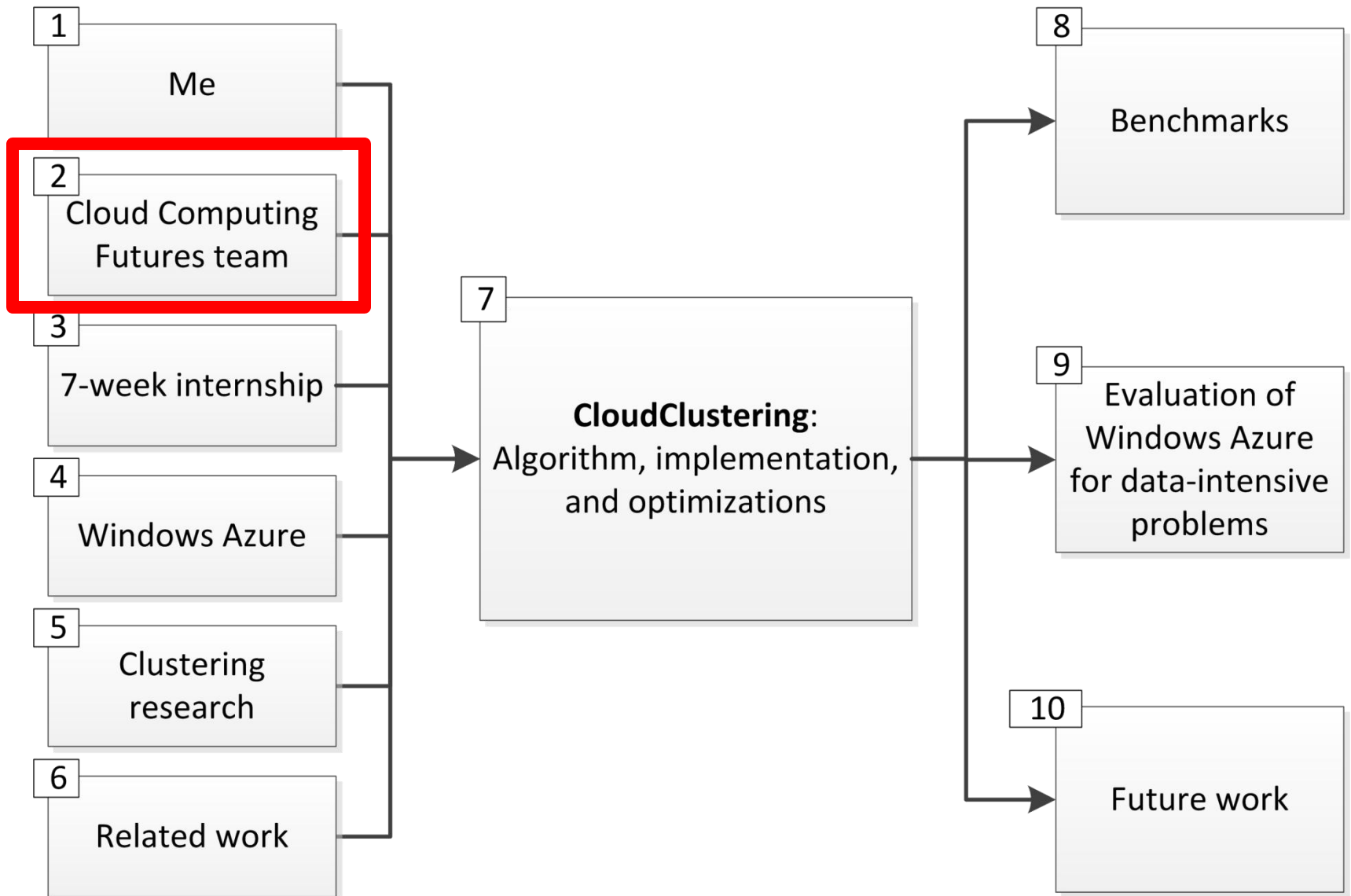
## WORK



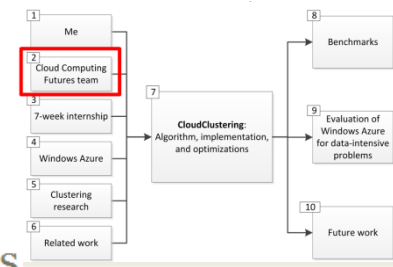
## HOBBIES



# Agenda



# My team



## Cloud Computing Futures

To create novel data center solutions, designs must be based on comprehensive optimization of all attributes, rather than gradually accruing incremental changes based on current technologies and best practices. The Cloud Computing Futures team is tasked to invent on a large scale. Our goal is to reduce data center costs by four-fold or greater, including power consumption, while accelerating deployment and increasing adaptability and resilience to failures.

## Azure Research Engagement



The Azure Research Engagement project aims to change the paradigm for scholarly and scientific research by extending the power of the computer into the cloud. We build the components of cloud technology and work with researchers in the field on projects that push the frontier of client and cloud computing.



Dan Reed



Dennis Gannon



Jaliya Ekanayake



Jared Jackson



Nelson Araujo

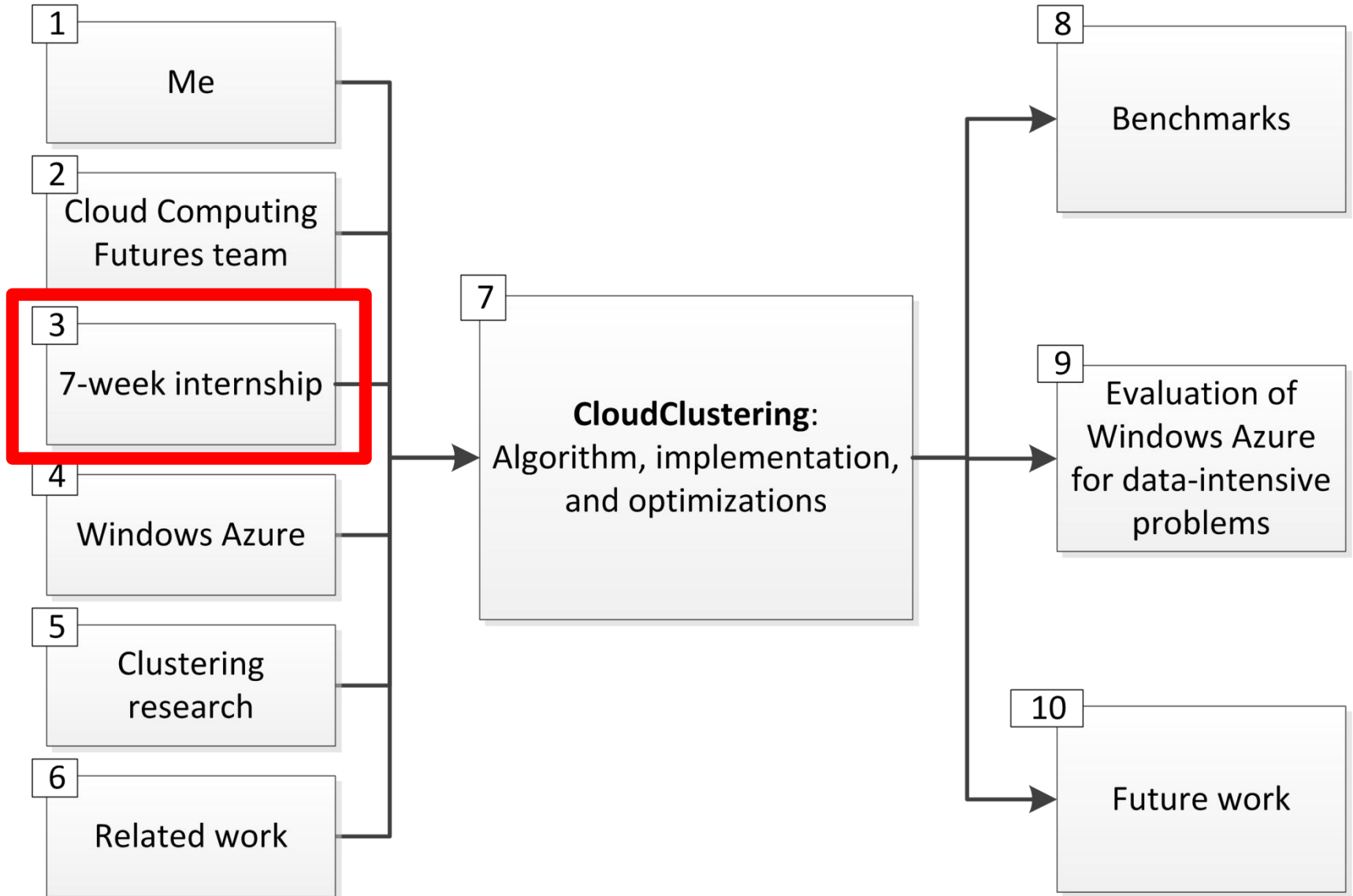


Roger Barga

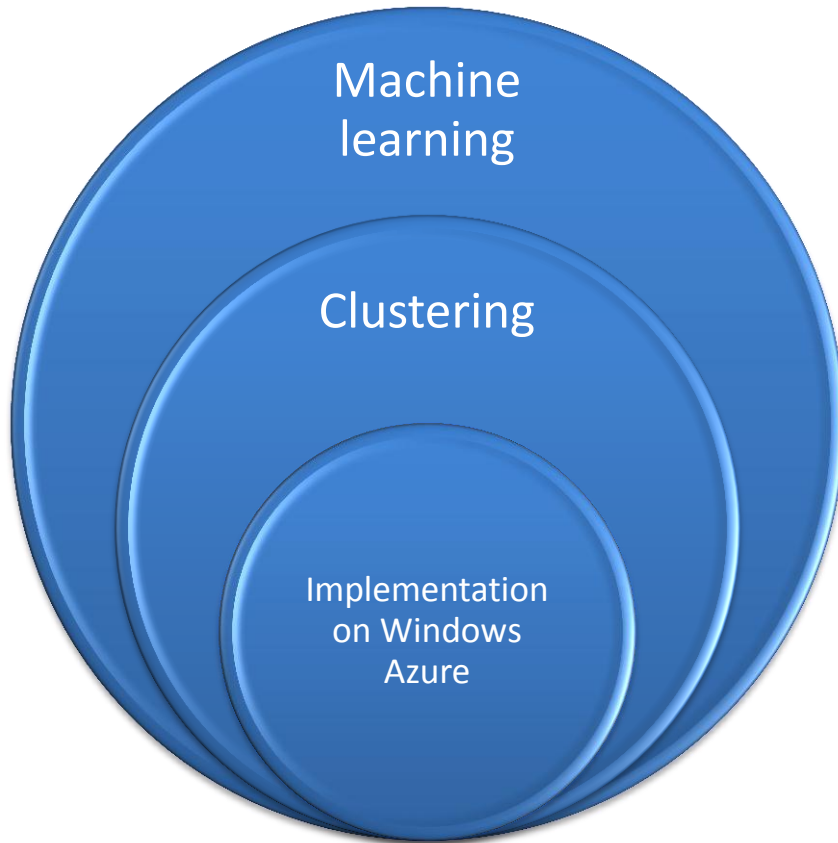
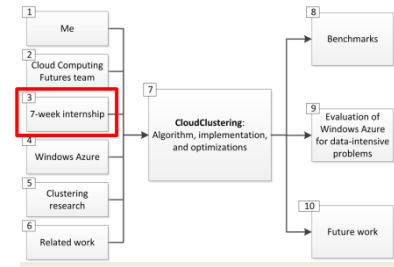


Wei Lu

# Agenda



# Internship: Domain



## Goals:

- Build a scalable clustering algorithm on Azure
- Explore clustering and the cloud by reading papers
- Meet a variety of researchers at MSR

**Week 1:** Ramp-up; architecture planning

**Week 2:** Building CloudClustering base impl.

**Week 3:** Building CloudClustering base impl.

**Week 4:** Multicore parallelism with PLINQ

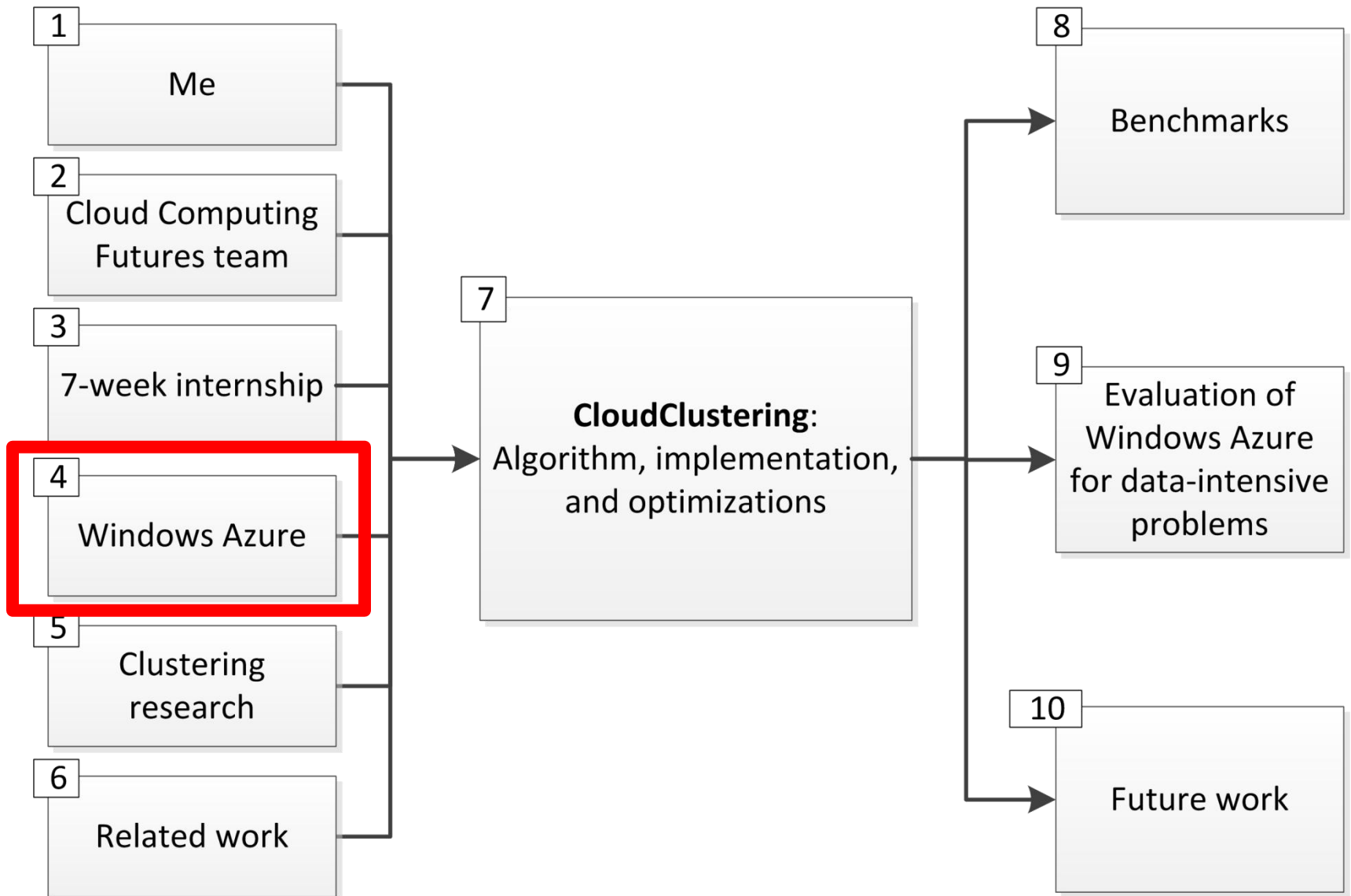
**Week 5:** Performance testing on Azure fabric

