Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Clustering

Gruppe 3
Robert Pfeiffer
Tobias Schmidt

20. Juli 2009

Clustering

1. Unsere Daten
2. Was ist „Clustern“?
3. K-Means
4. Hadoop
5. Implementierungsdetails
6. Evaluierung
7. Fazit

Unsere Daten

Wappen	Deutschlandkarte
	

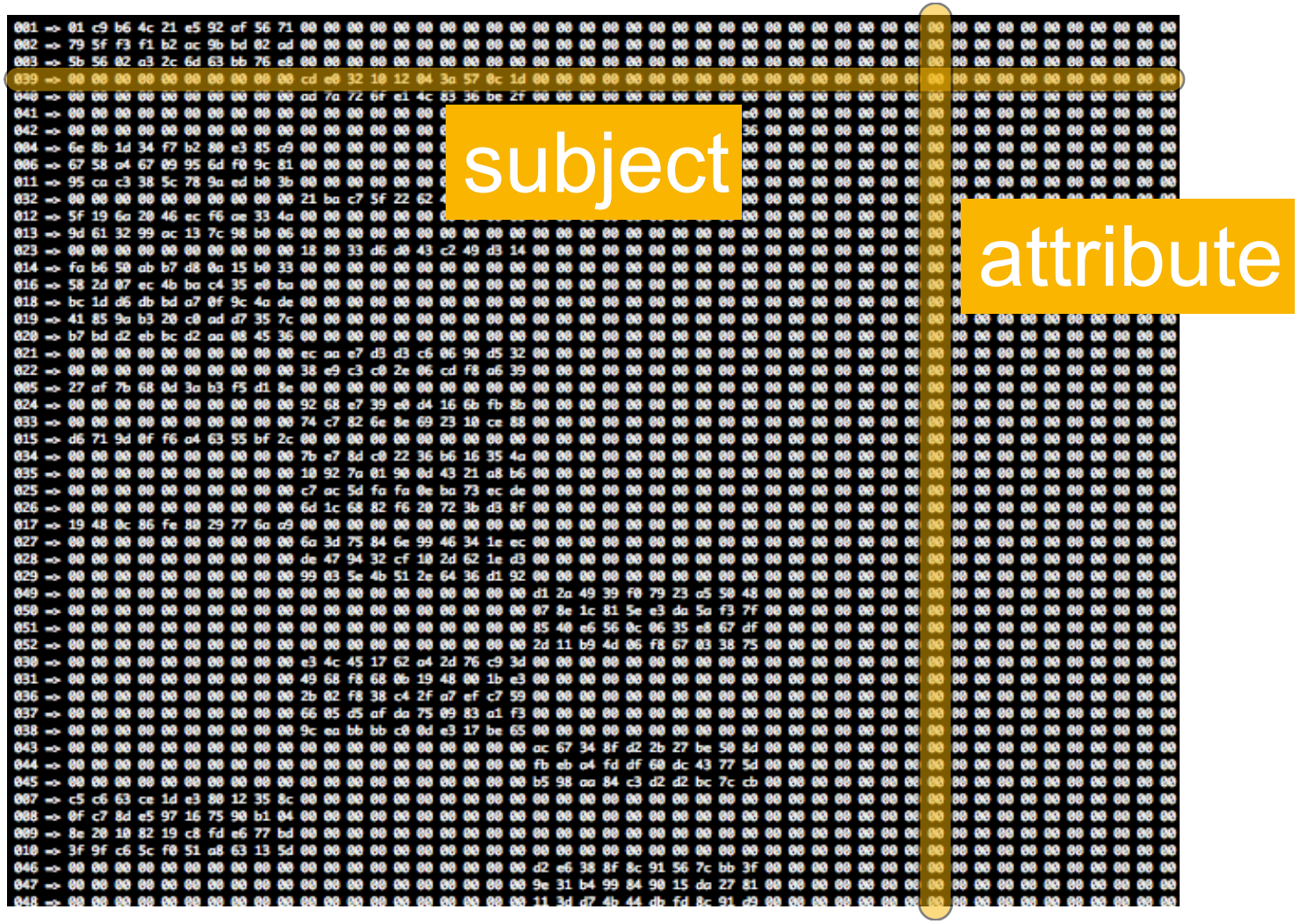
Basisdaten	
Bundesland:	Brandenburg
Landkreis:	Kreisfreie Stadt
Höhe:	35 m ü. NN
Fläche:	187,27 km²
Einwohner:	150 833 *(31. Dez. 2007)*
Bevölkerungsdichte:	805 Einwohner je km²
Postleitzahlen:	14401–14482
Vorwahl:	0331
Kfz-Kennzeichen:	P
Gemeindeschlüssel:	12 0 54 000
Stadtgliederung:	7 Wohngebiete und 9 neue Ortsteile mit je einem Ortsbeirat

