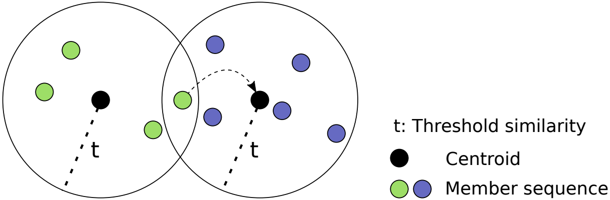
Bachelor's thesis defense

Nikolaj Dybdahl Rathcke

Background

- Inexpensive sequencing data
- Centroid-based clustering
- What does `klust` solve?



Distance metric

- Sequence alignment is expensive
- *k*-mer is cheap to compute
- The Manhattan distance
- Windows ensures that string of different length are still comparable
- K-DIST

k-mer	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
s1				2			1	1					1	2		
s2				2			1	1					2	2		1

K-DIST example with $k = 2$

Manhattan distance: 4

A T C T A T C G

T T A T C T A T C G

K-DIST example with $k = 2$

Manhattan distance: 2

	A	T	C	T	A	T	C	G	
T	T	A	T	C	T	A	T	C	G

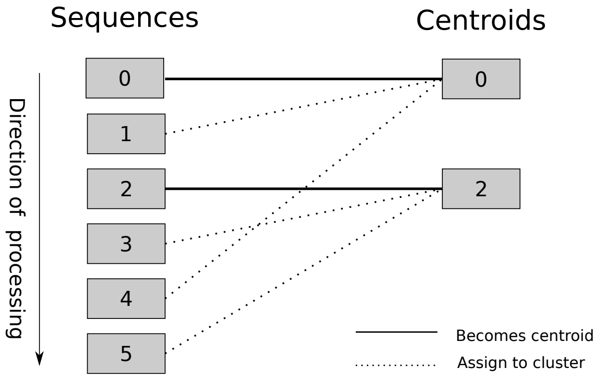
K-DIST example with $k = 2$

Manhattan distance: 0

		A	T	C	T	A	T	C	G
T	T	A	T	C	T	A	T	C	G

Clustering algorithm

- Greedy algorithm improves time complexity



Clustering algorithm

- Greedy algorithm improves time complexity
- Intersection criterion to quickly dismiss sequences that are not likely to belong to a cluster

Intersection criterion

$$|K(s) \cap K(c)| \geq |K(c)| \cdot id$$

k-mer	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
s1				2			1	1					1	2		
s2				2			1	1					2	2		1

Clustering algorithm

- Greedy algorithm improves time complexity
- Intersection criterion to quickly dismiss sequences that are not likely to belong to a cluster

Intersection criterion

$$|K(s) \cap K(c)| \geq |K(c)| \cdot id$$

k-mer	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
K(s1)				1			1	1					1	1		
K(s2)				1			1	1					1	1		1

Clustering algorithm

- Greedy algorithm improves time complexity
- Intersection criteria to quickly dismiss sequences that are not likely to belong to a cluster
- Ordering centroids can improve performance

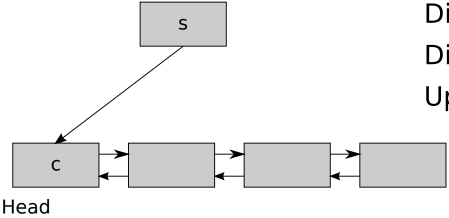
Clustering algorithm

- Greedy algorithm improves time complexity
- Intersection criteria to quickly dismiss sequences that are not likely to belong to a cluster
- Ordering centroids can improve performance
- The centroid structure

Clustering algorithm

- Greedy algorithm improves time complexity
- Intersection criteria to quickly dismiss sequences that are not likely to belong to a cluster
- Ordering centroids can improve performance
- The centroid structure
- K-Clust