**Week 6:** Multicore w/threads; data affinity

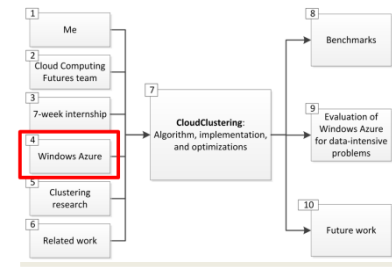
**Week 7:** Presentation and report-out



# Agenda



# Windows Azure



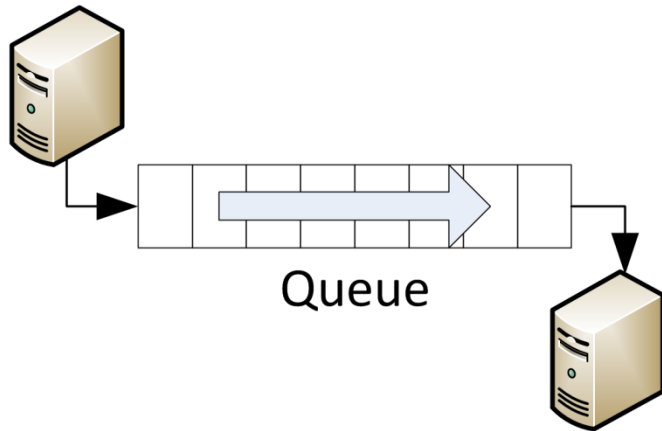
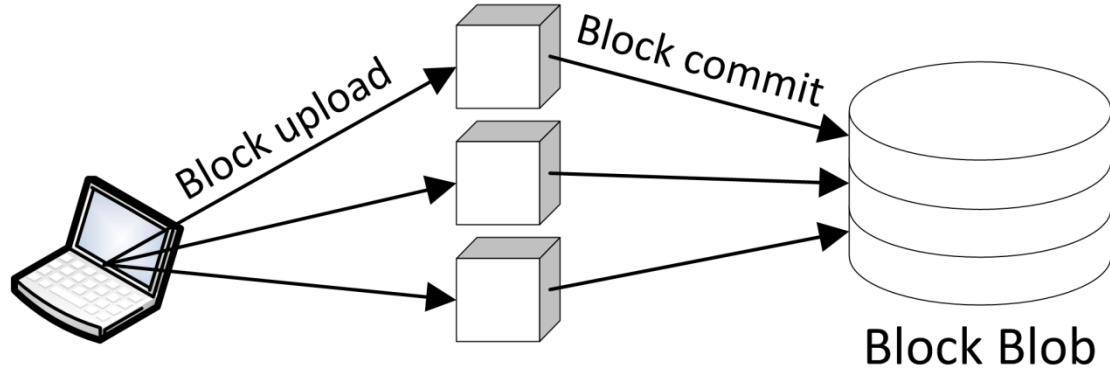
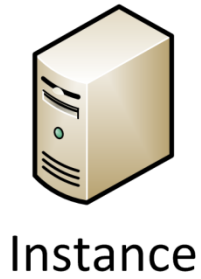
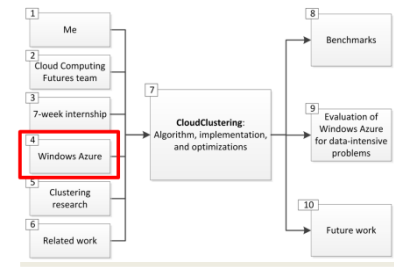
Exploring



# Windows® Azure™

for data-intensive research

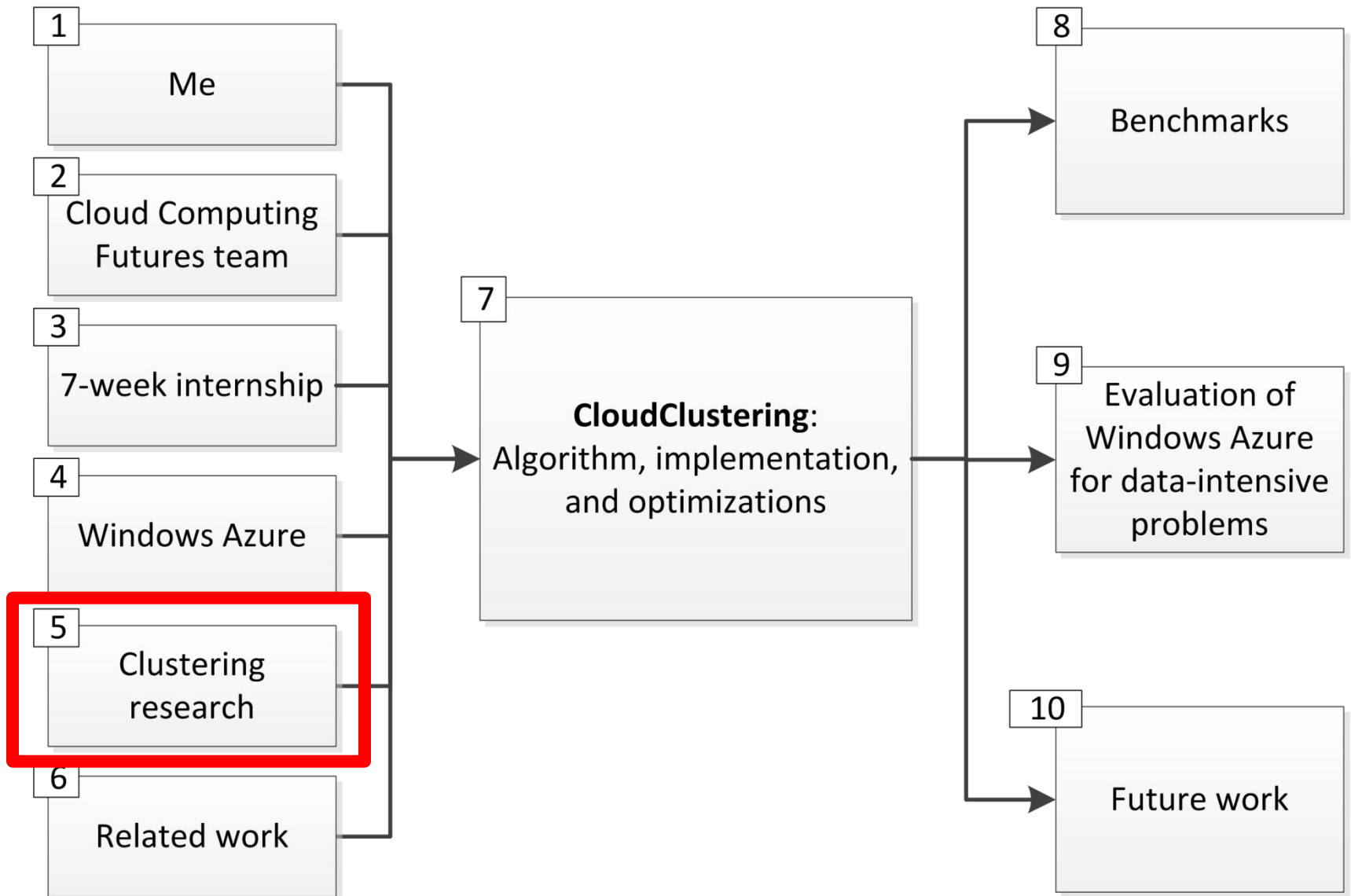
# Windows Azure



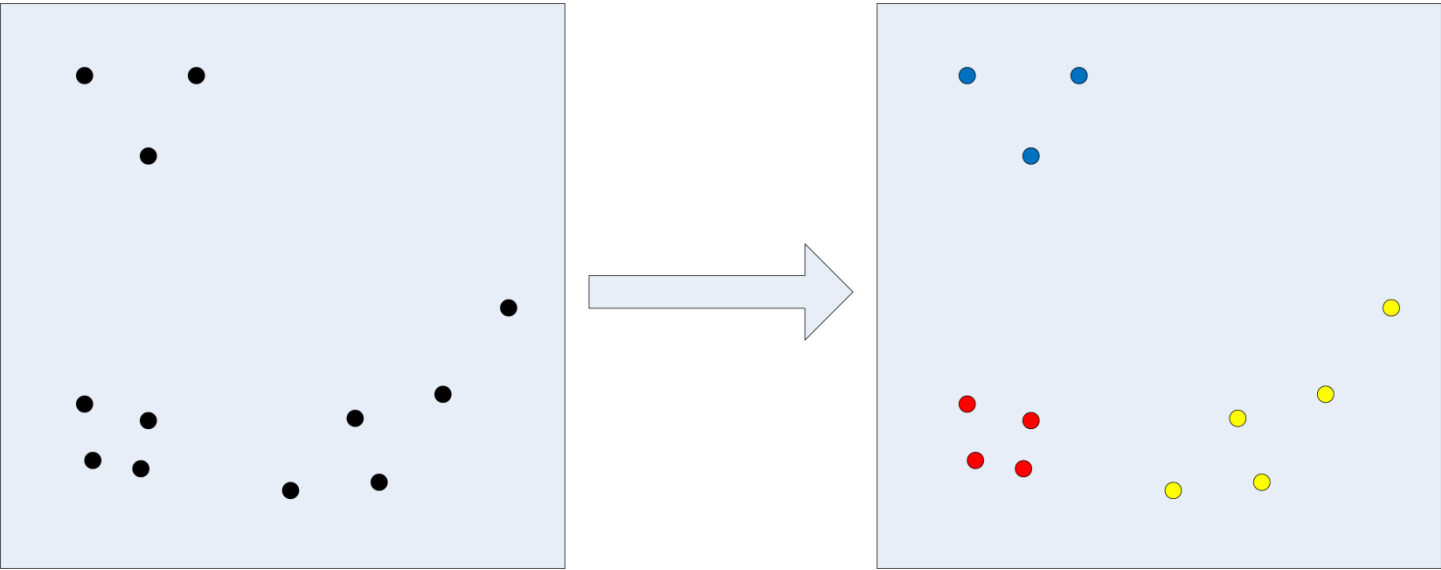
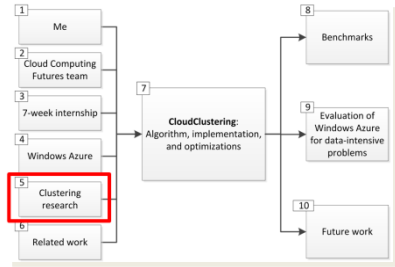
PartitionKey	RowKey	Col1	Col2

Table

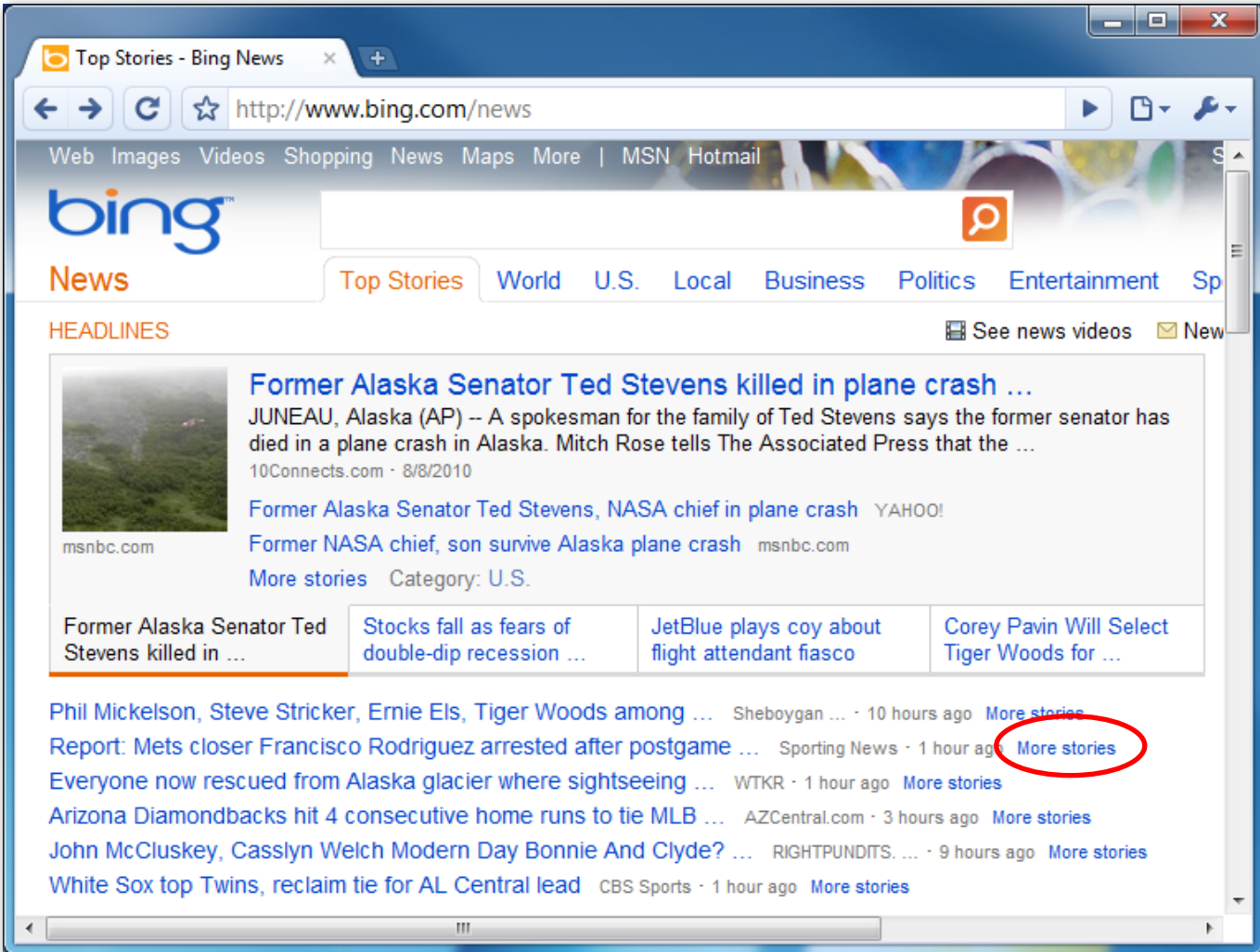
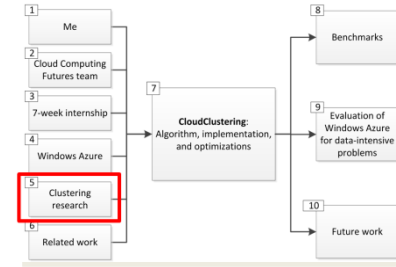
# Agenda



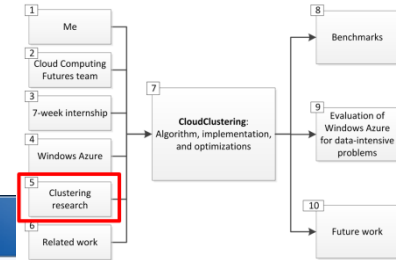
# Clustering



# Clustering



# Clustering



SEE ALSO  
Today's top stories  
Related blogs  
Related tweets

PUBLISH TIME  
Past hour  
Past 24 hours  
Past 7 days

CATEGORY  
Sports

LOCATION  
Washington sources  
More >

SEARCH HISTORY  
Search more to see your history

See all  
Clear all · Turn off

Sort by: Best match | Most recent

News alerts RSS

1-10 of 9,870 results

STORY AT A GLANCE

**Tiger Woods has leading role in golf's soap opera**  
Burlington County Times - 6 hours ago

Fans are coming out in droves to catch a glimpse of their favorite players at the PGA championship. WISN.com - 17 hours ago Read article

Golfer Phil Mickelson revealed he has been diagnosed with a severe form of arthritis that attacks the body's immune system and in his case left him unable to even get out of bed. Bangkok Post - 1 day ago Read article

Washington Post

SHARE Facebook Twitter Messenger Email

Usual suspects are suspect at the PGA Championship

Whistling and Straits barely rank among the popular buzzwords this week. Getting far more play: "mosquitoes," "arthritis," "Jim Gray" and "wide open." The "wide open" part reflects the winners of the last ...  
Los Angeles Times - 11 minutes ago

2010 PGA Championship: Ranking Phil Mickelson and Tiger Woods' Chances

The 92 nd PGA Championship is upon us. The 2010 edition sees a return to Whistling Straits for the first time since 2004, when Vijay Singh took home the Wanamaker trophy. And there is no shortage of story lines this year ...  
Bleacherreport.com - 1 day ago

Phil Mickelson, Steve Stricker, Ernie Els, Tiger Woods among favorites to win PGA...

Why he can win: Desperately wants to become No. 1 in the world and can do so this week. Long hitter who can overpower one of the longest major championship courses and has the short game to match. Played well ...  
Sheboygan Press.com - 10 hours ago

Tiger Woods, Phil Mickelson have much at stake at PGA Championship for different reasons

Articles Tiger Woods struggles continue at Bridgestone Invitational with 75; likely to lose No. 1 ranking Phil Mickelson enters final events of season with ...

