PROJECT
HIERARCHICAL CLUSTERING

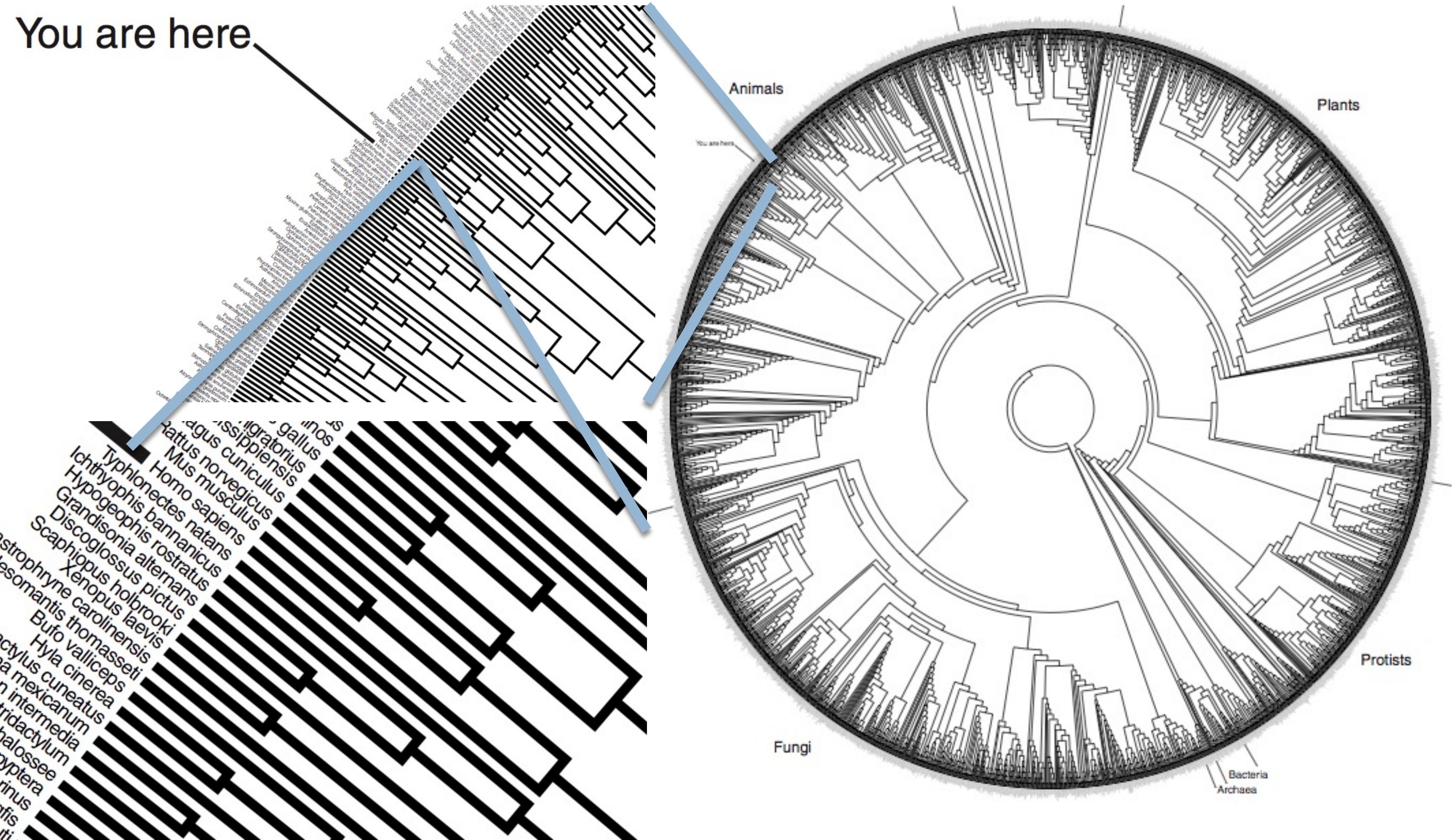
Clustering

- Cluster analysis
  - Gather objects into groups
- A Cluster is a subset of objects:
  - <u>Similar</u> to any other object in the *same* cluster
  - <u>Dissimilar</u> to any other object in a *different* cluster
- Clustering is an unsupervised classification task:
  - We don't know the classes in advance,
    every cluster can be interpreted as a class