K-CLUST



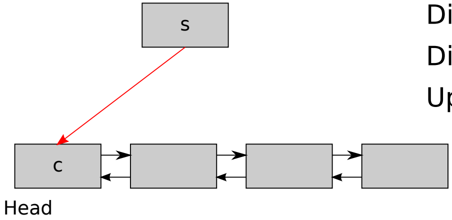
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



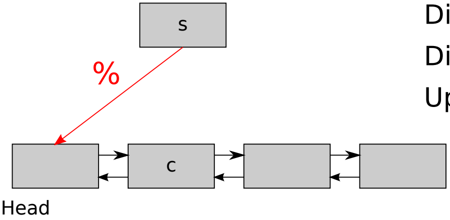
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



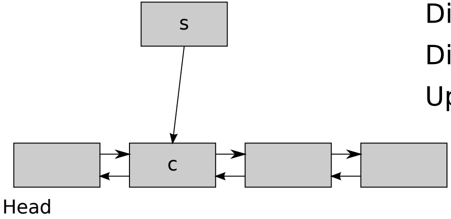
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-CLUST



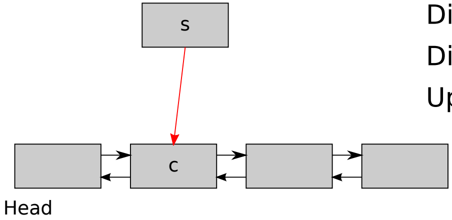
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



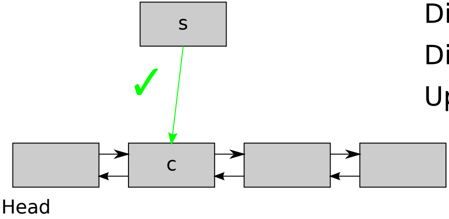
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



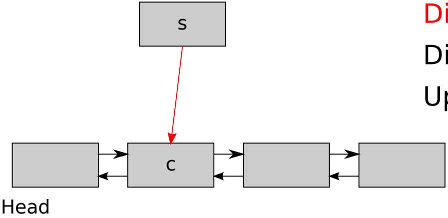
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



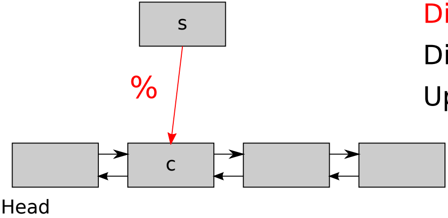
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



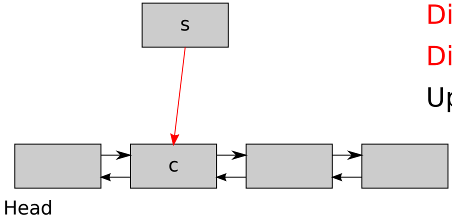
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



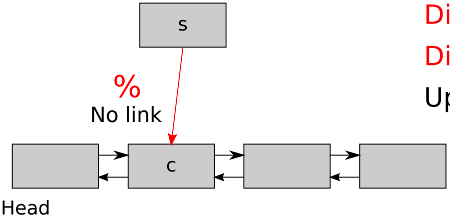
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



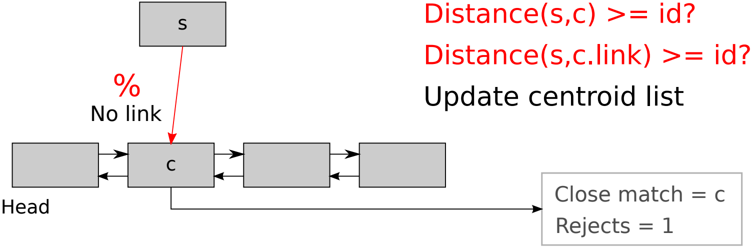
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-CLUST