LATEST STORIES

Drama unfolds at Whistling Straits on eve of PGA Championship  
Temple Daily Telegram - 3 minutes ago

PGA is Players Gone Amok  
Evansville Courier-Press - 4 minutes ago

It's Open season for 1st-timers  
Houston Chronicle - 4 minutes ago

More updates

STORY DEVELOPMENT

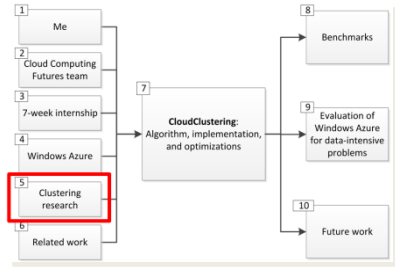
News Volume

12AM 8/7 3PM 8/8 6AM 8/10 9PM 8/11

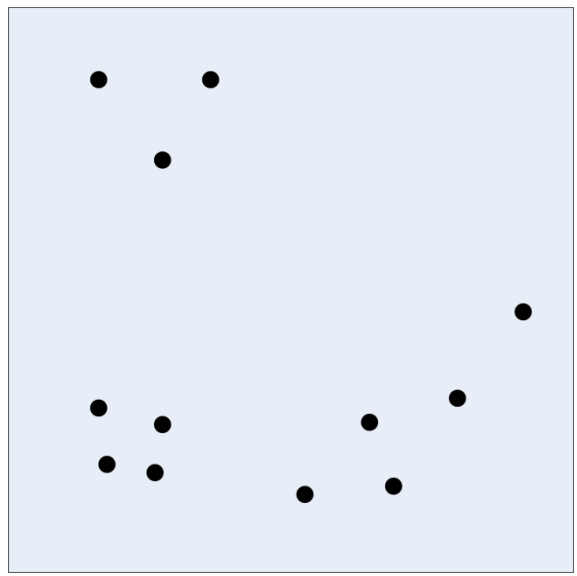
- 1 Tiger Woods has leading role in golf's soap opera  
The Guardian - 1 hour ago
- 2 Ranking the PGA Championship field  
ESPN.com - 11 hours ago
- 3 What players are saying about Woods at Whistling Straits  
The Guardian - 1 day ago
- 4 Firestone behind him, Woods is upbeat about game  
San Jose Mercury News - 1 day ago

See today's top stories

# Clustering



## *k*-means



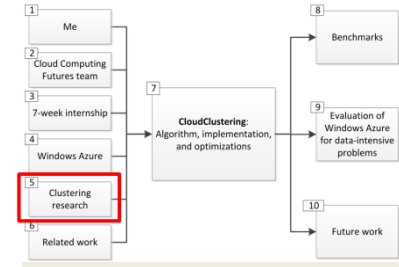
*k* = 3

- Minimizes the *within-cluster sum of squares*:

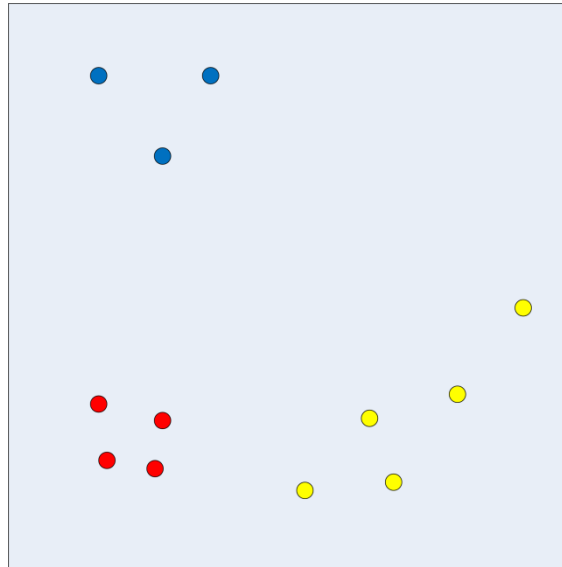
$$\sum_{i=1}^k \underbrace{\sum_{p \in C_i}}_{\text{For each cluster}} \underbrace{\|p - \mu_i\|^2}_{\text{Distance from point to centroid}}$$



# Clustering

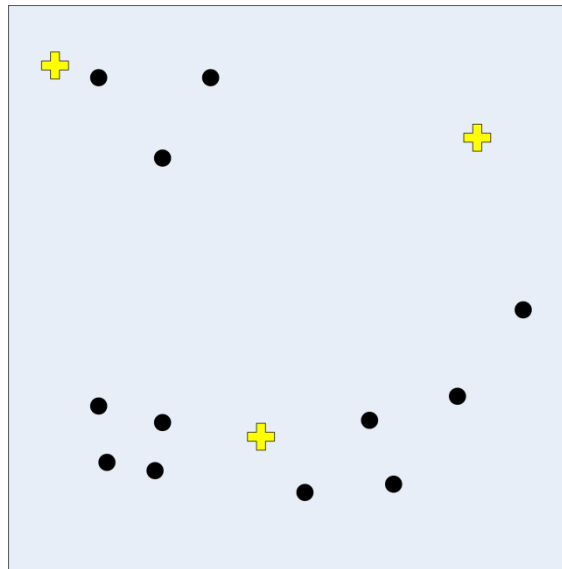


## Target clustering

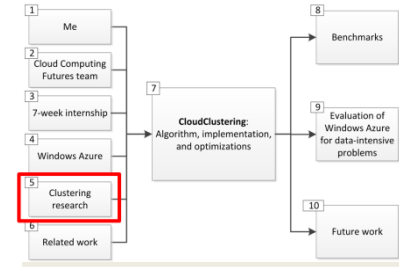


# Clustering

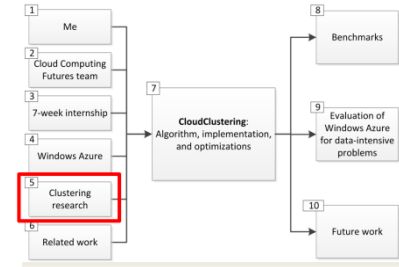
## 1. Initialization



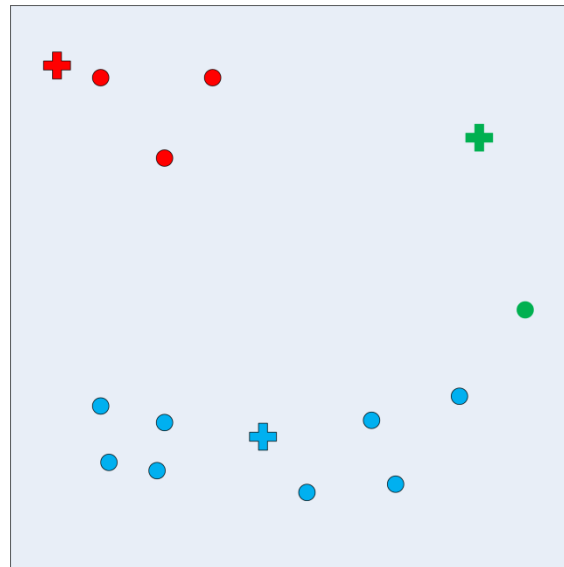
$$k = 3$$



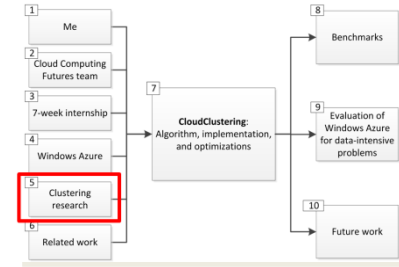
# Clustering



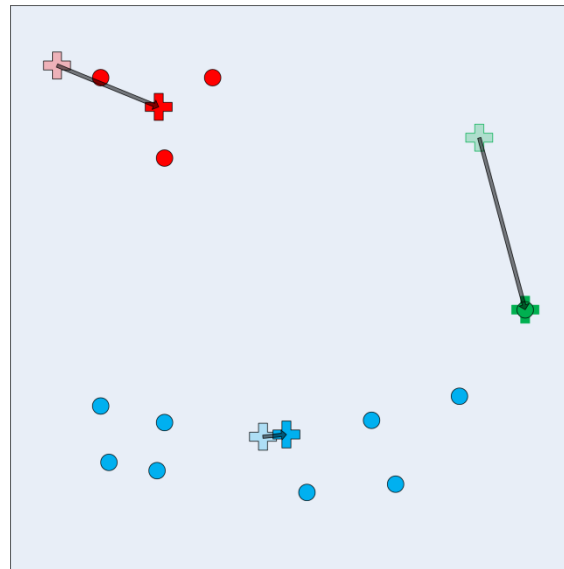
## 2. Assign Points to Centroids #1



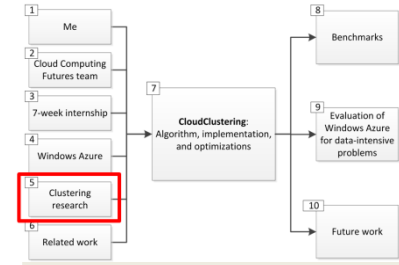
# Clustering



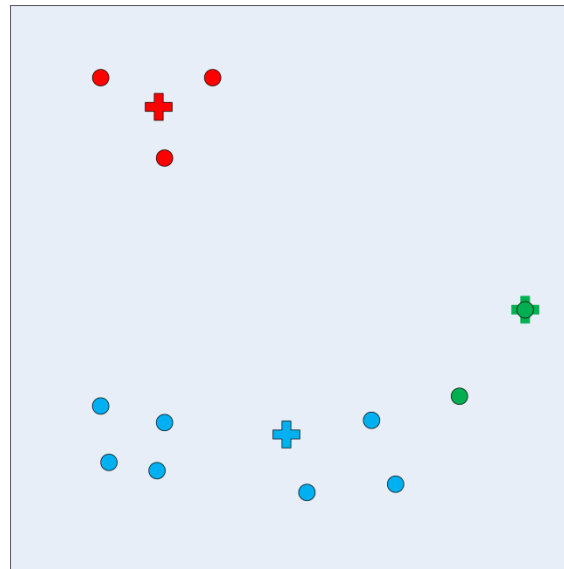
## 3. Recalculate Centroids #1



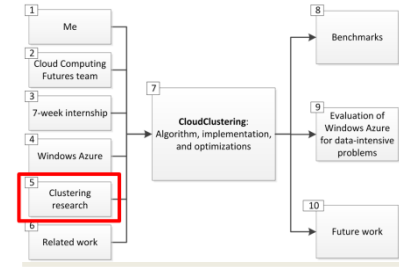
# Clustering



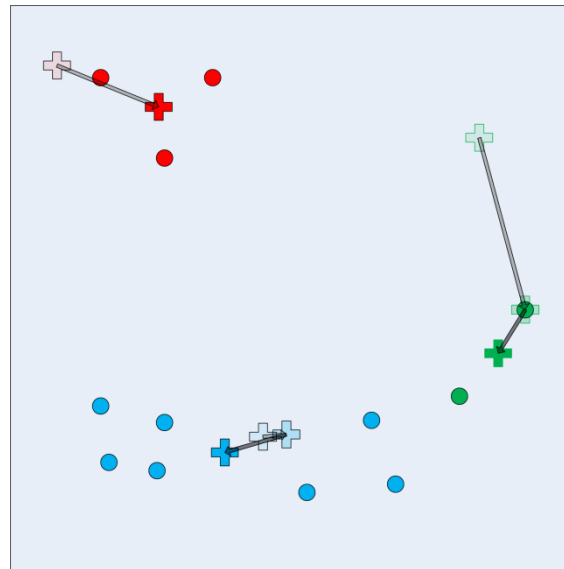
## 4. Assign Points to Centroids #2



# Clustering

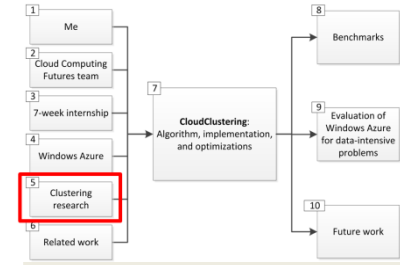
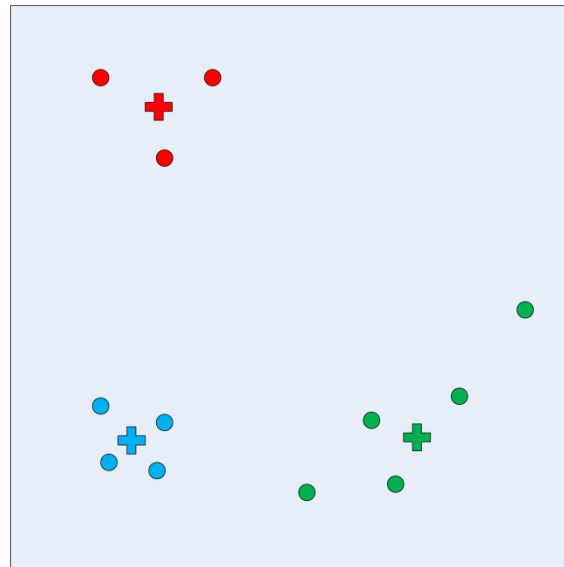