→

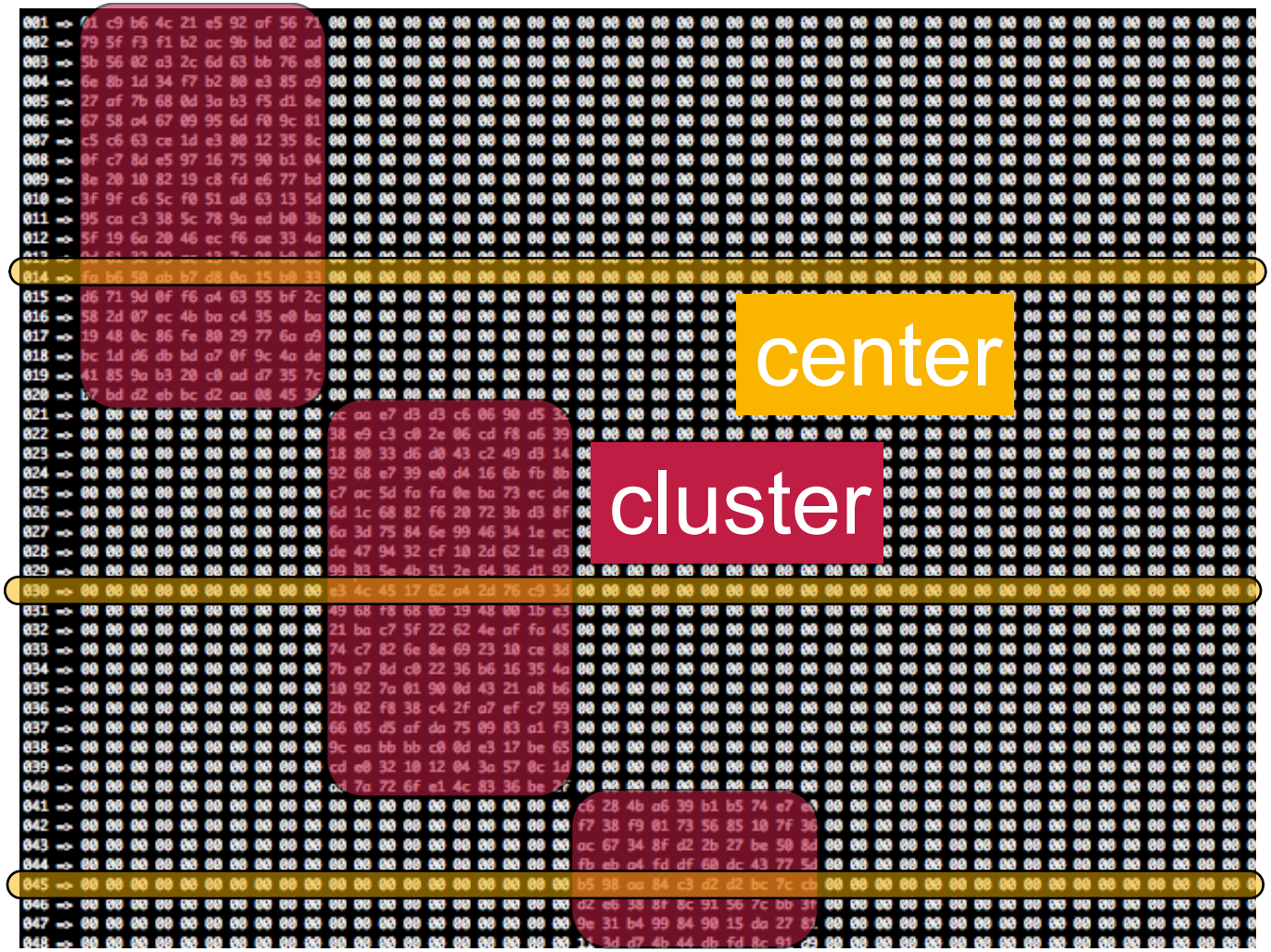
DBpedia

Property	Value
dbpedia-owl:areaCode	0331
dbpedia-owl:areaTotal	187.28
dbpedia-owl:coordinates	{{coord|52|24|0|13|4|0}}
dbpedia-owl:elevation	35-114
dbpedia-owl:leaderTitle	Oberbürgermeister
dbpedia-owl:populationAsOf	2008
dbpedia-owl:populationTotal	151725
dbpedia-owl:postalCode	14401–14482

Was ist „Clustern“?



Was ist „Clustern“?