Applications

- Detecting groups of similar users:
  - With the same purchasing patterns
  - With the same Web-site browsing patterns
  - With the same tastes
- Detecting similar object:
  - With similar properties
- Other applications:
  - As a stand-alone tool for data analysis
  - As a preprocessing step for other tasks/algorithm (outlier detection, summarization)
  - Clustering of documents related to the same topic
  - Classes of customers of an insurance company
  - Biology, phylogenetic trees

Phylogenetic trees



A good clustering algorithm should be:

- Scalable
- Able to deal with clusters with arbitrary shapes
- Able to handle noise and outliers
- Independent from input order
- Able to deal with highly dimensional data
- Able to exploit user constraints (or hints)
- Able to provide easily understandable results

Clustering algorithms:

- Partitional algorithms: Partition the objects into dis-joint sets. Usually iterative methods.
- Hierarchical algorithms: Creates an hierarchy/tree of objects, such that similar objects are “close” in the tree.
- Density-based: Number of objects in a given region of the space.
- Model-based: Statistical methods: assumes a given distribution function of the data, and finds the best fitting to the data.

The input

- **Data Matrix**
  - *n* objects with
    p attributes/dimensions
- **Distance Matrix**
  - *d(i, j)* is the distance between i and *j*
  - *d(i, j)* = *0*
    identical / non distinguishable objects

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

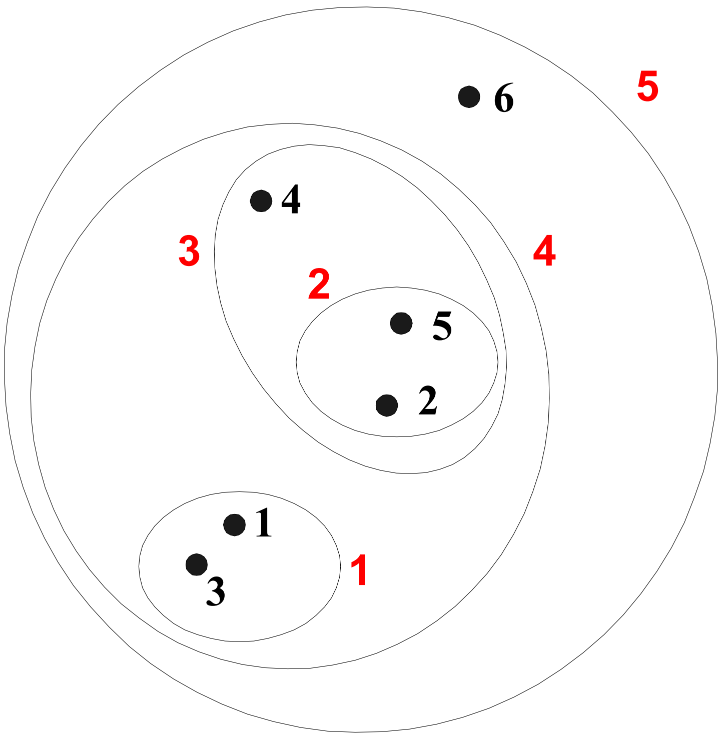
$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