## 5. Recalculate Centroids #2

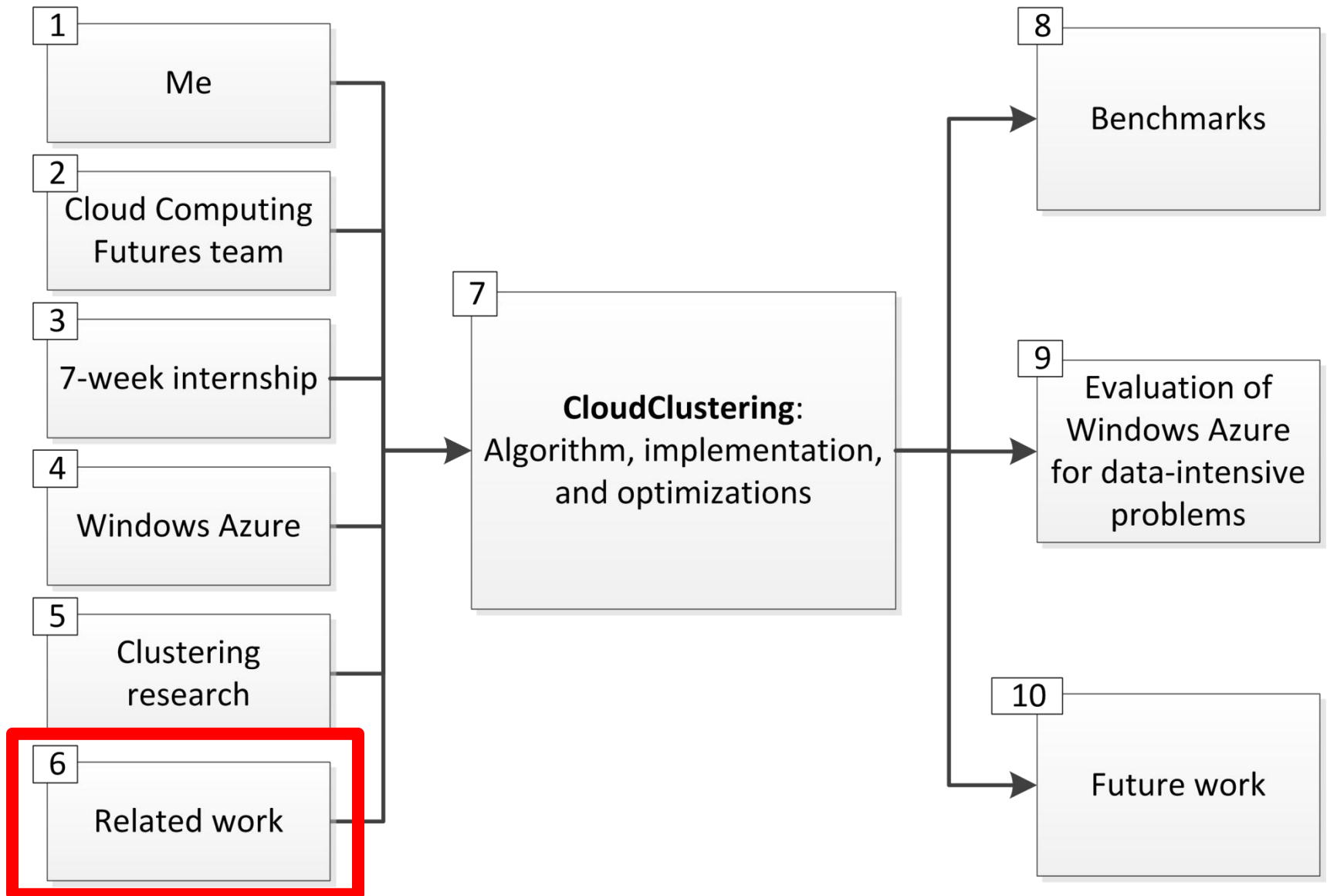


# Clustering

## Stopping Condition

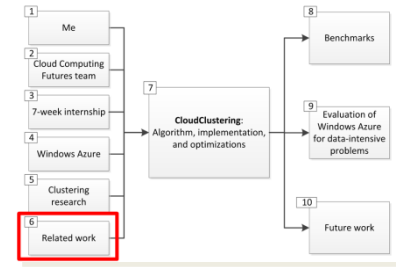


# Agenda



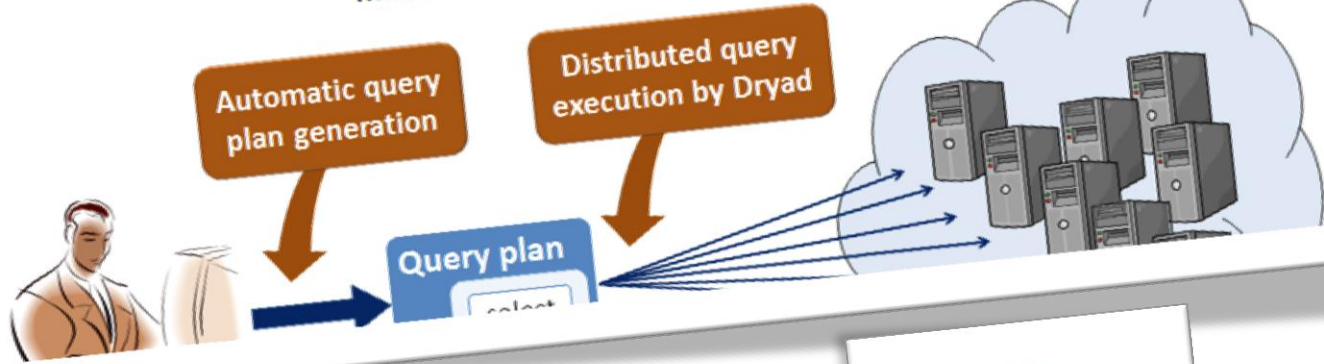


# Related Work

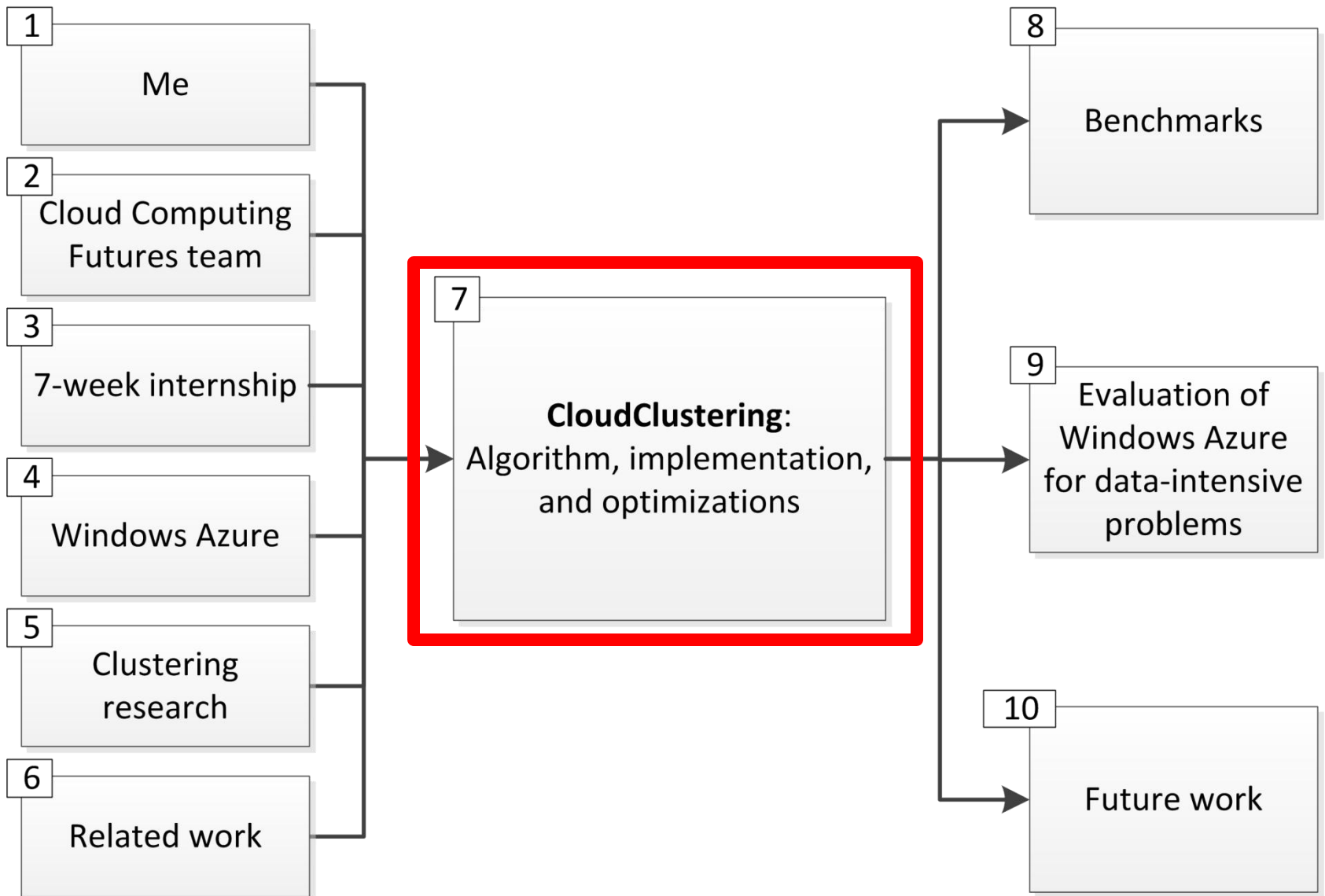


## DryadLINQ: Making Large-Scale Distributed Computing Simple

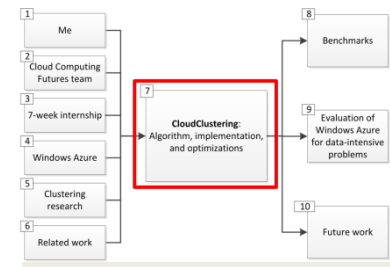
Microsoft Research Silicon Valley



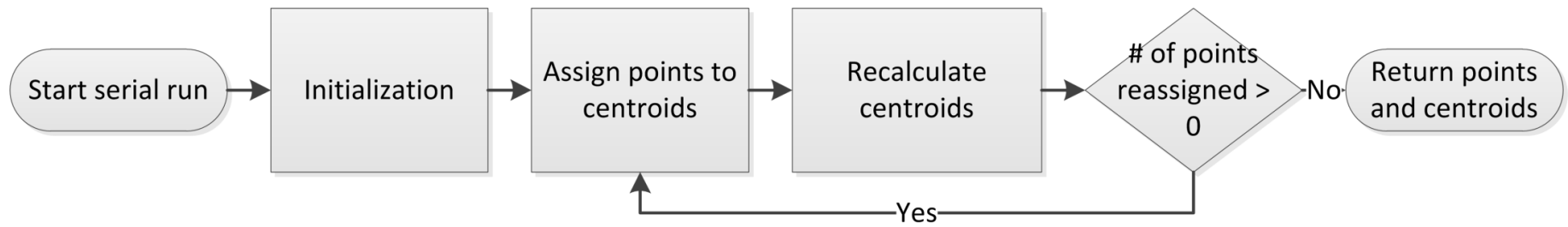
# Agenda



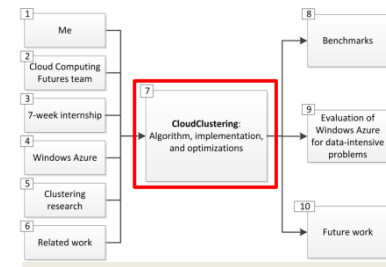
# CloudClustering: Algorithm



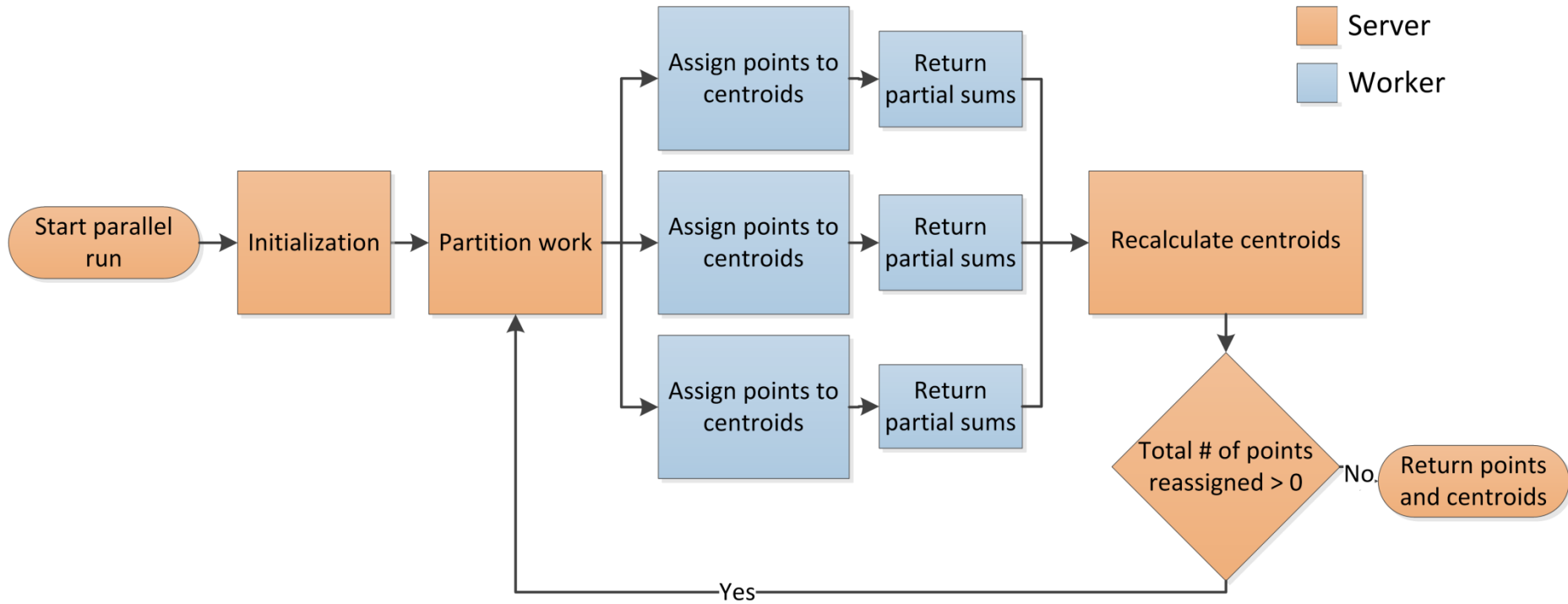
## Conventional (Serial) *k*-means



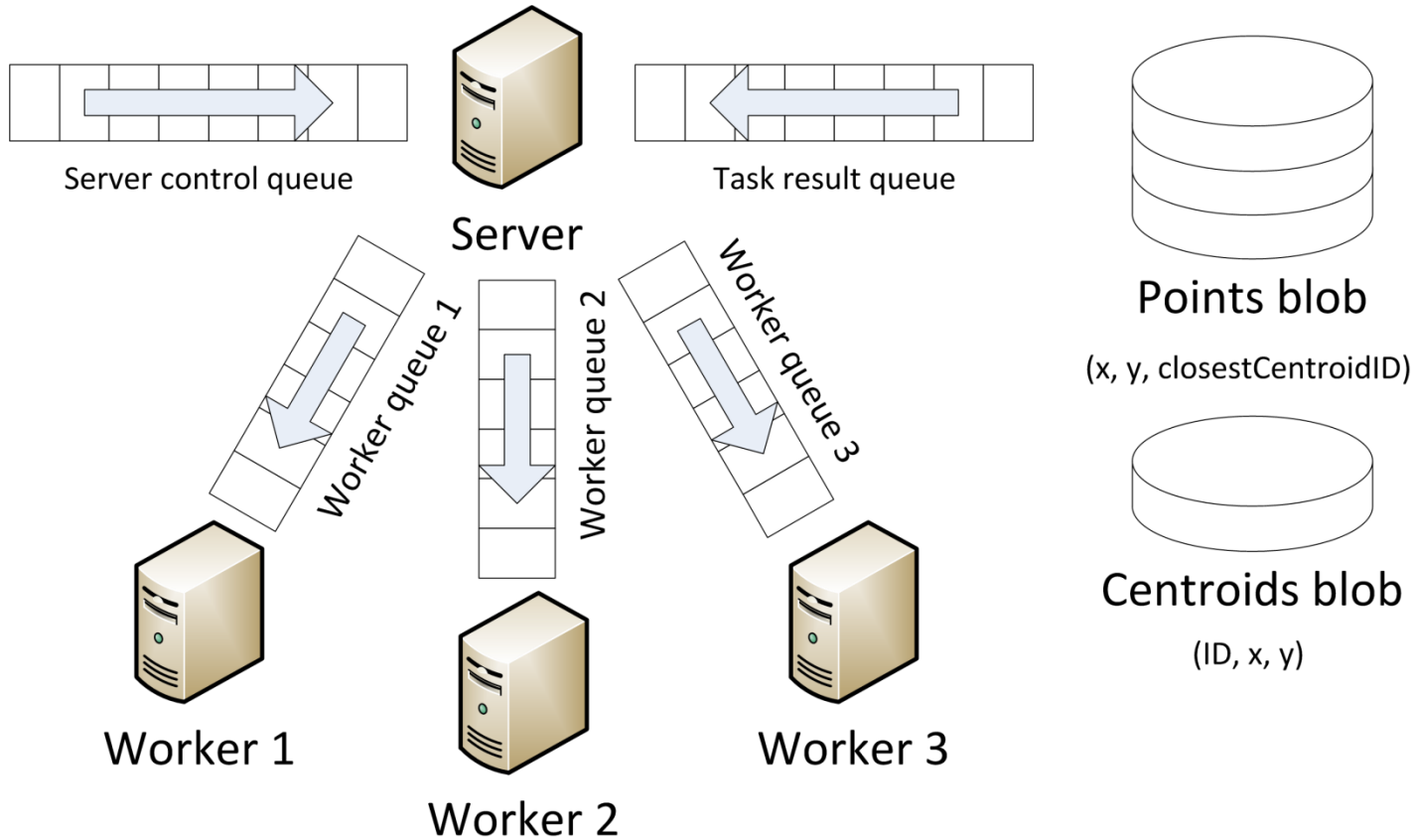
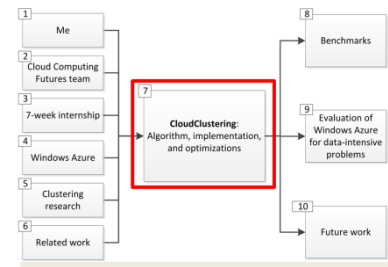
# CloudClustering: Algorithm