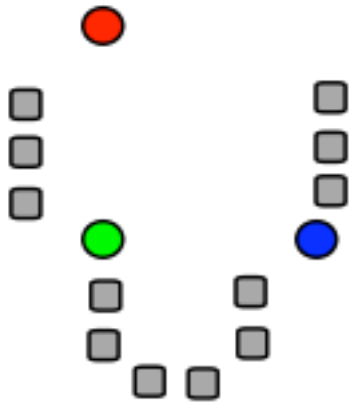


K-Means

Verfahren um Cluster von ähnlichen Subjekten zu ermitteln

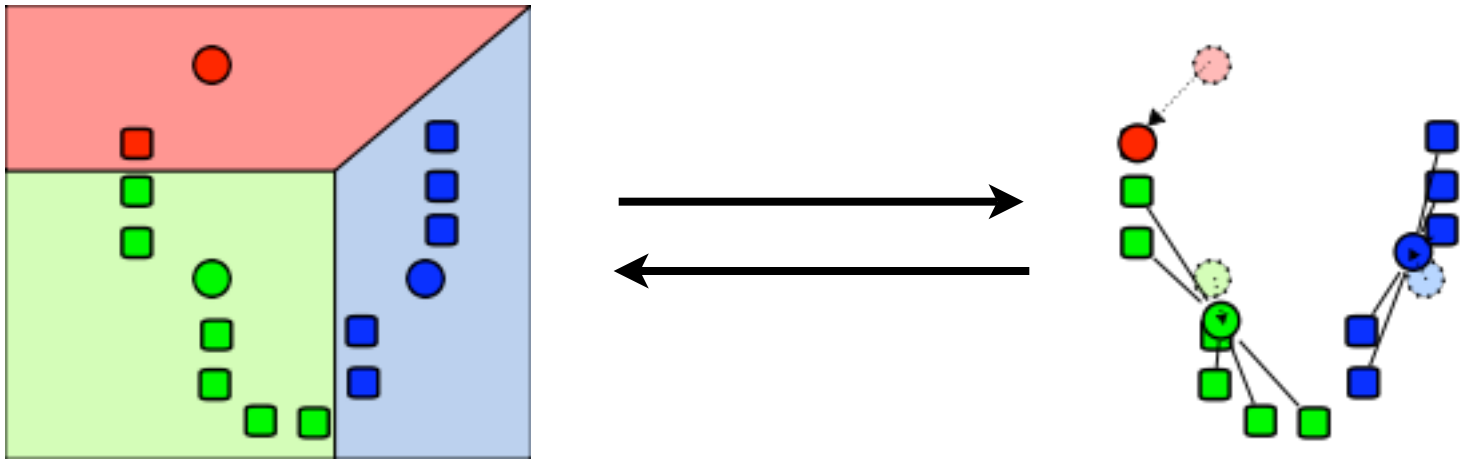
1. Schritt:

- willkürliche Wahl der Clusterzentren



K-Means

2. Schritt:
   - Zuordnung der Subjekte zum nächsten Clusterzentrum
3. Schritt:
   - Neuberechnung des Clusterzentrums



Abstandsmaß

- Ähnlichkeit zwischen zwei Objekten
- totale Ordnung zwischen den Rückgabewerten
- verschiedene Maße in k-means möglich

euklidisches Abstandsmaß

$$d_{AB} = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \quad \text{wobei} \quad A = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \in \mathbb{R}^n \quad \text{und} \quad B = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \in \mathbb{R}^n$$

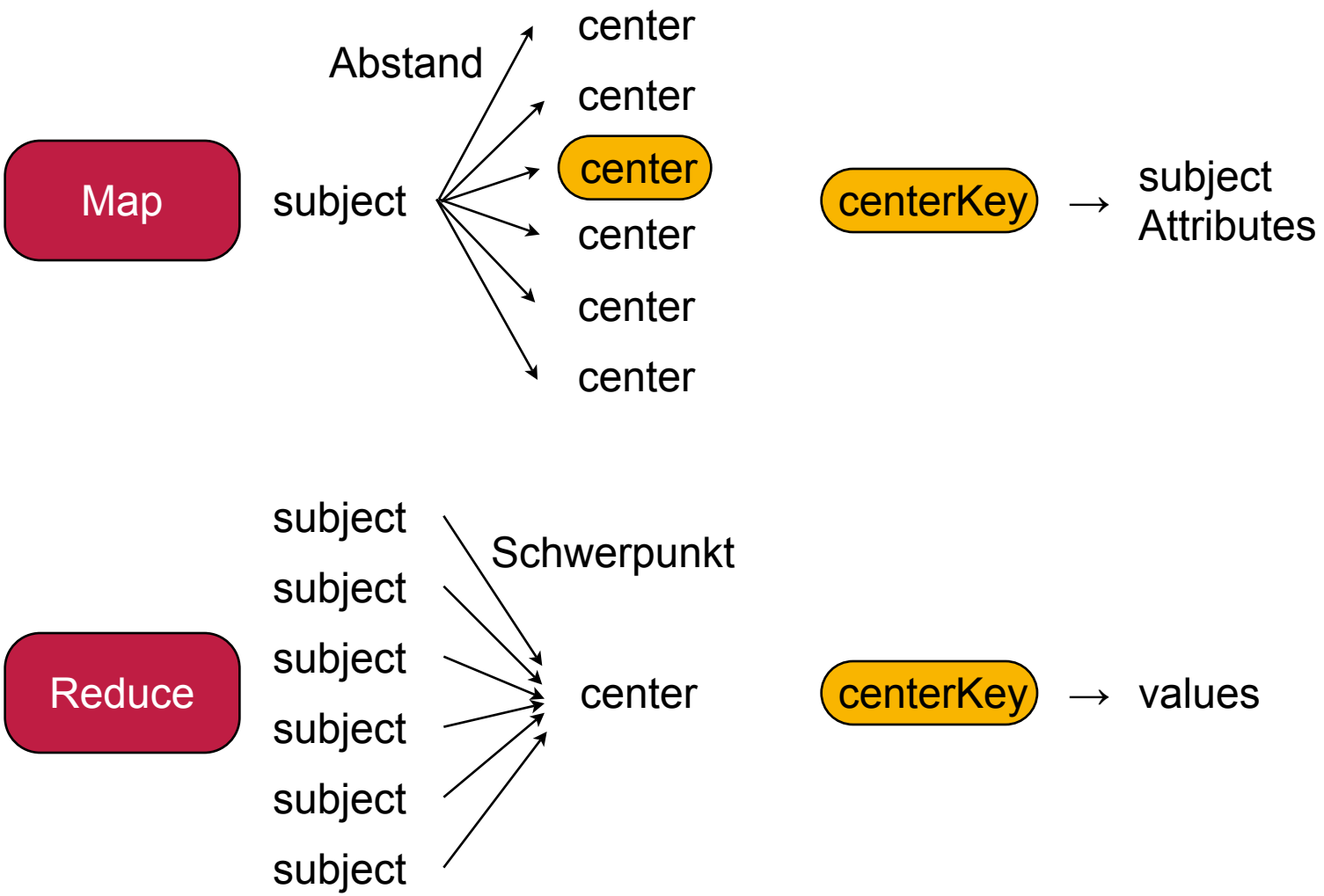
- unsere Subjekte entsprechen Punkte in einem Raum
- Dimension (n) entspricht der Anzahl der Attribute
- Rückgabewert: positive, reelle Zahlen