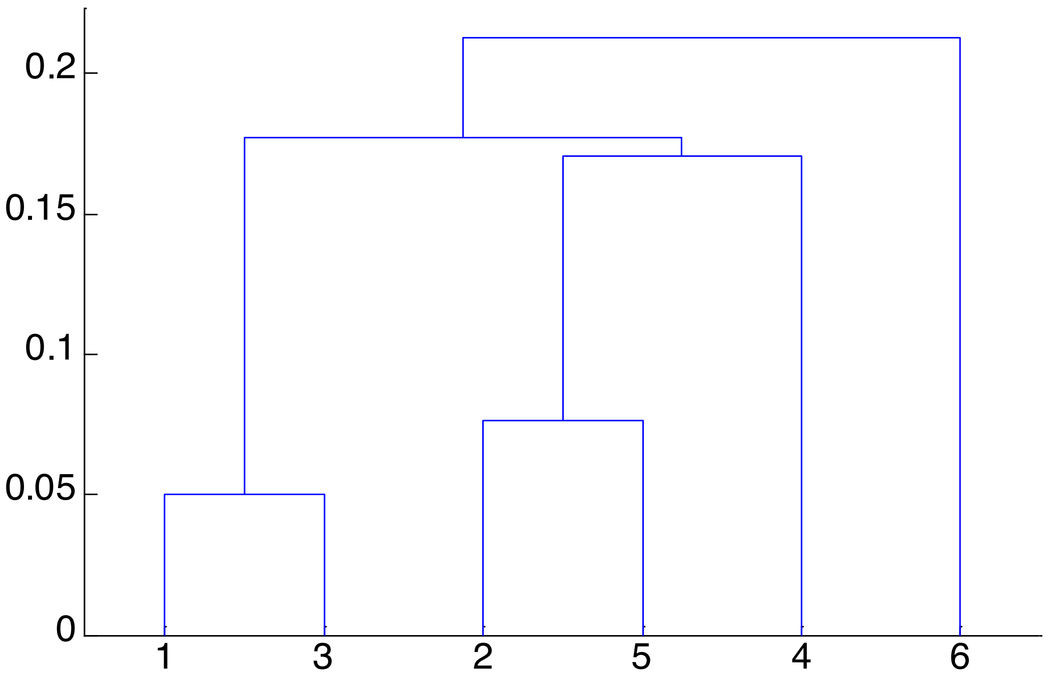
Hierarchical Agglomerative Algorithm

1. Compute similarity matrix
2. Every point is a cluster
3. <u>Repeat</u>
4. Join the two closest clusters
5. Update the distance matrix
6. <u>Until</u> one cluster is left

- The key operation is how to compute distance between clusters
  - Algorithms differ on the distance function they use

Hierarchical Clustering

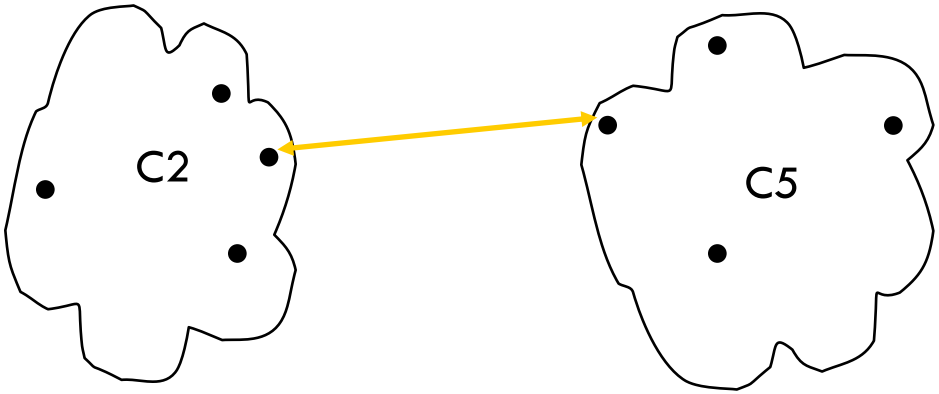




Hierarchical Clustering **Dendrogram**

Min (Single Linkage)