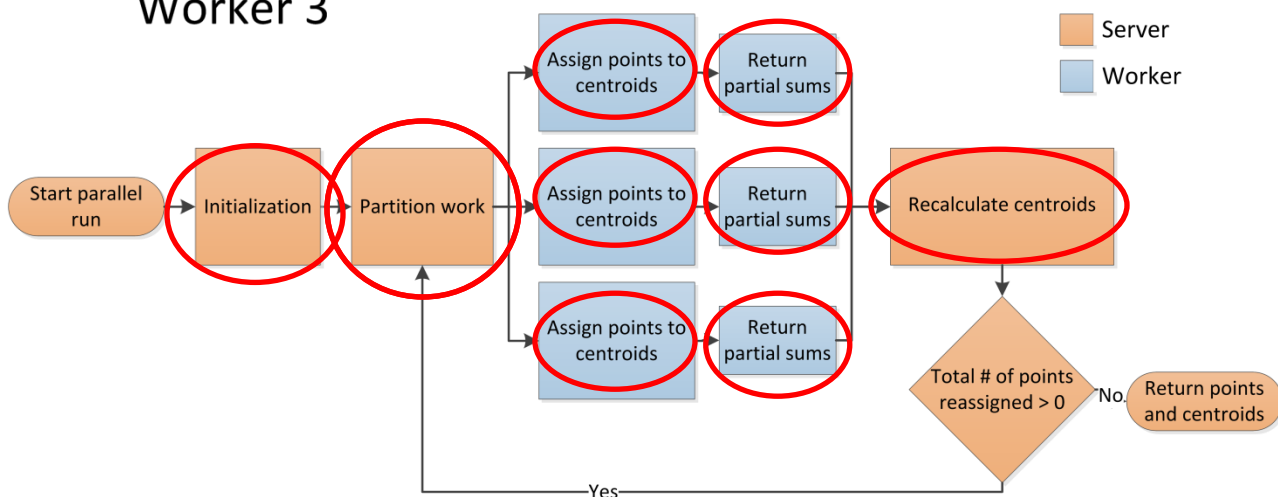
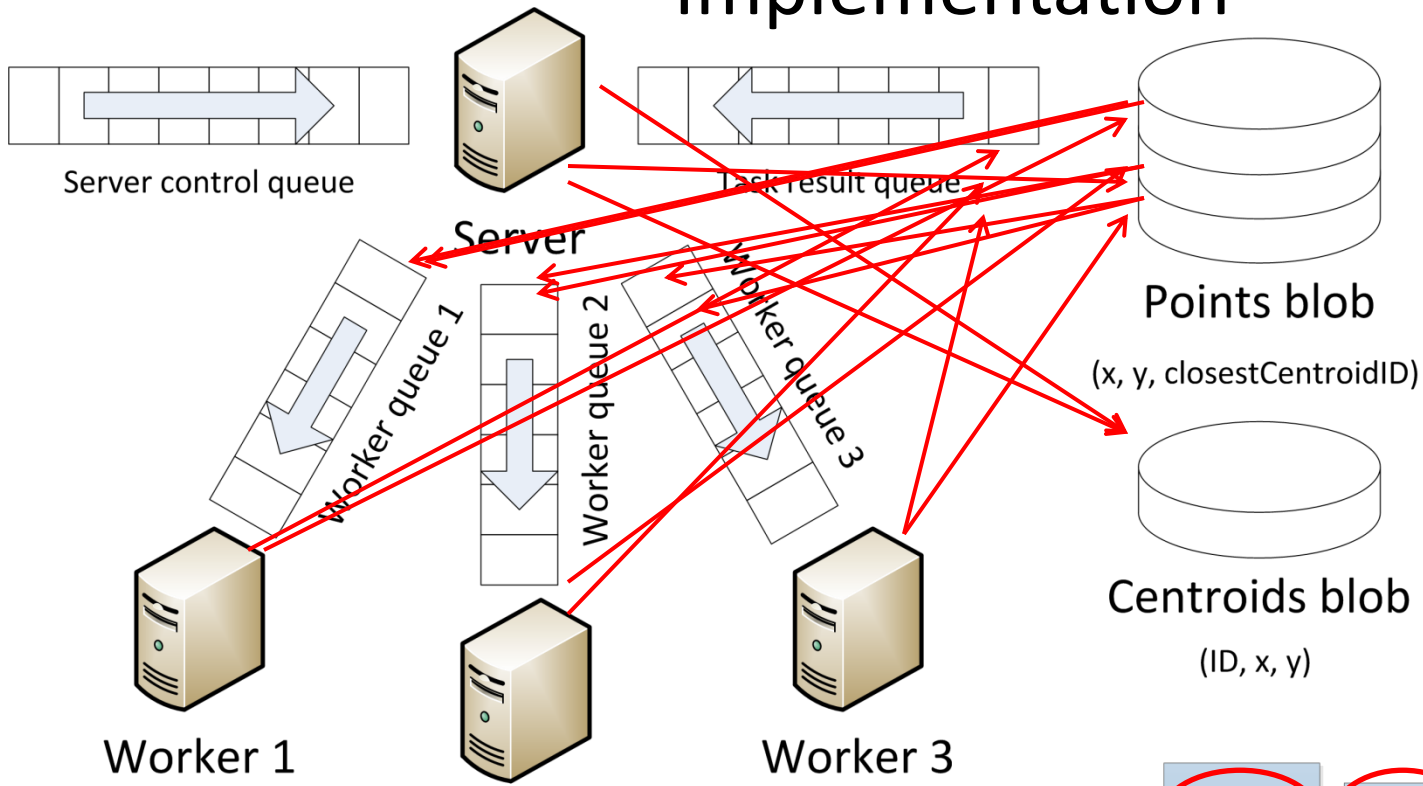
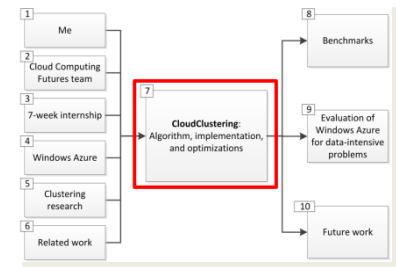
## Parallel *k*-means



# CloudClustering: Implementation

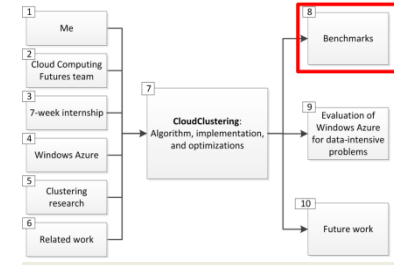


# CloudClustering: Implementation



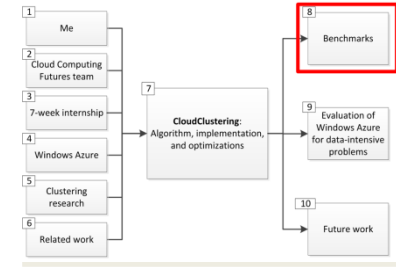
Server  
 Worker

# CloudClustering: Leveraging the cloud

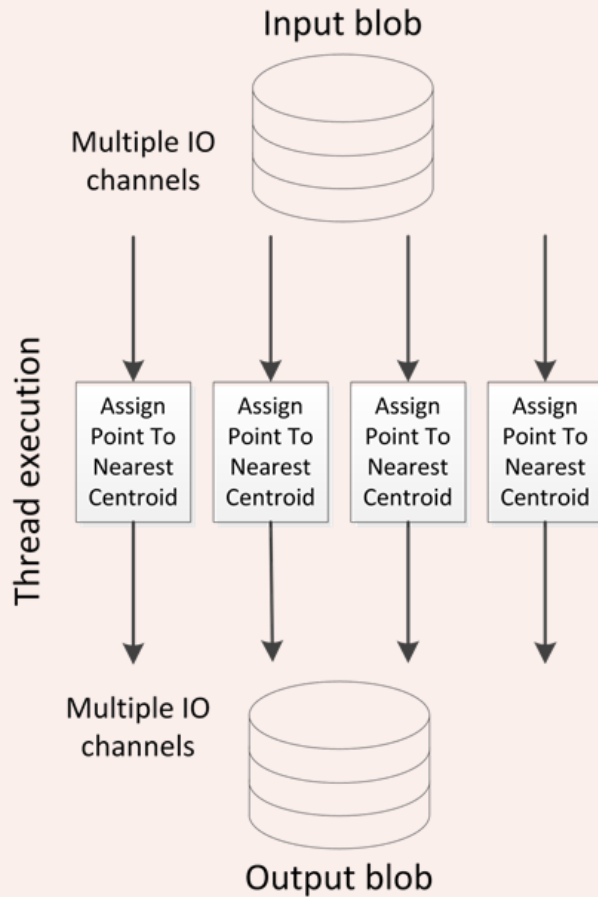


- **Multicore Parallelism**
- **Data Affinity**
- **Efficient Blob Concatenation**
- **Dynamic Scalability**

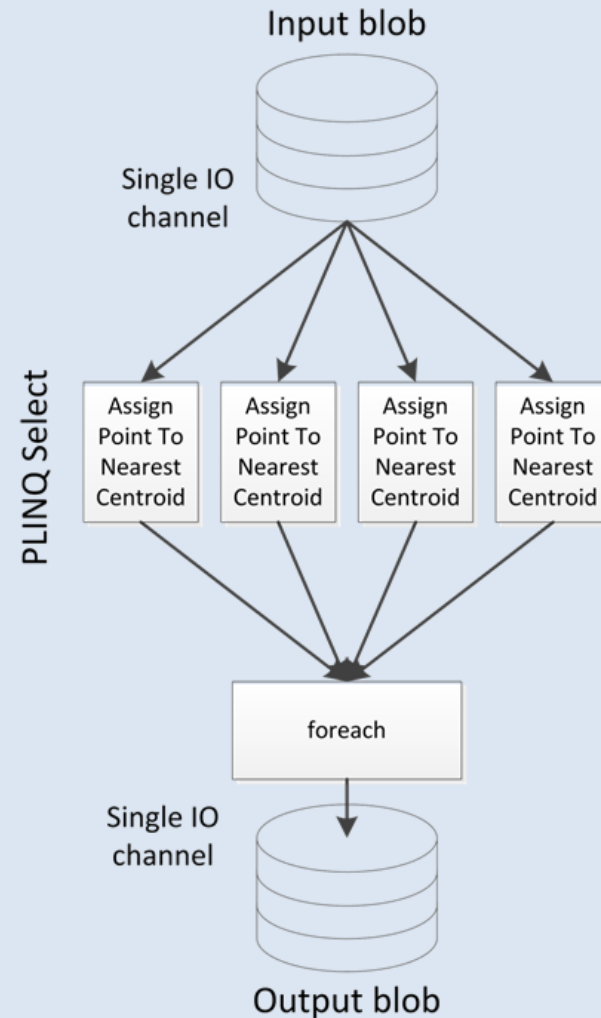
# CloudClustering: Multicore Parallelism



## Thread-based data flow



## PLINQ data flow





# Thread-based

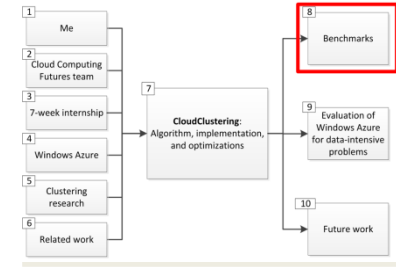
```
private void ProcessPoints()
{
    CloudBlobBlob pointsBlob = AzureHelper.GetBlob(task.Points);
    // Do the mapping and write the new blob
    int numThreads = Environment.ProcessorCount;
    PointsProcessedData[,] pointsSumsPerCentroidPerThread = new PointsProcessedData[numThreads, task.K];
    int[] pointsChangedPerThread = new int[numThreads];
    string[][] blockIDsPerThread = new string[numThreads][];
    System.Threading.Tasks.Parallel.For(0, numThreads, threadID =>
    {
        using (ObjectCachedStreamReader<ClusterPoint> stream = new ObjectCachedStreamReader<ClusterPoint>(pointsBlob,
ClusterPoint.FromByteArray, ClusterPoint.Size, AzureHelper.GetLocalResourceRootPath("cache"), task.JobID.ToString(),
task.PartitionNumber, task.M, subPartitionNumber: threadID, subTotalPartitions: numThreads))
        {
            // Process the points
            ObjectBlockWriter<ClusterPoint> writeStream = new ObjectBlockWriter<ClusterPoint>(pointsBlob, point =>
point.ToByteArray(), ClusterPoint.Size);
            foreach (var point in stream)
            {
                // Assign the point to the nearest centroid
                Guid oldCentroidID = point.CentroidID;
                int closestCentroidIndex = centroids.MinIndex(centroid => Point.Distance(point, centroid));
                Guid newCentroidID = point.CentroidID = centroids[closestCentroidIndex].ID;
                // Write the updated point to the writeStream
                writeStream.Write(point);
                // Update the number of points changed
                if (oldCentroidID != newCentroidID)
                {
                    pointsChangedPerThread[threadID]++;
                }
                // Update the point sums
                if (pointSumsPerCentroidPerThread[threadID, closestCentroidIndex] == null)
                {
                    pointSumsPerCentroidPerThread[threadID, closestCentroidIndex] = new PointsProcessedData();
                }
                pointSumsPerCentroidPerThread[threadID, closestCentroidIndex].PartialPointSum += point;
                pointSumsPerCentroidPerThread[threadID, closestCentroidIndex].NumPointsProcessed++;
            }
            // Collect the block IDs from writeStream
            writeStream.FlushBlock();
            blockIDsPerThread[threadID] = writeStream.BlockList.ToArray();
        }
    });
    // Combine the per-thread block lists and write the full block list to a blob. Then include that as part of TaskResult
    List<string> blockIDs = new List<string>();
    foreach (string[] blockIDsFromThread in blockIDsPerThread)
    {
        blockIDs.AddRange(blockIDsFromThread);
    }
    CloudBlob blockIDsBlob = AzureHelper.CreateBlob(task.JobID.ToString(), Guid.NewGuid().ToString());
    using (Stream stream = blockIDsBlob.OpenWrite())
    {
        BinaryFormatter bf = new BinaryFormatter();
        bf.Serialize(stream, blockIDs);
    }
    TaskResult.PointsBlockListBlob = blockIDsBlob.Uri;
    // Total up the per-thread pointSumsPerCentroid
    TaskResult.PointsProcessedDataByCentroid = new Dictionary<Guid, PointsProcessedData>();
    for (int i = 0; i < task.K; ++i)
    {
        Guid centroidID = centroids[i].ID;
        TaskResult.PointsProcessedDataByCentroid[centroidID] = new PointsProcessedData();
        for (int j = 0; j < numThreads; ++j)
        {
            if (pointSumsPerCentroidPerThread[j, i] != null)
            {
                TaskResult.PointsProcessedDataByCentroid[centroidID].PartialPointSum += pointSumsPerCentroidPerThread[j,
i].PartialPointSum;
                TaskResult.PointsProcessedDataByCentroid[centroidID].NumPointsProcessed += pointSumsPerCentroidPerThread[j,
i].NumPointsProcessed;
            }
        }
    }
    // Total up the per-thread numPointsChanged
    TaskResult.NumPointsChanged = 0;
    foreach (int threadPointsChanged in pointsChangedPerThread)
    {
        TaskResult.NumPointsChanged += threadPointsChanged;
    }
}
```

# PLINQ

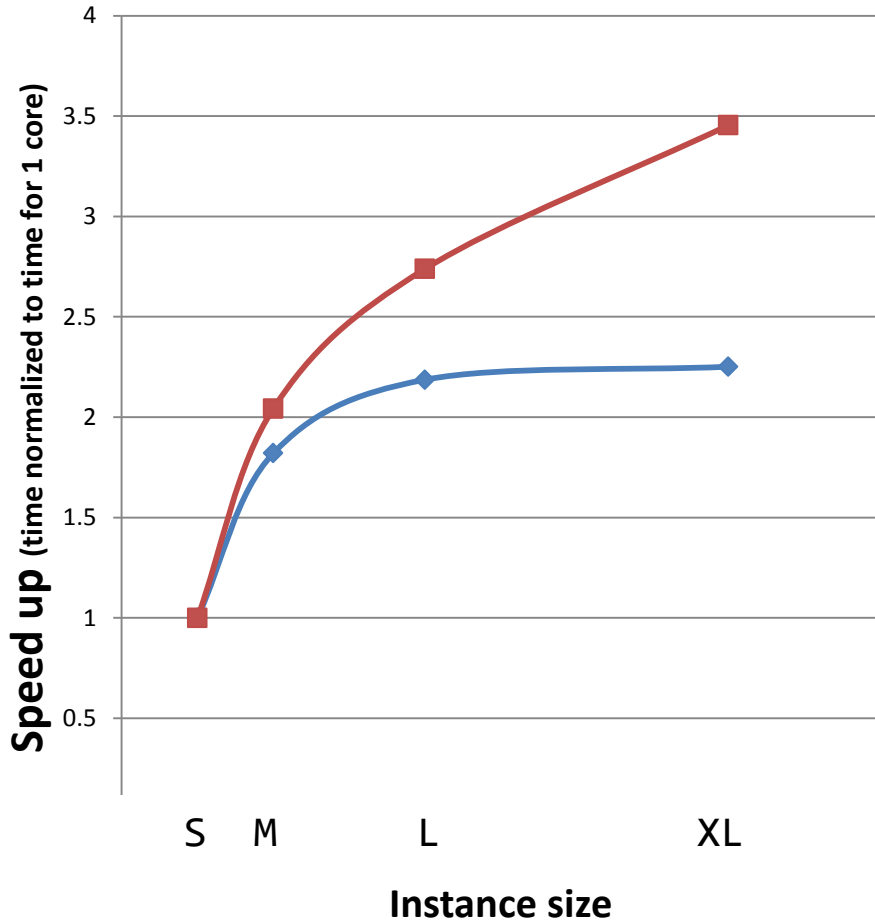
```
private void ProcessPoints()
{
    CloudBlobBlob pointsBlob = AzureHelper.GetBlob(task.Points);
    // Do the mapping and write the new blob
    using (ObjectStreamReader<ClusterPoint> stream = new ObjectStreamReader<ClusterPoint>(pointsBlob,
ClusterPoint.FromByteArray, ClusterPoint.Size, task.PartitionNumber, task.M))
    {
        var assignedPoints = stream.AsParallel().Select(AssignClusterPointToNearestCentroid);
        ObjectBlockWriter<ClusterPoint> writeStream = new ObjectBlockWriter<ClusterPoint>(pointsBlob, point =>
point.ToByteArray(), ClusterPoint.Size);
        TaskResult.NumPointsChanged = 0;
        TaskResult.PointsProcessedDataByCentroid = new Dictionary<Guid, PointsProcessedData>();
        // Piped execution -- see http://msdn.microsoft.com/en-us/magazine/cc163329.aspx
        foreach (var result in assignedPoints)
        {
            // Write the point to the new blob
            writeStream.Write(result.Point);
            // Update the number of points changed counter
            if (result.PointWasChanged)
            {
                TaskResult.NumPointsChanged++;
            }
            // Add to the appropriate centroid group
            if (!TaskResult.PointsProcessedDataByCentroid.ContainsKey(result.Point.CentroidID))
            {
                TaskResult.PointsProcessedDataByCentroid[result.Point.CentroidID] = new PointsProcessedData();
            }
            TaskResult.PointsProcessedDataByCentroid[result.Point.CentroidID].NumPointsProcessed++;
            TaskResult.PointsProcessedDataByCentroid[result.Point.CentroidID].PartialPointSum += result.Point;
        }
        // Send the block list as part of TaskResult
        writeStream.FlushBlock();
        TaskResult.PointsBlockList = writeStream.BlockList;
    }
}

private ClusterPointProcessingResult AssignClusterPointToNearestCentroid(ClusterPoint clusterPoint)
{
    ClusterPoint result = new ClusterPoint(clusterPoint);
    result.CentroidID = centroids.MinElement(centroid => Point.Distance(clusterPoint, centroid)).ID;
    return new ClusterPointProcessingResult
    {
        Point = result,
        PointWasChanged = clusterPoint.CentroidID != result.CentroidID
    };
}
```