Jaccard-Abstandsmaß

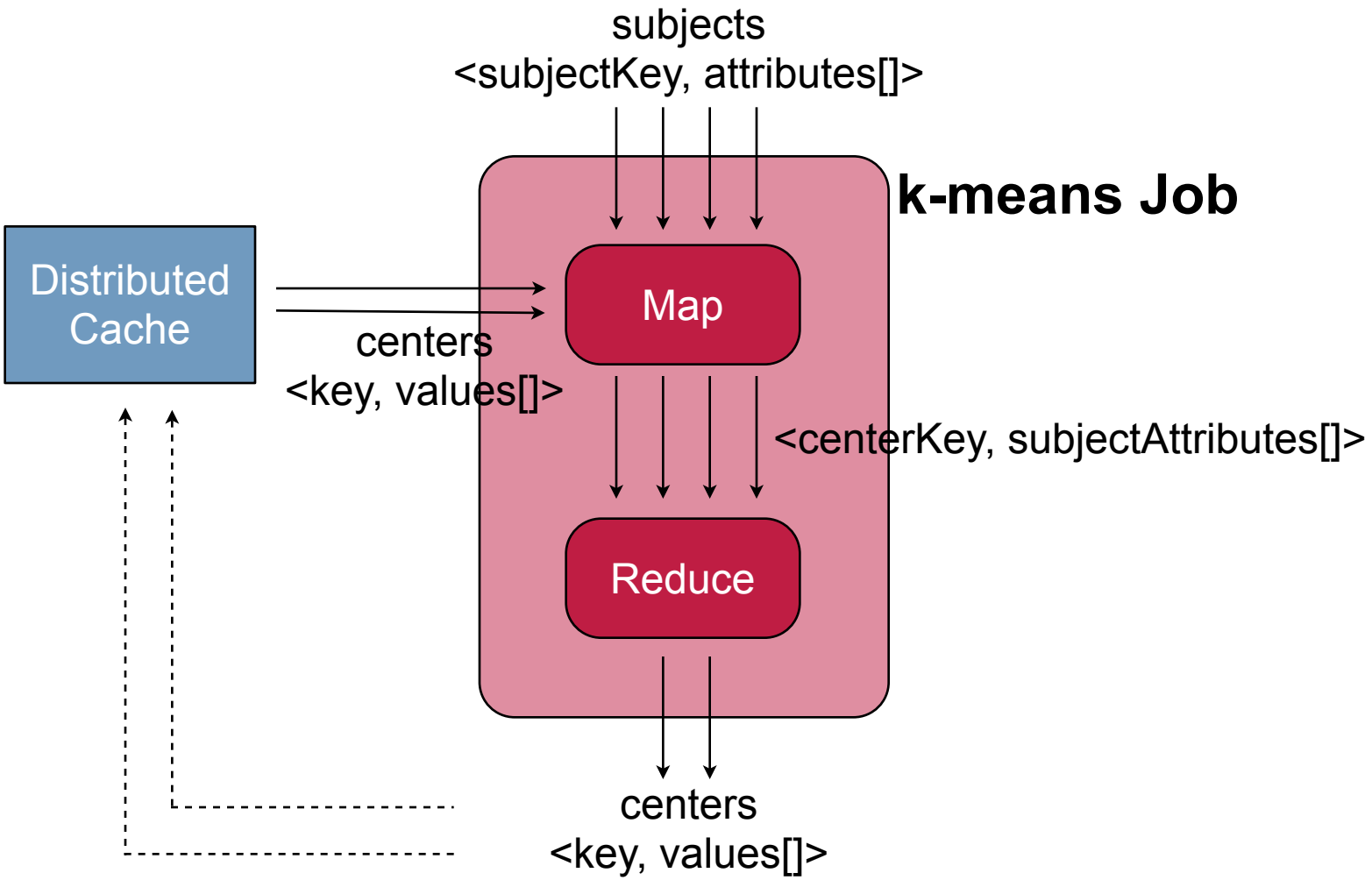
$$J_\delta(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

- unsere Subjekte entsprechen Mengen
- Problem: gemittelte Clusterzentren
- Lösung: Fuzzy Mengen
  - jedes Element ist mit einer bestimmten Wahrscheinlichkeit Teil der Menge
- Rückgabewert: reelle Zahlen zwischen 0 und 1