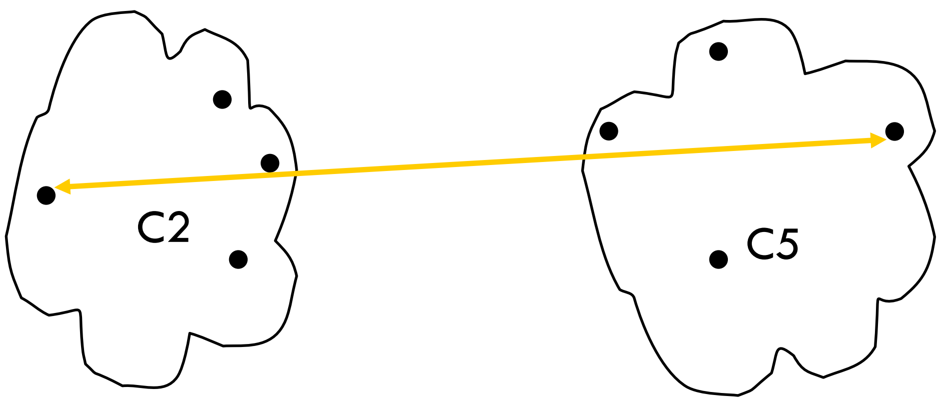
- The distance between two clusters is the distance of the closest points



- Cons:
  - it overestimates similarity, and it may produce chaining

Max (Complete Linkage)

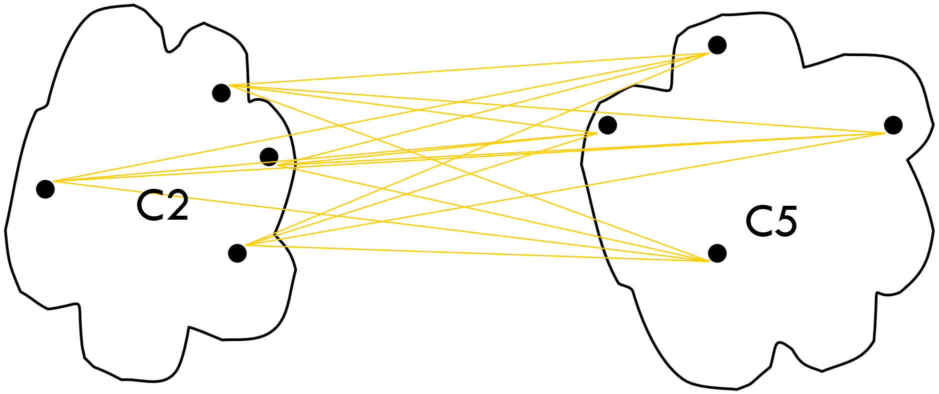
- The distance between two clusters is given by the distance of the farthest points



- Cons:
  - It underestimates similarity, favors globular clusters

Average

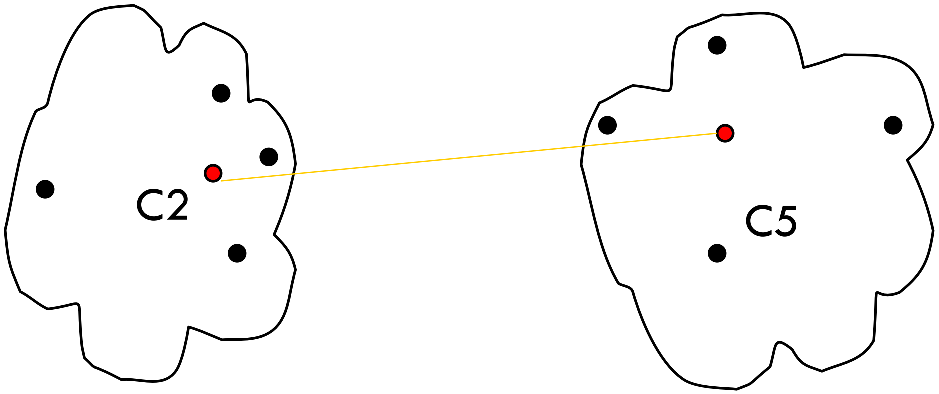
- The distance between two clusters is given by the average distance between every couple of points (divided by the product of the cluster sizes)