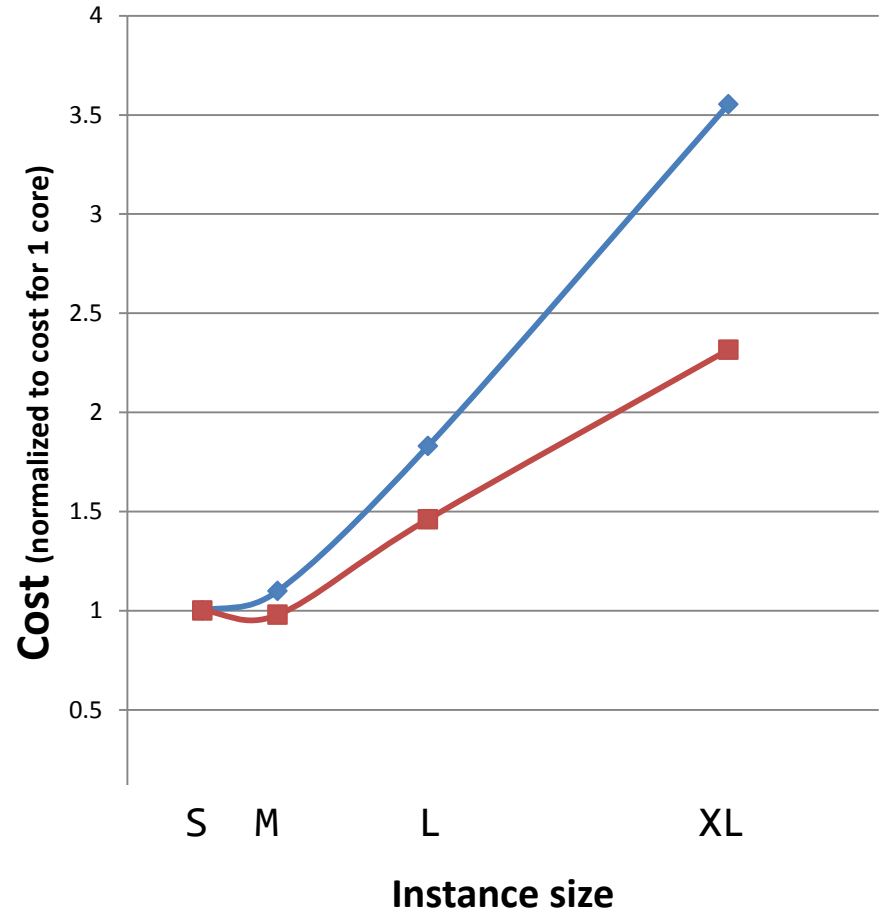
# CloudClustering: Multicore Parallelism



**Scale-up: Speed up for varying instance size, PLINQ vs. Threads**

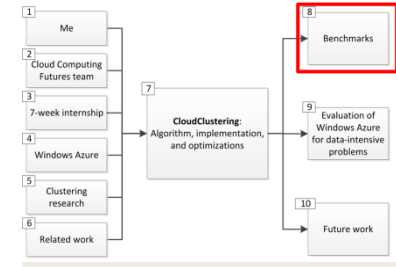


**Scale-up: Cost for varying instance size, PLINQ vs. Threads**

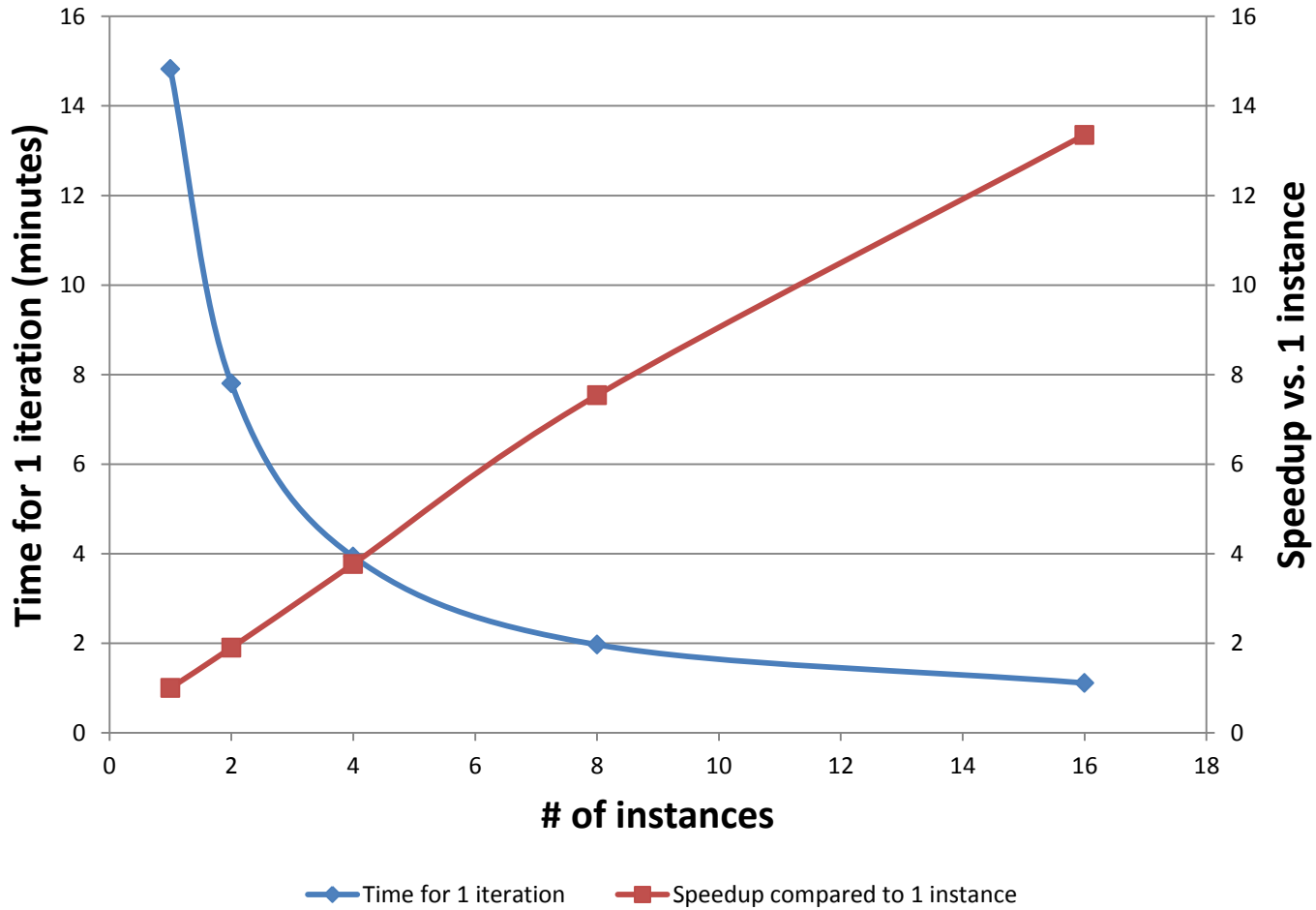


◆ PLINQ speedup    ■ Threads speedup

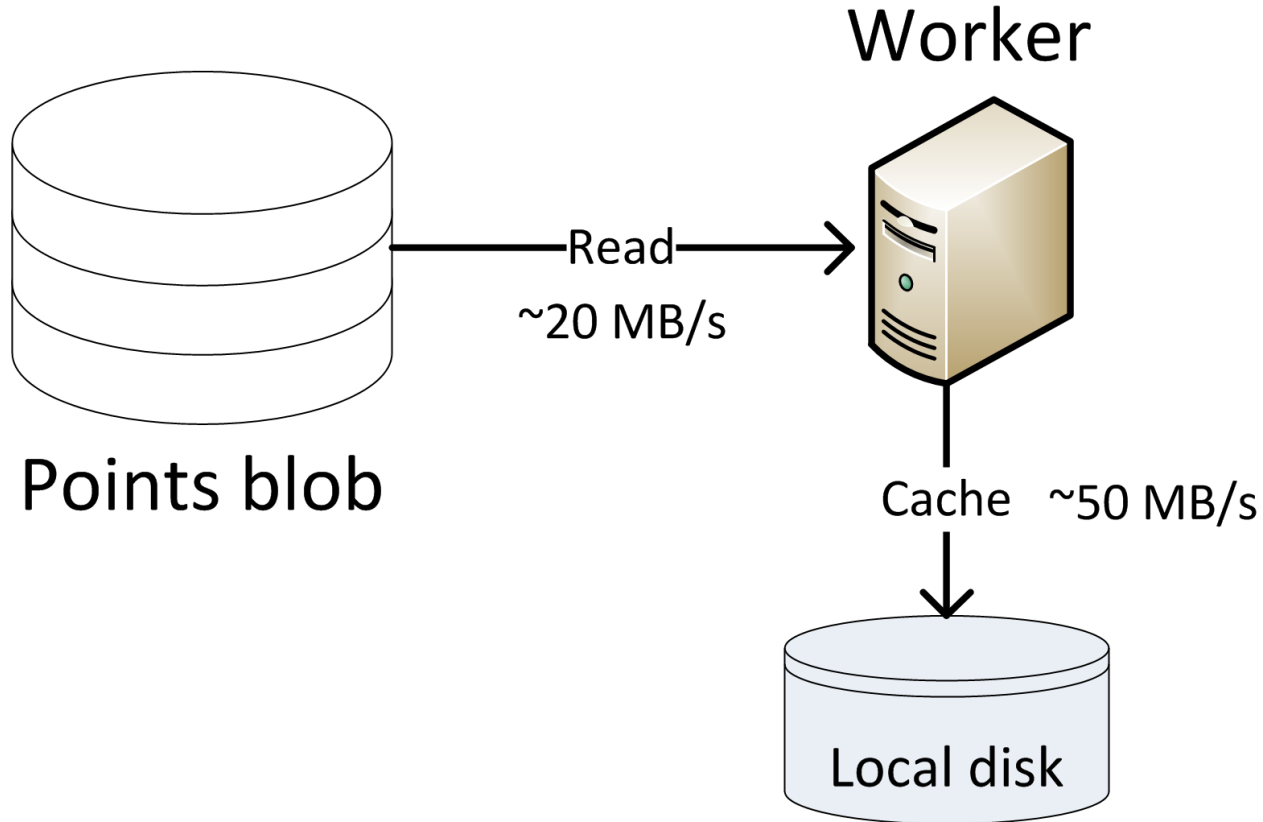
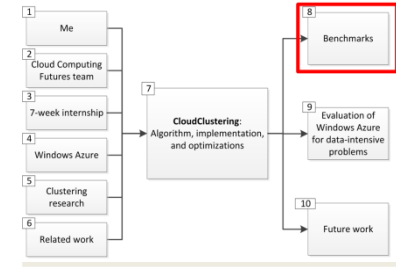
◆ PLINQ cost    ■ Threads cost



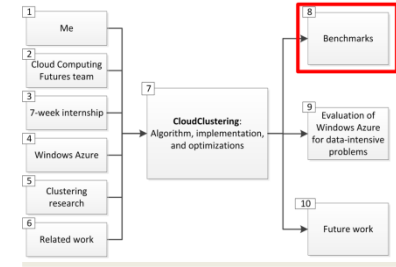
## Scale-out: Time and speedup for varying instance counts



# CloudClustering: Data Affinity

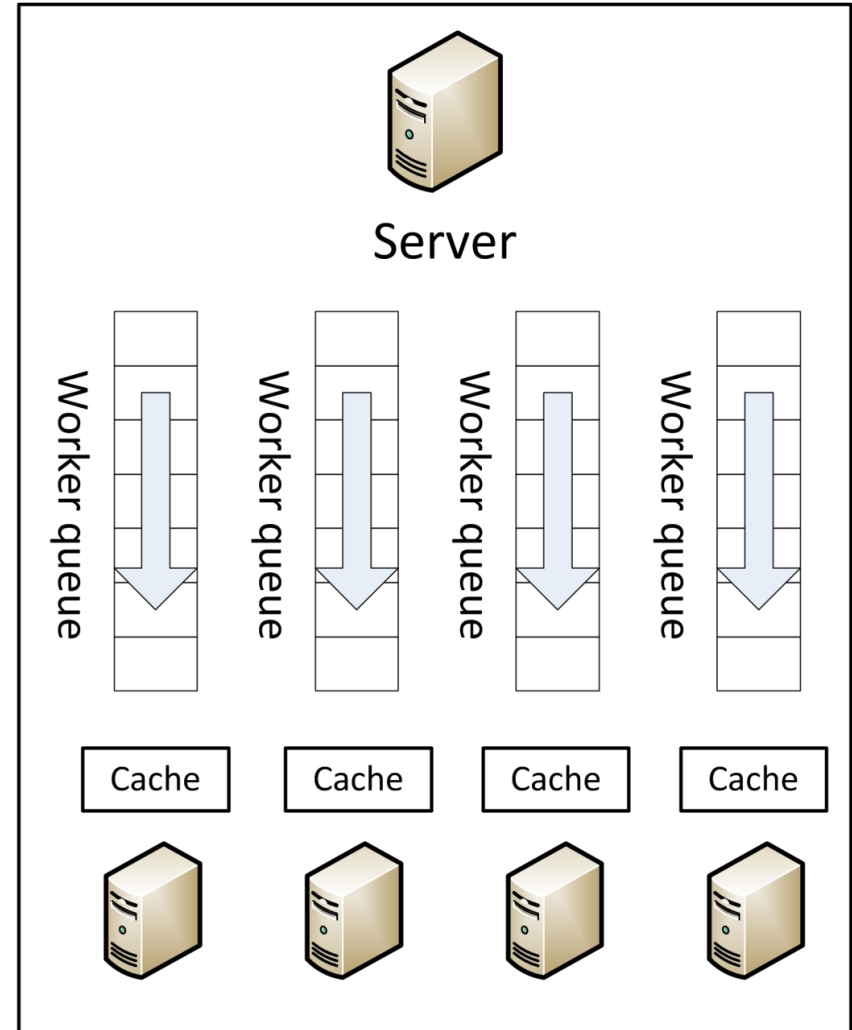
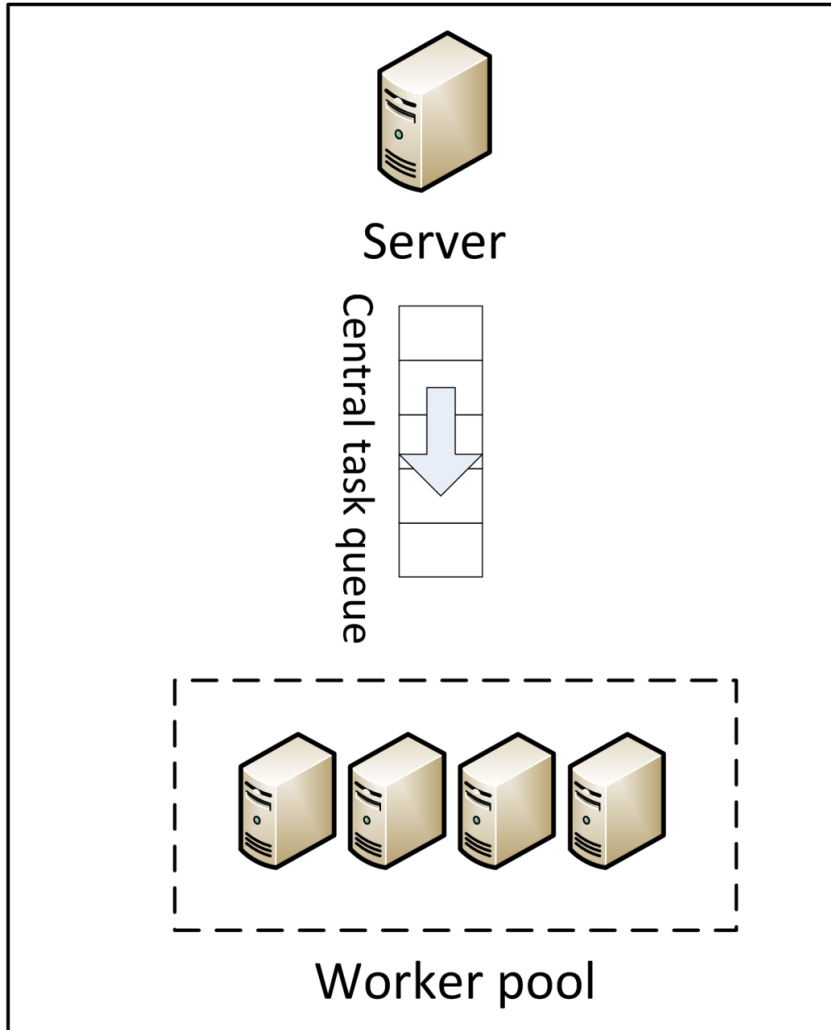