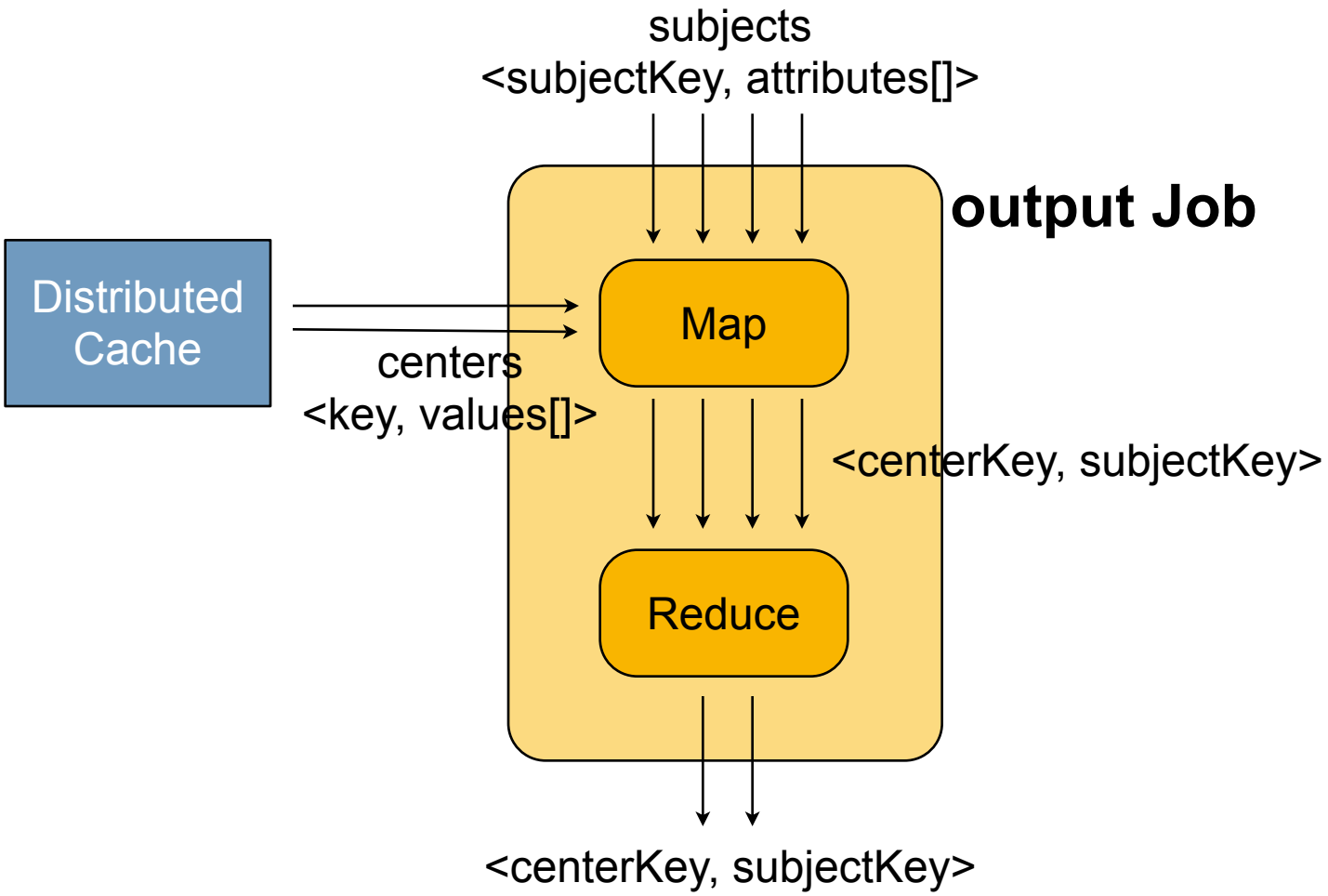
Hadoop


Map
subject
Abstand
center
center
center
center
center
center
centerKey
→
subject
Attributes
Reduce
subject
subject
subject
subject
subject
subject
Schwerpunkt
center
centerKey
→
values