- <u>Pros</u>:
  - In the middle of the other two

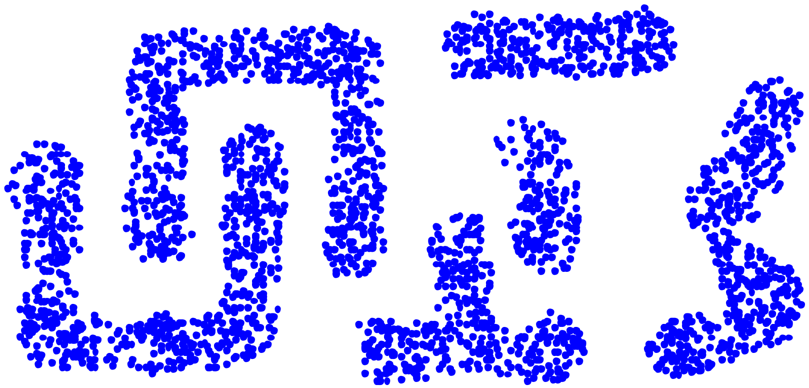
Centroid based

- The distance between two clusters is given by the distance between their centroids (medoids)

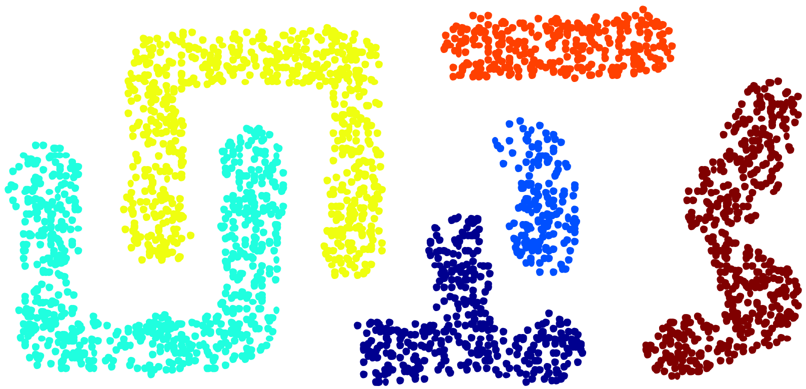


- Pros:
  - Fast !

Strength of MIN



Original Points

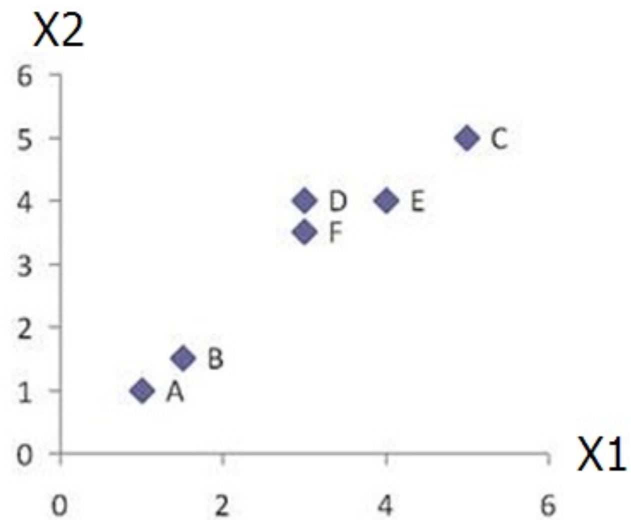


Six Clusters

- **Can handle non-elliptical shapes**

Single linkage clustering Example

- Problem: clustering analysis with agglomerative algorithm



	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5

data matrix

$$d_{AB} = \left((1-1.5)^2 + (1-1.5)^2 \right)^{\frac{1}{2}} = \sqrt{\tfrac{1}{2}} = 0.7071$$