# CloudClustering: Data Affinity

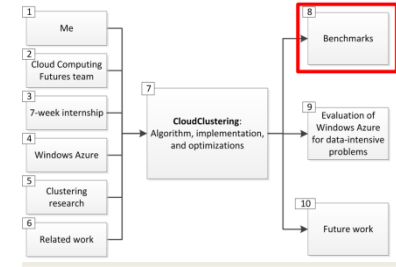


Typical Azure architecture

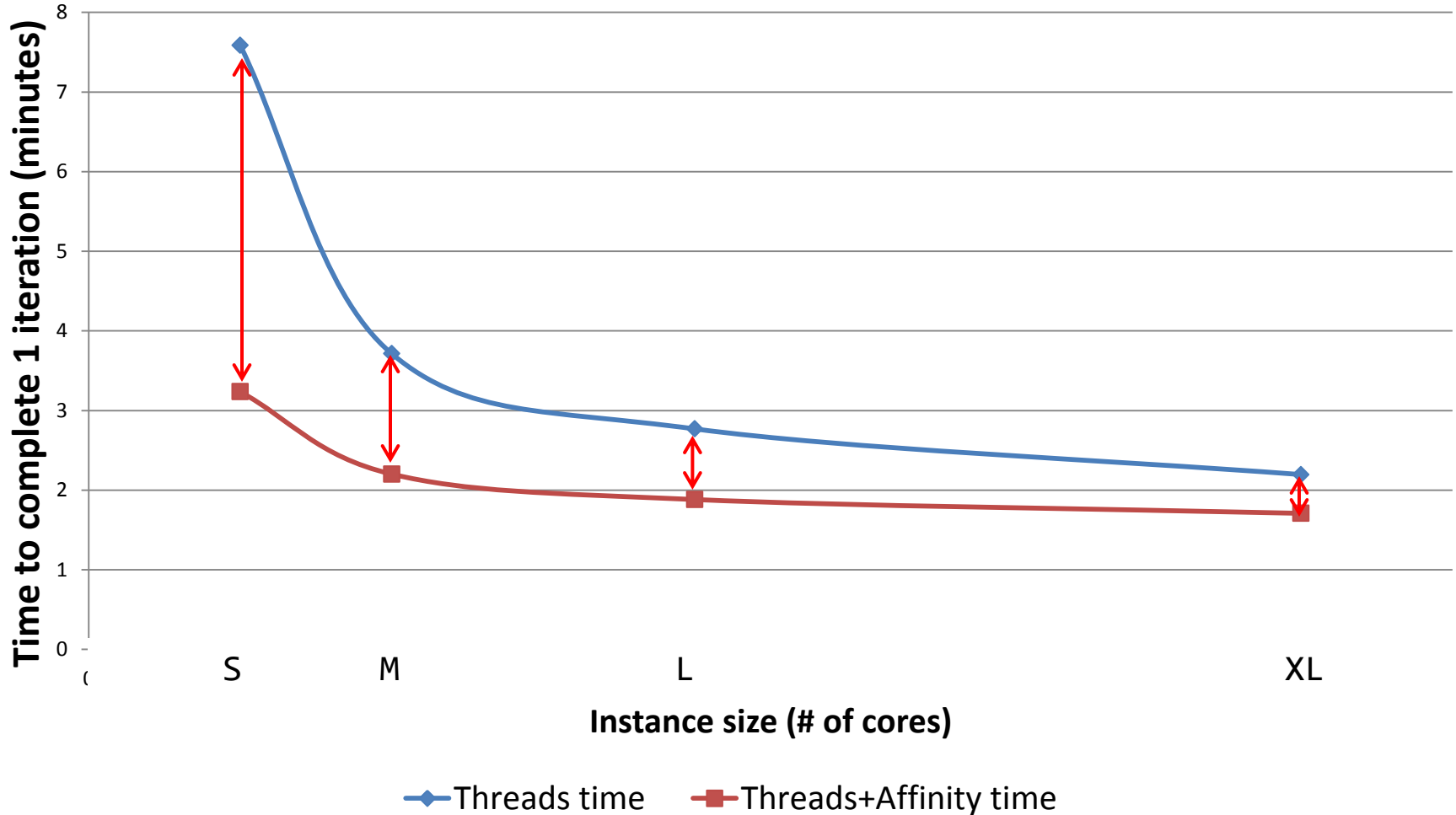
Architecture with data affinity



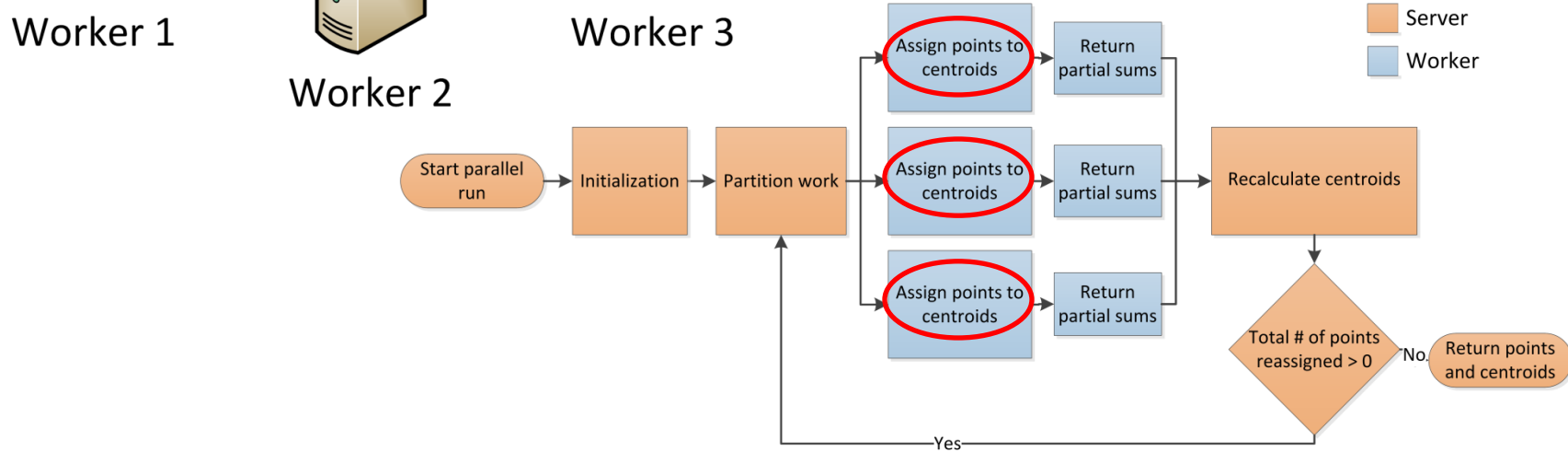
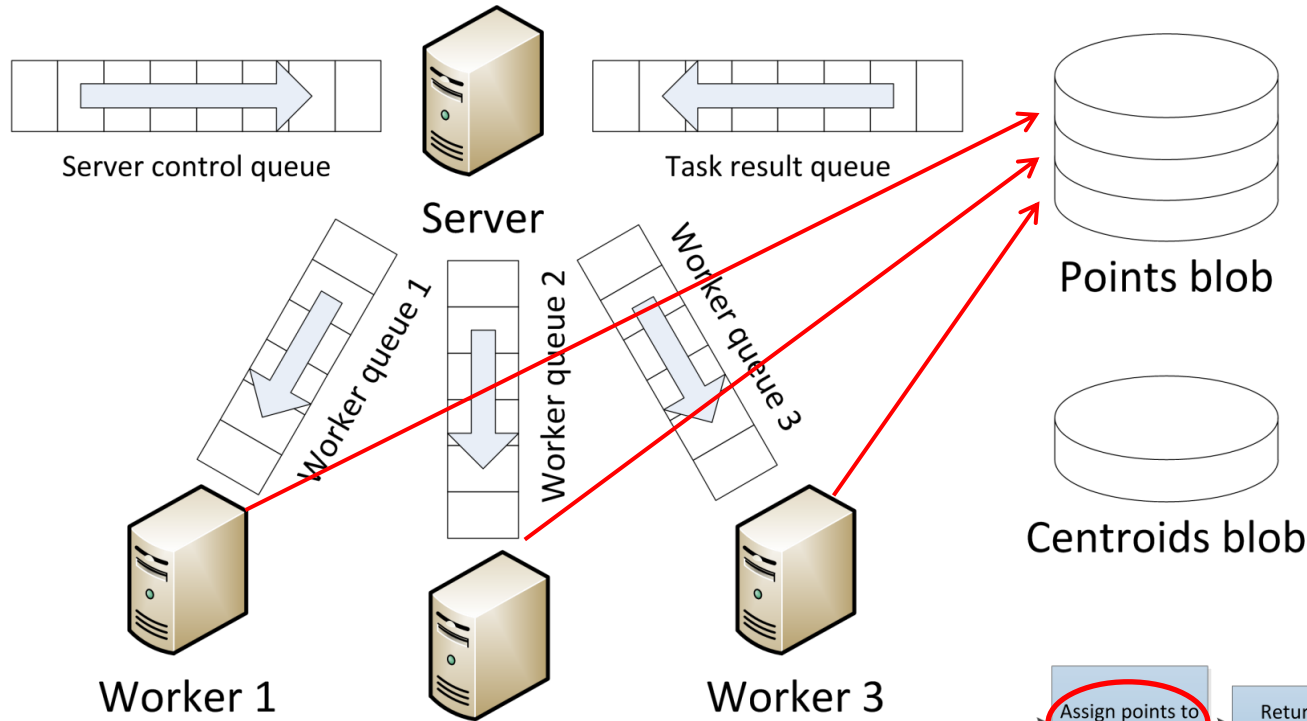
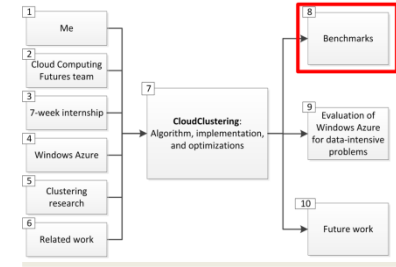
# CloudClustering: Data Affinity



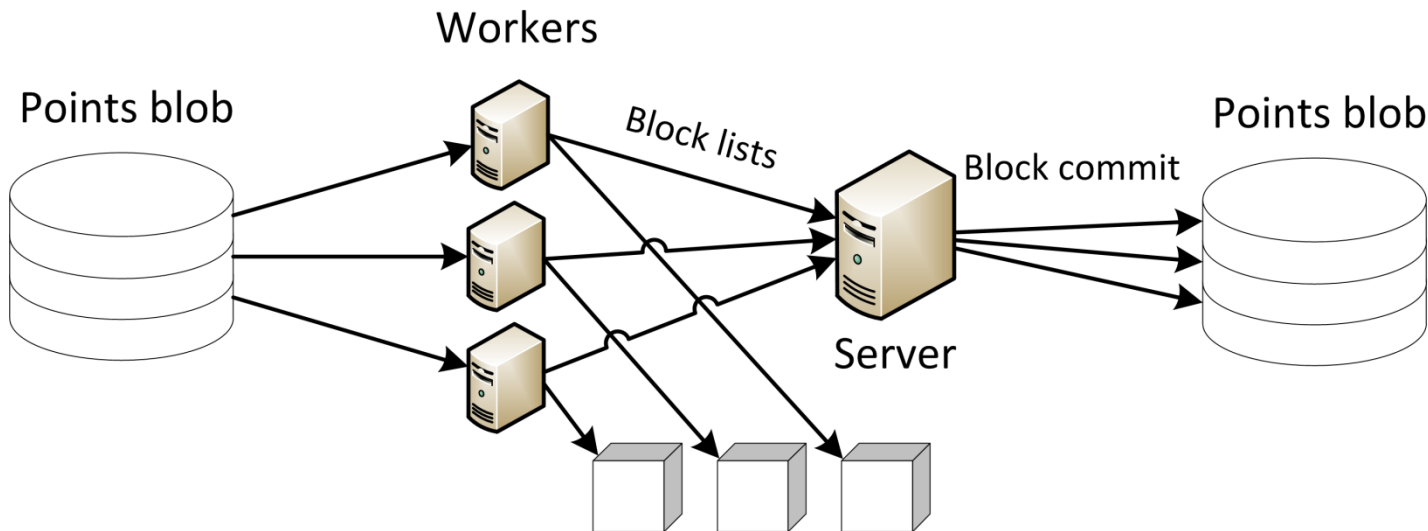
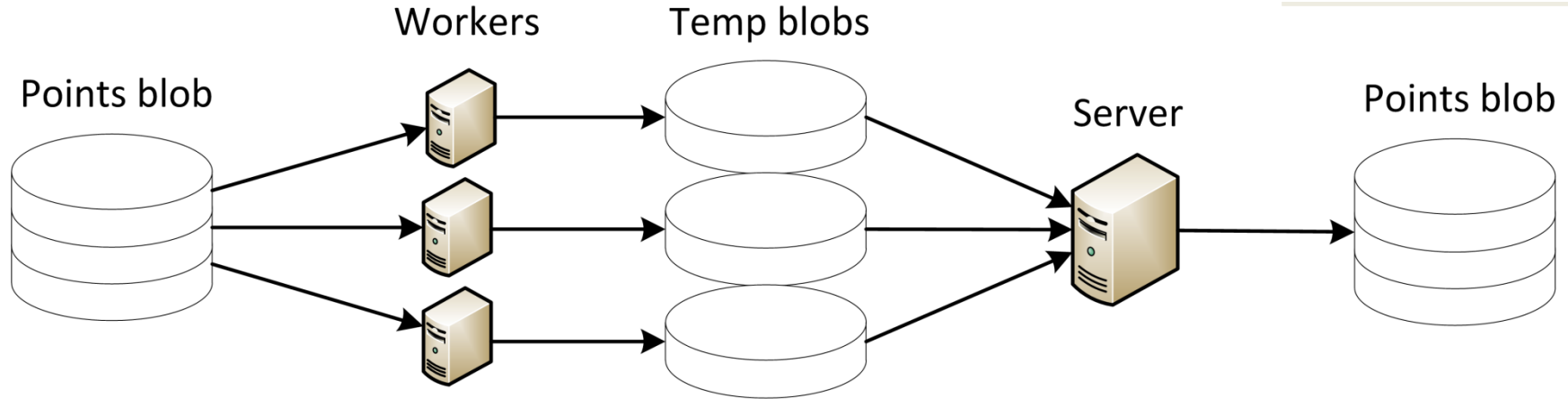
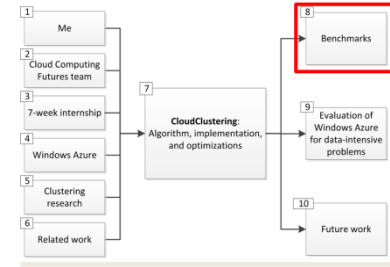
Time for varying instance size, No Affinity vs. Affinity (Threads, 4 instances)