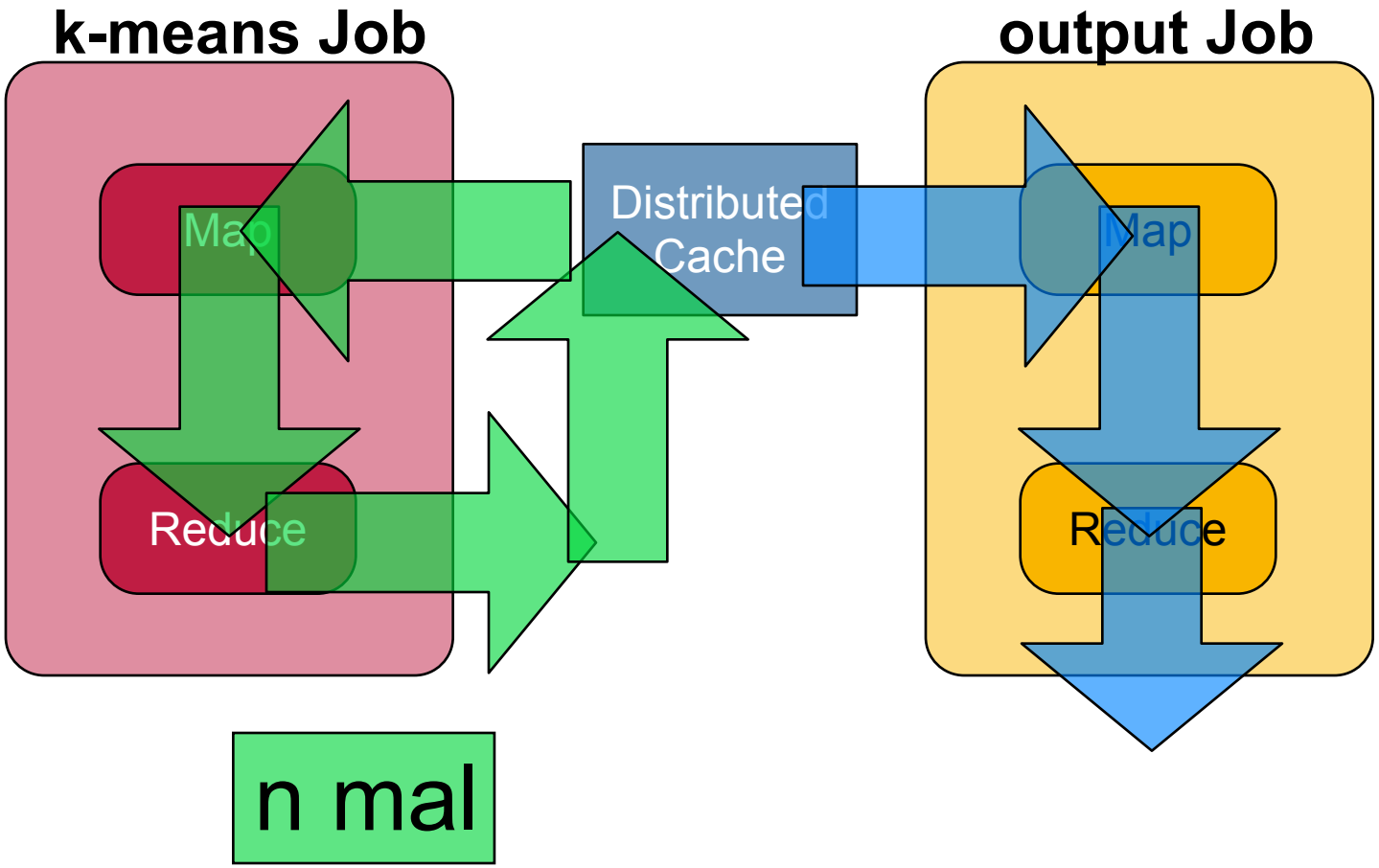
k-means Job


subjects
<subjectKey, attributes[]>
k-means Job
Distributed Cache
Map
centers
<key, values[]>
<centerKey, subjectAttributes[]>
Reduce
centers
<key, values[]>

output Job


subjects
<subjectKey, attributes[]>
output Job
Distributed Cache
centers
<key, values[]>
Map
<centerKey, subjectKey>
Reduce
<centerKey, subjectKey>

Ablauf


k-means Job
output Job
Map
Reduce
Distributed Cache
Map
Reduce
n mal

Sequence Files

- Dateiformat
  - Key/Value-Paare mit Typ
  - Kompression
- Binäres Eingabeformat schlecht verteilbar
- Vorteile
  - gleiches Format für Subjekte und Centers
  - gleiches Format für Input und Output
  - geringe Dateigröße durch Kompression
- Nachteile
  - Dekompression braucht Zeit
  - Vorbereitung der Eingabe nötig

Distributed Cache

- Speicherplatz auf den Nodes
  - vor Beginn eines Jobs synchronisiert
  - zusätzlicher Parameter für Job
- Warum?
  - Alle Nodes brauchen alle Center
- Nachteil
  - Overhead für Synchronisation
  - Mapper nicht mehr unabhängig

Optimierungen

- Distributed Cache
- Sequence Files
- Centers als Festkommazahlen
- weniger Arraydurchläufe bei Jaccard-Abstand
  - Berechnung von Kardinalität, Vereinigung und Durchschnitt in einem Schritt

asymptotische Laufzeitkomplexität

Map-Schritt

- Abstandsmass: Iteration über alle n Attribute
- Abstandsberechnung zu allen k Clustern
- Laufzeit $\in O(n*k)$