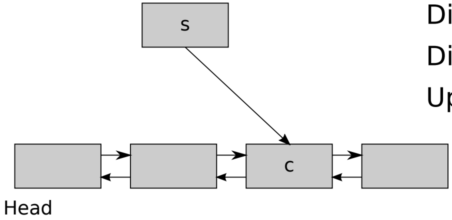
Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

K-Clust



Intersection criteria

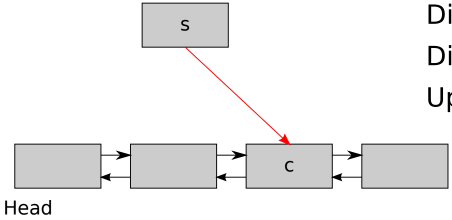
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-CLUST



Intersection criteria

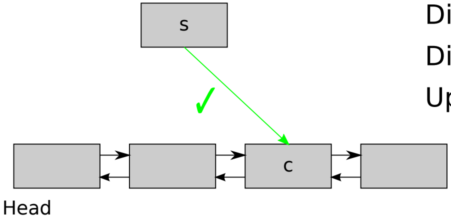
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

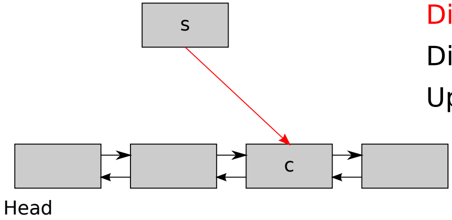
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

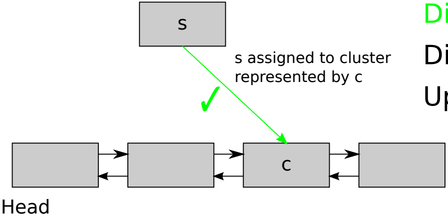
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

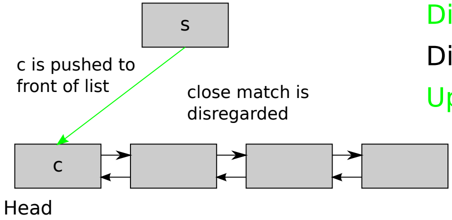
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

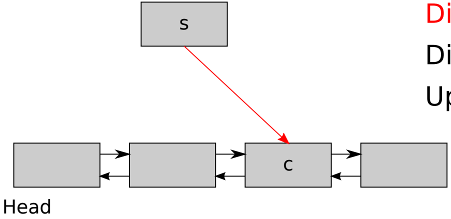
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

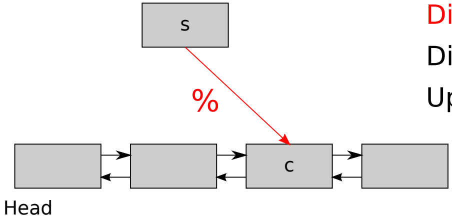
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

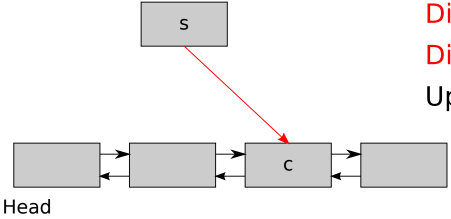
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

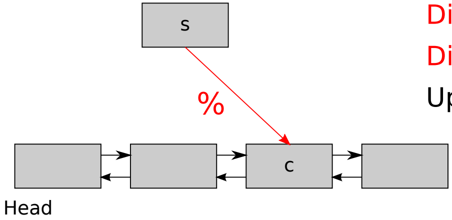
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-CLUST



Intersection criteria

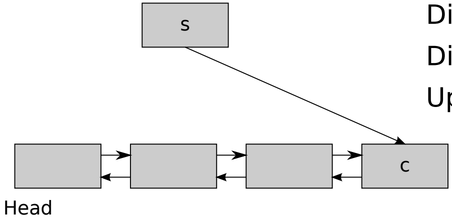
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-CLUST



Intersection criteria

Distance(s,c) >= id?