# CloudClustering: Efficient Blob Concatenation

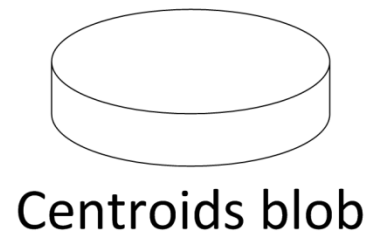
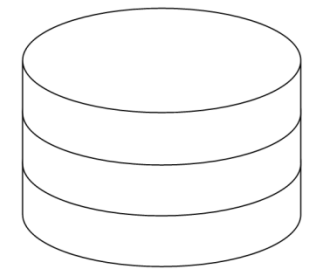
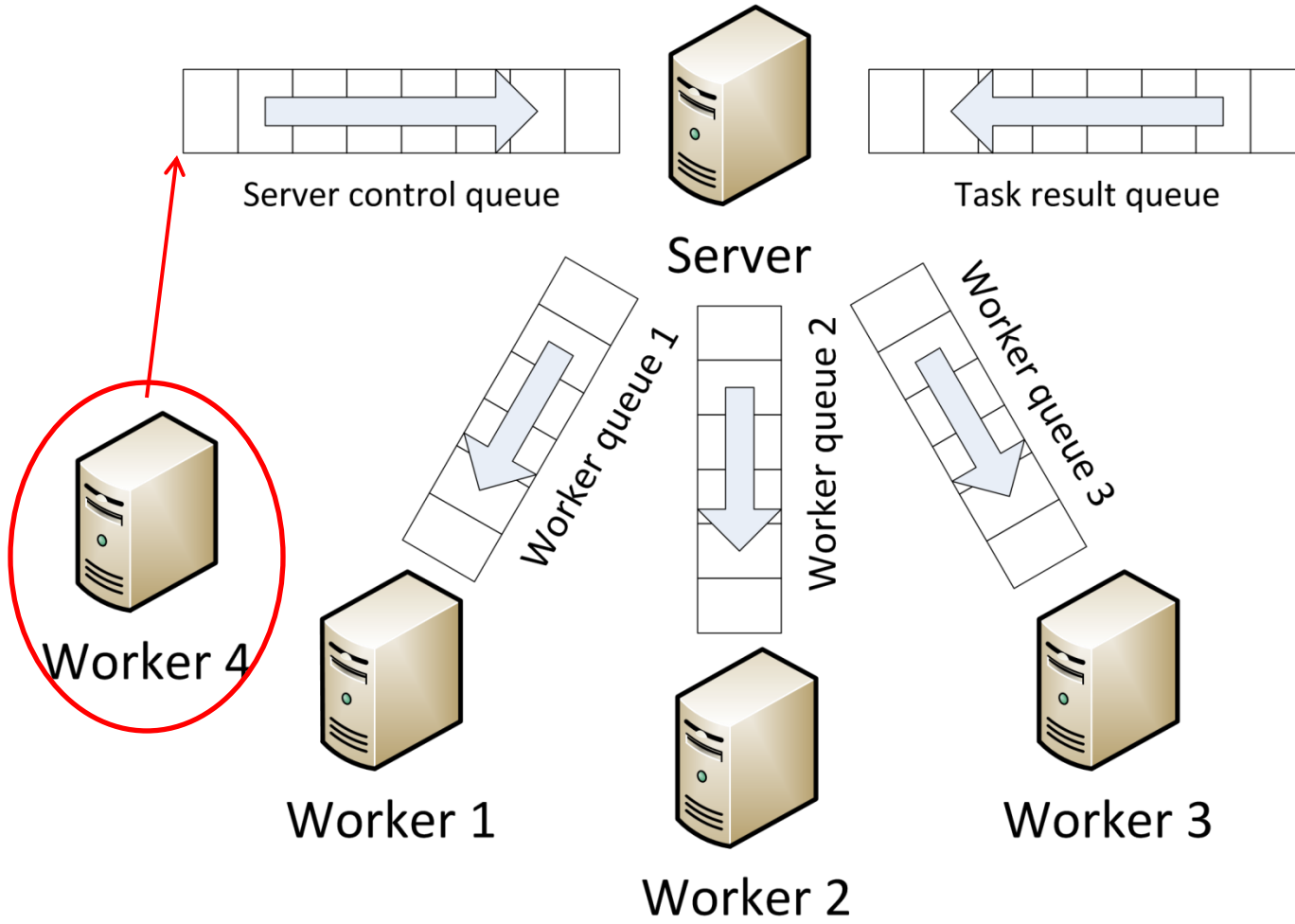
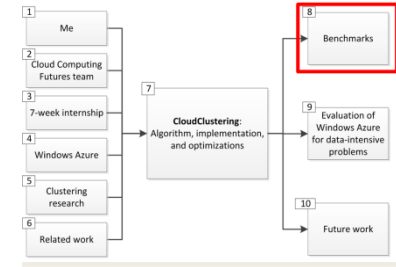


# CloudClustering: Efficient Blob Concatenation

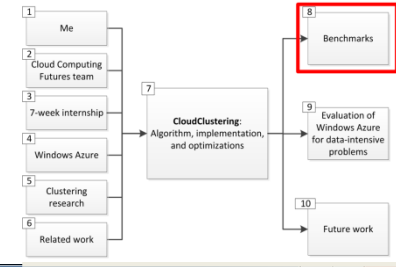




# CloudClustering: Dynamic Scalability



# CloudClustering: Demo



Home Page x +

http://cloudclustering.cloudapp.net/

## CLOUD CLUSTERING

ANKUR DAVE [ Log In ]

Home About

### JOB INPUT

Points (n):

or Points File:  No file chosen

or Points Blob URI:

Clusters (k):

Max iterations:  (0 is unlimited)

Notification email:

STATUS

Click Run K-Means to see results.

Download Log

POINTS BLOB

CENTROIDS BLOB

### PERFORMANCE

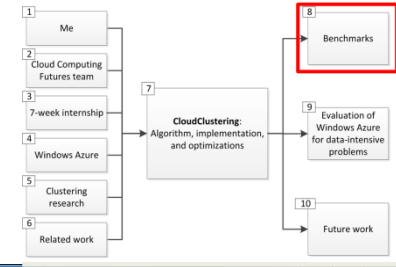
Method	Min time (s)	Average time (s)	Max time (s)	Count
Iteration Method				Time taken (s)

### WORKERS

Worker ID
-----------

### VISUALIZATION

# CloudClustering: Demo



Home Page x +

http://cloudclustering.cloudapp.net/

## CLOUD CLUSTERING

ANKUR DAVE [ Log In ]

Home About

### JOB INPUT

Points (n):

or Points File:  No file chosen

or Points Blob URI:

Clusters (k):

Max iterations:  (0 is unlimited)

Notification email:

STATUS

Done!

[Download Log](#)

POINTS BLOB

<http://cloudclusteringstore.blob.core.windows.net/27e05c29-0782-4556-b84a-e94467dfc13d/points>

CENTROIDS BLOB

<http://cloudclusteringstore.blob.core.windows.net/27e05c29-0782-4556-b84a-e94467dfc13d/centroids>

### PERFORMANCE

	Method	Min time (s)	Average time (s)	Max time (s)	Count
0	InitializeStorage	6.2030853	6.2030853	6.2030853	1
0	EnqueueTasks	0.5624964	0.6874956	1.0781181	5
0	ProcessNewTask	2.5156411	4.23124963	9.5625612	20
0	ProcessWorkerResponse	0.1874988	0.246092175	0.4218723	20
0	RecalculateCentroids	0.7187454	0.75937014	0.8281197	5

Iteration	Method	Time taken (s)
0	InitializeStorage	6.2030853
0	EnqueueTasks	0.624996
0	ProcessNewTask	9.5625612
0	ProcessNewTask	9.4531511
0	ProcessNewTask	9.2968155
0	ProcessNewTask	8.90625
0	ProcessWorkerResponse	0.4218723
0	ProcessWorkerResponse	0.2343735
0	ProcessWorkerResponse	0.2187486
0	ProcessWorkerResponse	0.1874988
1	RecalculateCentroids	0.8281197
1	EnqueueTasks	0.6093711
1	ProcessNewTask	2.7500176
1	ProcessNewTask	3.0937302
1	ProcessNewTask	2.625
1	ProcessNewTask	2.5156411
1	ProcessWorkerResponse	0.2968731
1	ProcessWorkerResponse	0.3281229
1	ProcessWorkerResponse	0.2812482
1	ProcessWorkerResponse	0.2343735
1	RecalculateCentroids	0.7243702

### WORKERS

Worker ID

074b74bd-2954-4170-96f7-71676578cfdc

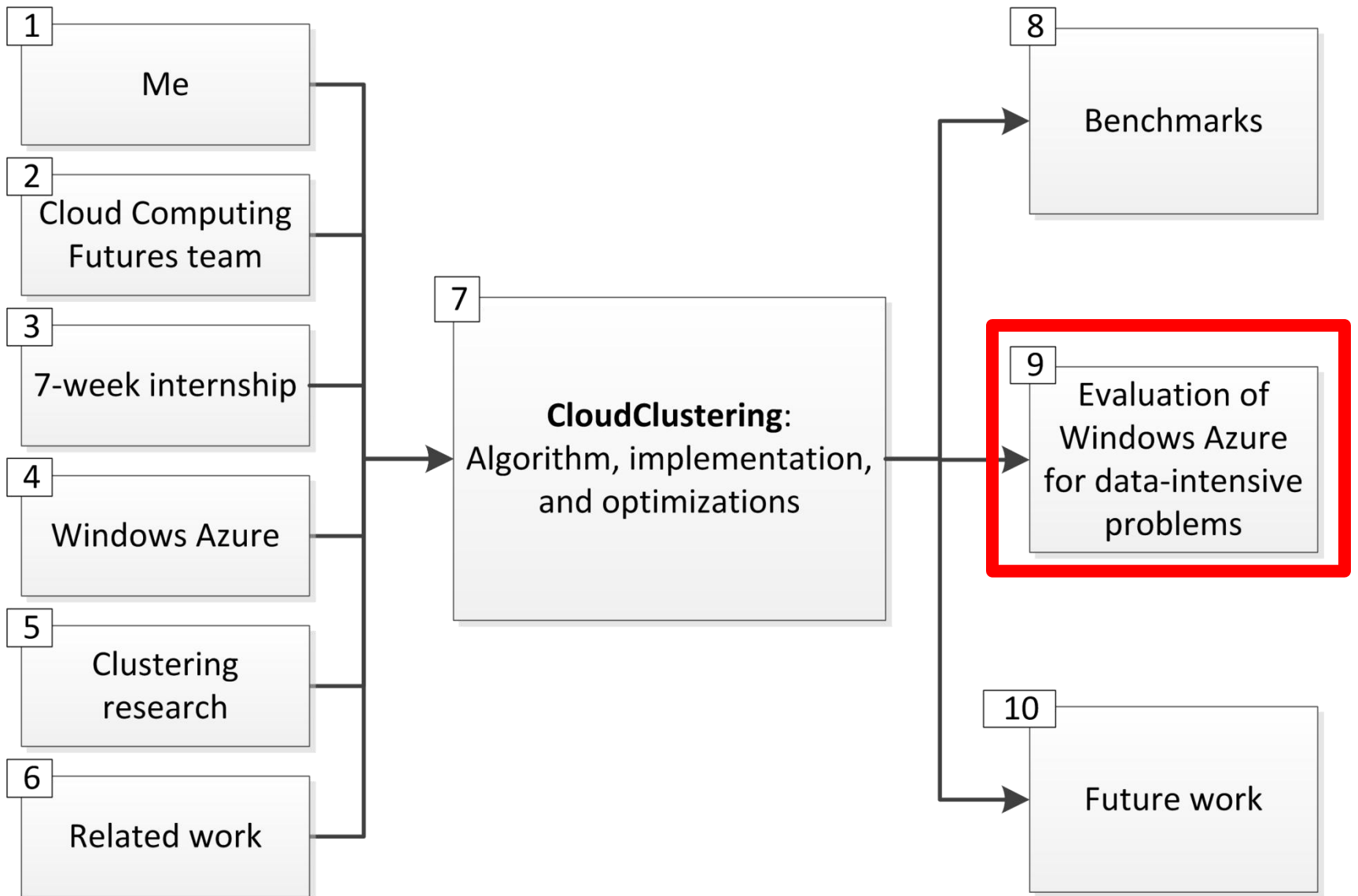
791dbb9b-73cc-4464-bea4-f473e1881760

c90eeba9-89a2-4db9-b751-fbe7139b8e65

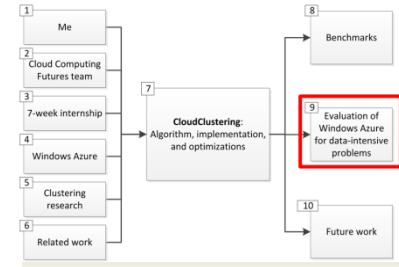
fda65ade-6968-47f2-9ed6-bc66330bfa93

### VISUALIZATION

# Agenda



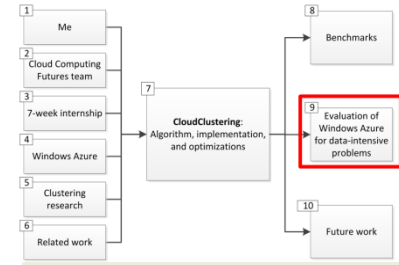
# Windows Azure: Benefits



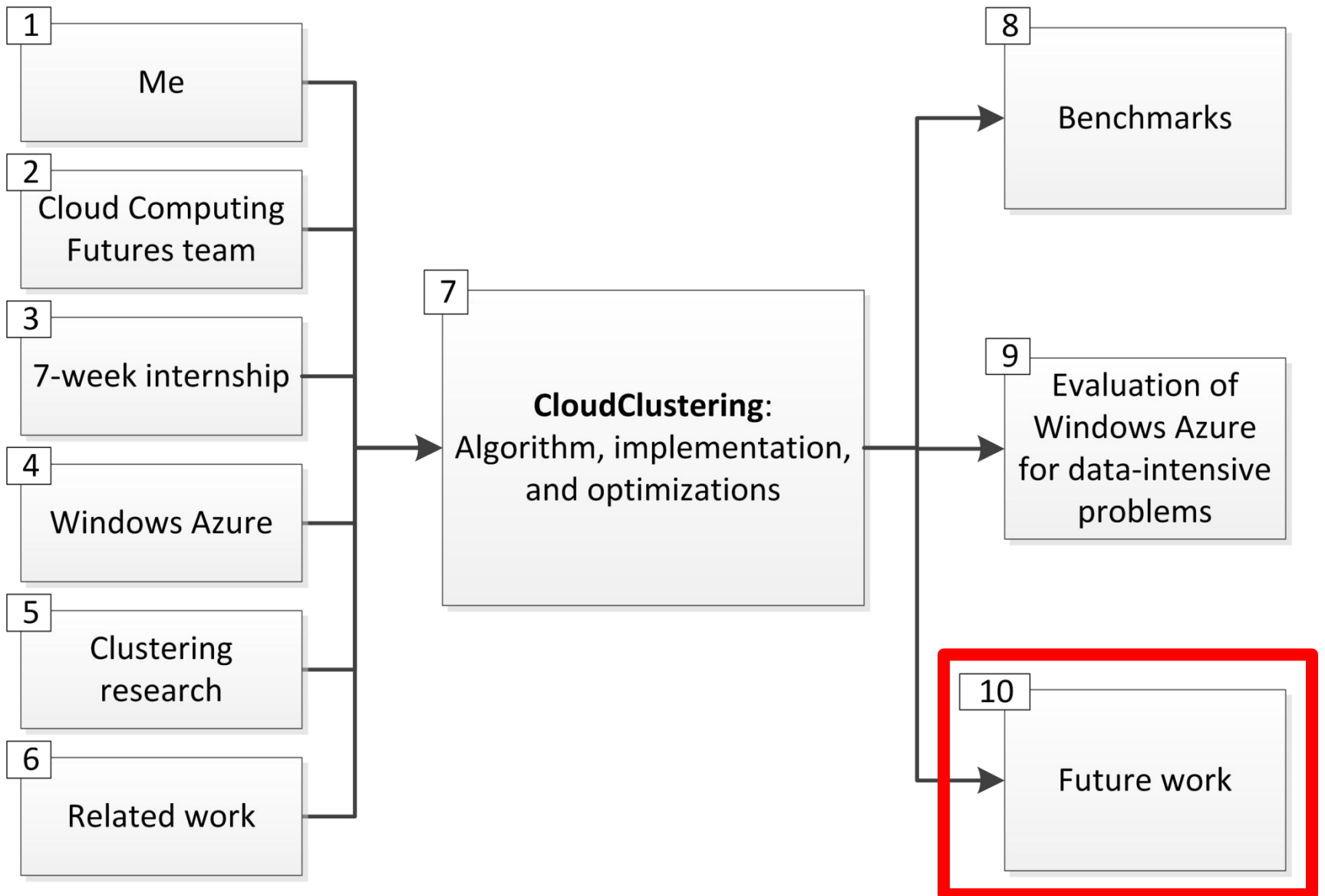
Azure is an **appropriate level of abstraction** for data-intensive algorithms like *k*-means.

# Windows Azure: Potential Problem Areas

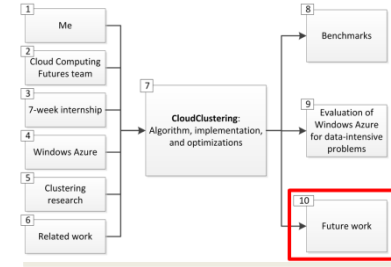
- On the cloud, **cost** scales directly with **usage**
  - Sub-linear speedups are not good enough!
- For data-intensive algorithms, **data affinity** gives great performance... but there's a **tradeoff**
  - Dynamic scaling is more complex
  - Fault-tolerance is even harder
- Performance test to find configuration **sweet spots**



# Agenda



# Future work



- A **compromise between worker pools and data affinity** that retains scalability and fault-tolerance
  - Buddy system
- Improved **caching** using blocks
- Fundamental improvements to the ***k*-means algorithm**
  - More efficient stopping condition
  - “Lazy” processing that eliminates synchronization barriers
- Further optimizations to **multicore parallelism**



# Ankur Dave

[ankurdave@gmail.com](mailto:ankurdave@gmail.com)

<http://ankurdave.com>