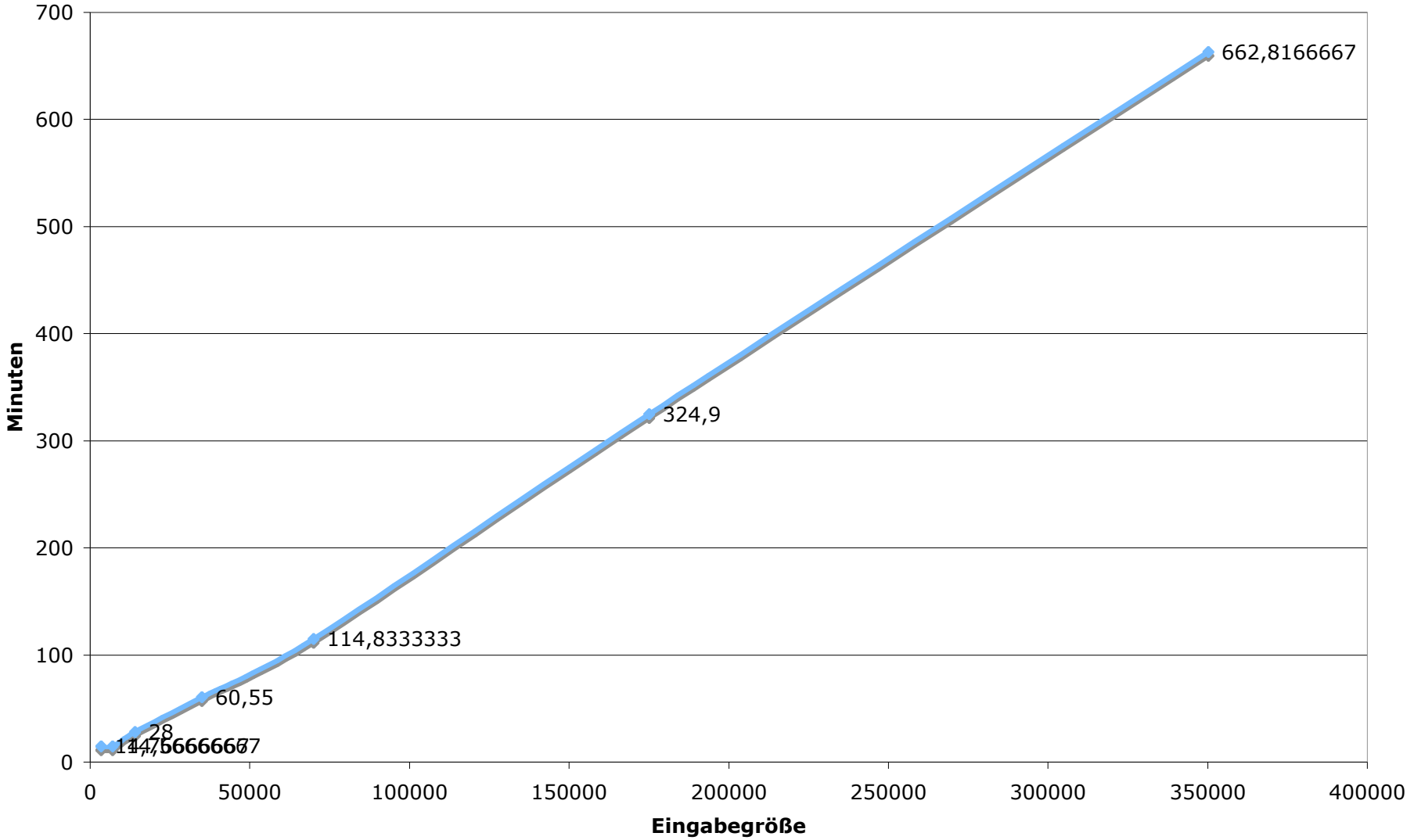
Reduce-Schritt

- Mittelung aller n Attribute aller si Subjekte des Clusters i
- Laufzeit $\in O(n*si)$