$$d_{DF} = \left((3-3)^2 + (4-3.5)^2 \right)^{\frac{1}{2}} = 0.5$$

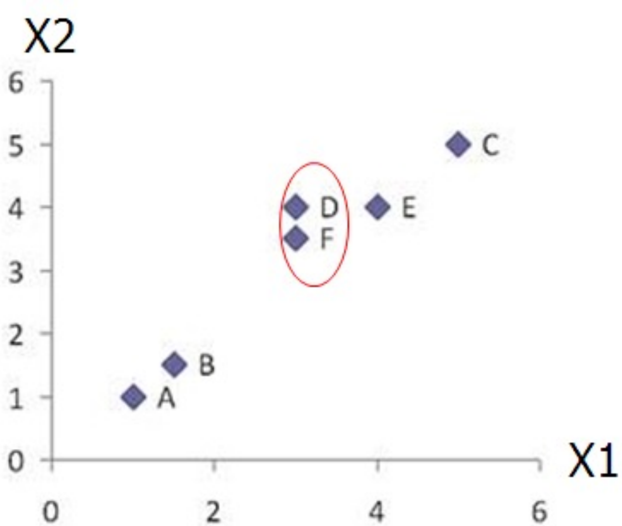
Euclidean distance

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

distance matrix

Single linkage clustering Example

- Merge two closest clusters (iteration 1)



Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

Single linkage clustering Example

- Update distance matrix (iteration 1)

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

$$d_{(D,F)\mapsto A} = \min\left(d_{DA}, d_{FA}\right) = \min\left(3.61, 3.20\right) = 3.20$$

$$d_{(D,F)\mapsto B} = \min\left(d_{DB}, d_{FB}\right) = \min\left(2.92, 2.50\right) = 2.50$$

$$d_{(D,F)\mapsto C} = \min\left(d_{DC}, d_{FC}\right) = \min\left(2.24, 2.50\right) = 2.24$$

$$d_{E\to(D,F)} = \min\left(d_{ED}, d_{EF}\right) = \min\left(1.00, 1.12\right) = 1.00$$

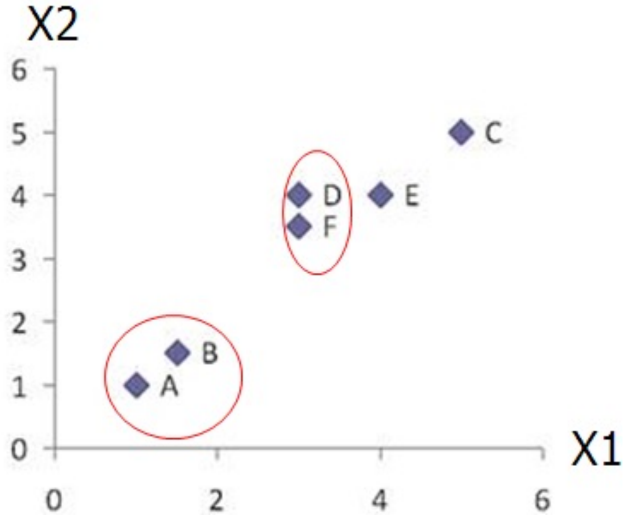
Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

Single linkage clustering Example

- Merge two closest clusters (iteration 2)



Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	**?**	0	2.24	1.41
(D, F)	**?**	2.24	0	1.00
E	**?**	1.41	1.00	0

Single linkage clustering Example

- Update distance matrix (iteration 2)

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

$$d_{C\to(A,B)} = \min(d_{CA}, d_{CB}) = \min(5.66, 4.95) = 4.95$$

$$d_{(D,F)\mapsto(A,B)} = \min(d_{DA}, d_{DB}, d_{FA}, d_{FB}) = \min(3.61, 2.92, 3.20, 2.50) = 2.50$$