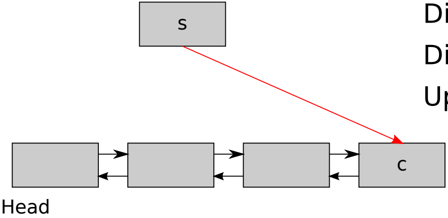
Distance(s,c.link) >= id?

Update centroid list

Close match set

Rejects = 1

K-Clust



Intersection criteria

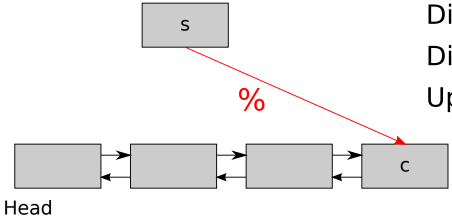
Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

K-Clust



Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list

Close match set
Rejects = 1

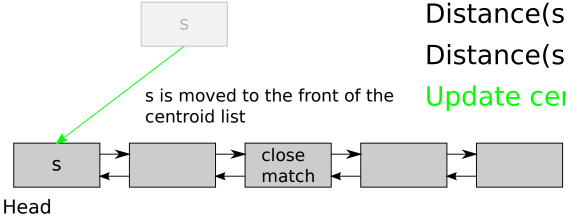
K-Clust

Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list



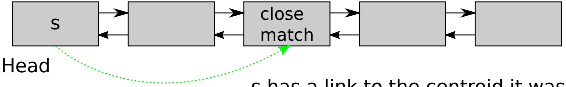
K-Clust

Intersection criteria

Distance(s,c) >= id?

Distance(s,c.link) >= id?

Update centroid list



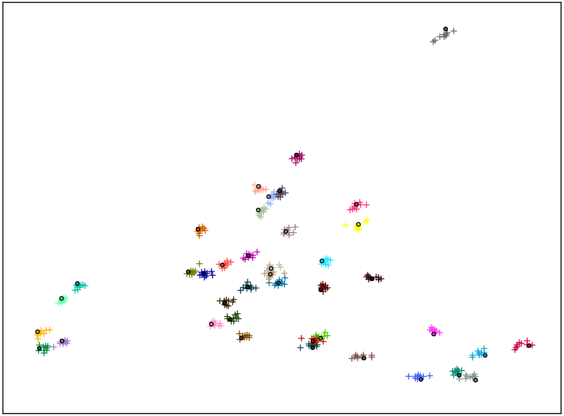
s has a link to the centroid it was close to matching

Results

- Evaluation of K-CLUST with synthetic data
- Evaluation of `klust` on real data

Multi-dimensional scaling of clustering output from SILVA

Clustering on 40 very different sequences with 9 copies of each that has been altered.



Comparison of `klust` and `USEARCH` on `RDP`

Clustering algorithm	Time (sec.)	Throughput (seqs./sec.)	Clusters	Cluster sizes	Max memory
K-CLUST, $k = 5$, $id = 0.85$, $m = 8$, incr. sort	5420.8	557.10	220 982	Max. 100 832 Avg. 13.67 Min. 1	≈ 2031 MB
K-CLUST, $k = 5$, $id = 0.9$, $m = 8$, incr. sort	11 948.7	252.74	344 122	Max. 55 992 Avg. 8.78 Min. 1	≈ 2031 MB
`USEARCH`, $id = 0.95$, decr. sort `-cluster_smallmem`	6874.0	439.20	261 880	Max. 65 654 Avg. 11.50 Min. 1	≈ 1433 MB
`USEARCH`, $id = 0.97$, decr. sort `-cluster_smallmem`	11 980.0	252.00	471 982	Max. 56 279 Avg. 6.40 Min. 1	≈ 2560 MB

Future work

- Link optimization effectively using max rejects
- Optimizing distance metric to better recognize mutations
- A better merge strategy for a solution using parallelization