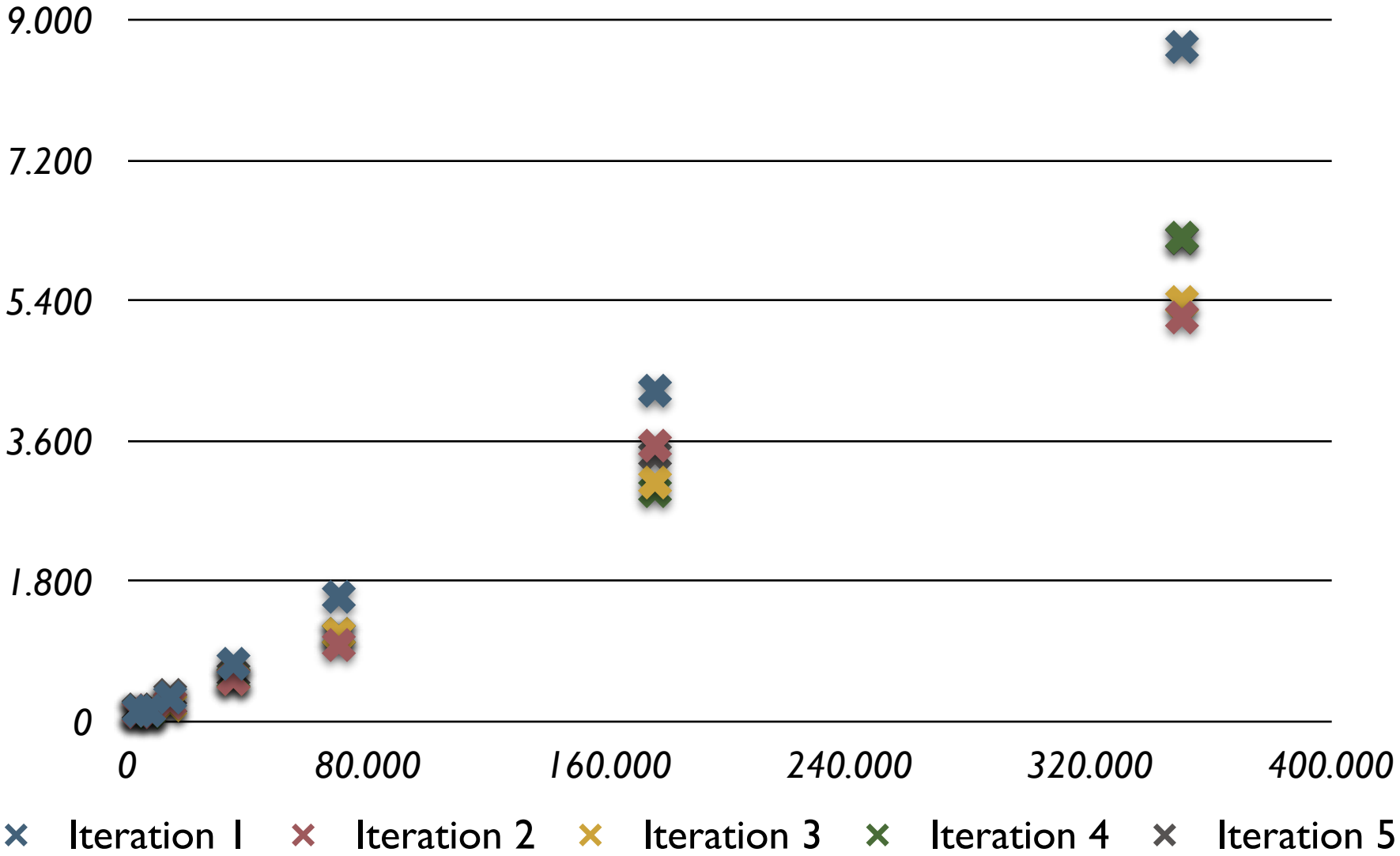
Map-Reduce-Vorgang

- s Subjekte und k Cluster
- s Map-Schritte, k Reduce-Schritte
- Laufzeit $\in O(n*s*k + n*s)$

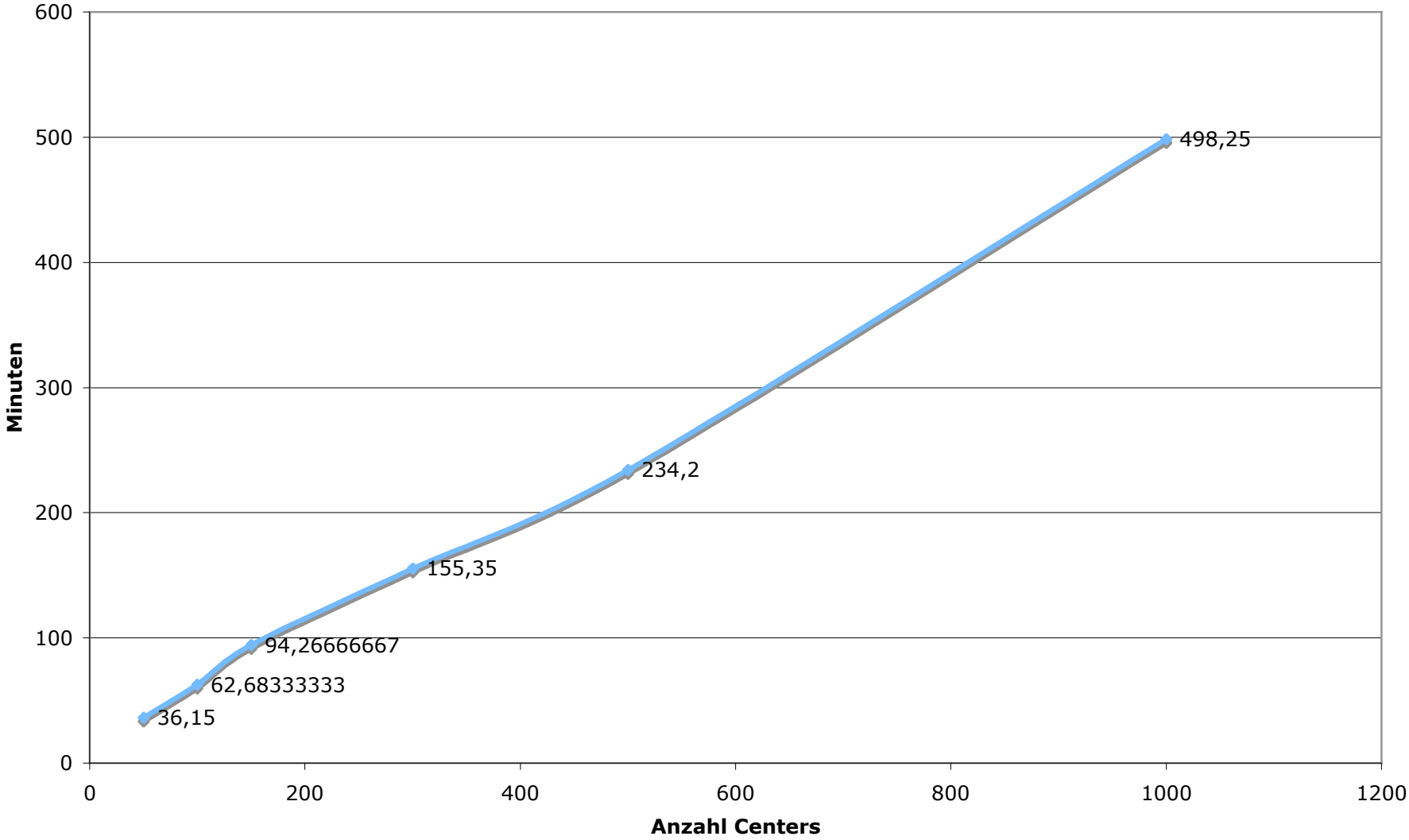
Evaluierung Eingabegröße



Evaluierung Eingabegröße



Evaluierung Center



Fazit

- mit ganz einfacher Implementierung beginnen, dann verfeinern
- Rechenleistung im Hadoop-Cluster stark unterschiedlich
  - 2 von 8 Nodes waren 7-9x schneller als die anderen
- verteiltes Rechnen dauert bei großen Datenmengen trotzdem lange
  - ~295 Tage bei 3,5 Millionen Subjekten, 50 Iterationen und 1000 Clustern

Vielen Dank!

Fork us on Github

github.com/robertpfeiffer/dbpedia-clustering