$$d_{E\to(A,B)} = \min(d_{EA}, d_{EB}) = \min(4.24, 3.54) = 3.54$$

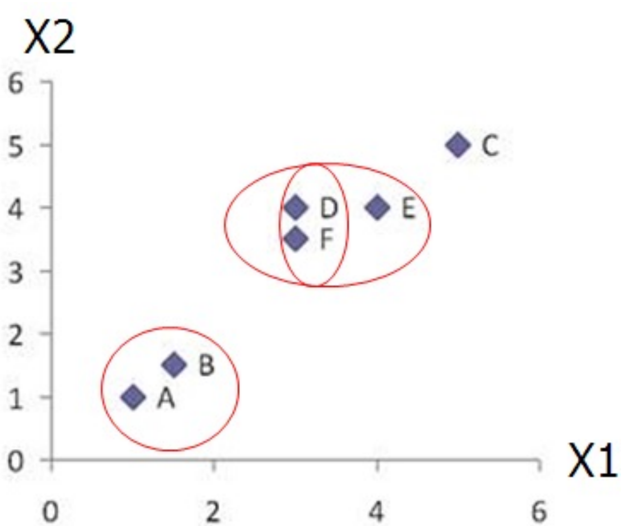
Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	**?**	0	2.24	1.41
(D, F)	**?**	2.24	0	1.00
E	**?**	1.41	1.00	0

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	**4.95**	0	2.24	1.41
(D, F)	**2.50**	2.24	0	1.00
E	**3.54**	1.41	1.00	0

Single linkage clustering Example

- Merge two closest clusters/update distance matrix (iteration 3)



Min Distance (Single Linkage)

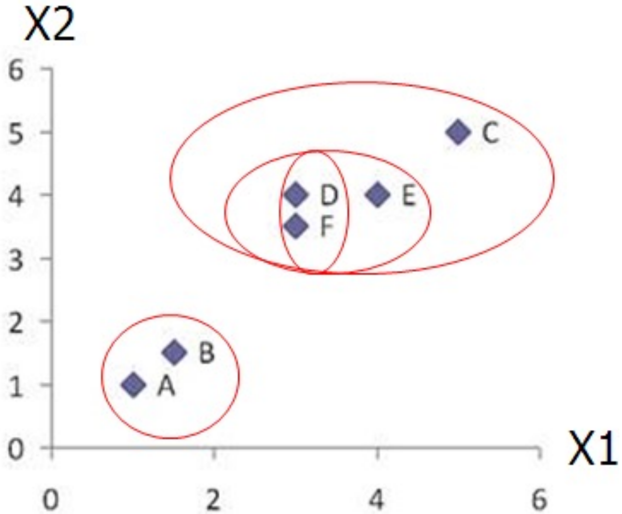
Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	**4.95**	0	2.24	1.41
(D, F)	**2.50**	2.24	0	1.00
E	**3.54**	1.41	1.00	0

Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	**2.50**	**1.41**	0.00

Single linkage clustering Example

- Merge two closest clusters/update distance matrix (iteration 4)



Min Distance (Single Linkage)

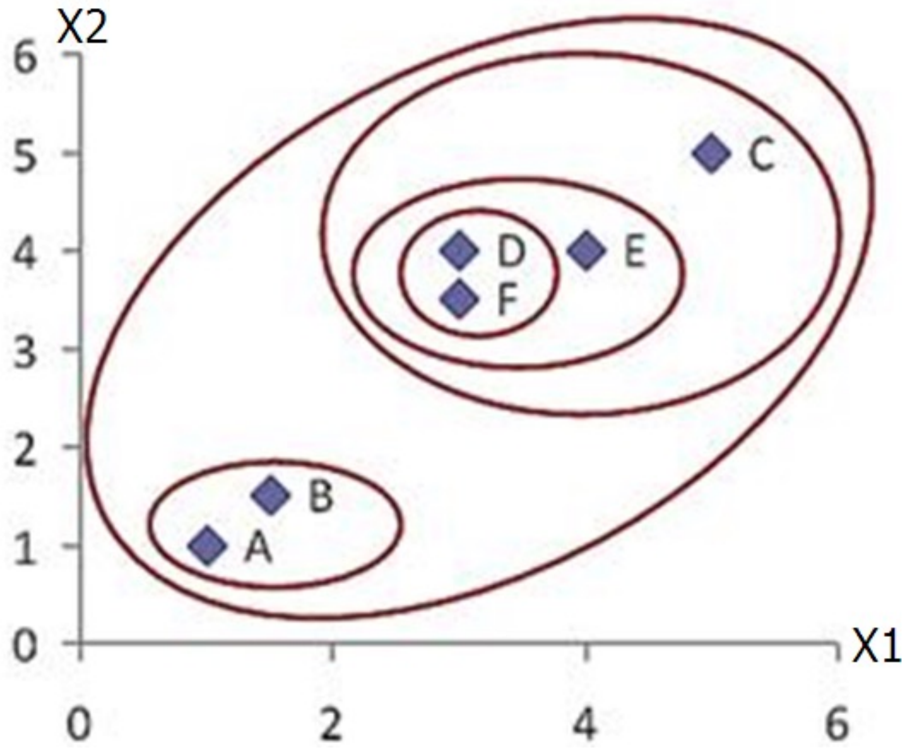
Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	**2.50**	**1.41**	0.00

Min Distance (Single Linkage)

Dist	(A,B)	((D, F), E),C
(A,B)	0.00	2.50
((D, F), E),C	**2.50**	0.00

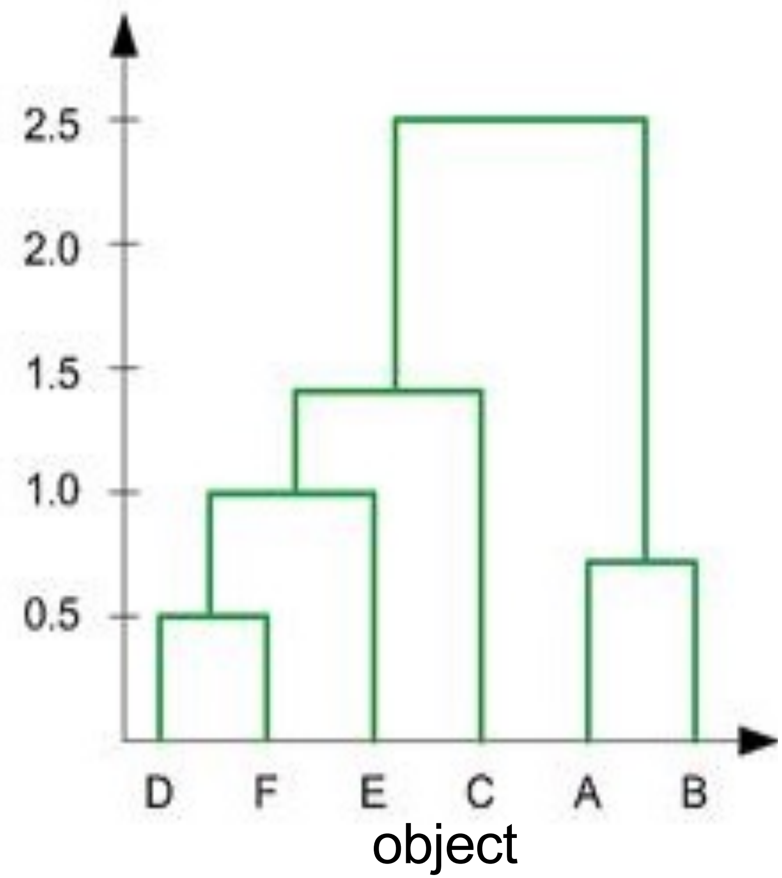
Single linkage clustering Example

- Final result (meeting termination condition)



Single linkage clustering Example

- Dendrogram tree representation



1. In the beginning we have 6 clusters: A, B, C, D, E and F
2. We merge clusters D and F into cluster (D, F) at distance 0.50
3. We merge cluster A and cluster B into (A, B) at distance 0.71
4. We merge clusters E and (D, F) into ((D, F), E) at distance 1.00
5. We merge clusters ((D, F), E) and C into (((D, F), E), C) at distance 1.41
6. We merge clusters (((D, F), E), C) and (A, B) into ((((D, F), E), C), (A, B)) at distance 2.50
7. The last cluster contain all the objects, thus conclude the computation

Complexity analysis-general single linkage clustering

- N: the number of data points

- Space complexity
  - $O(N^2)$: Requires to store the distance matrix

- Time complexity
  - In most cases, $O(N^3)$:
    - There are N steps and at each step the size, N^2, distance matrix must be updated and searched

SLINK

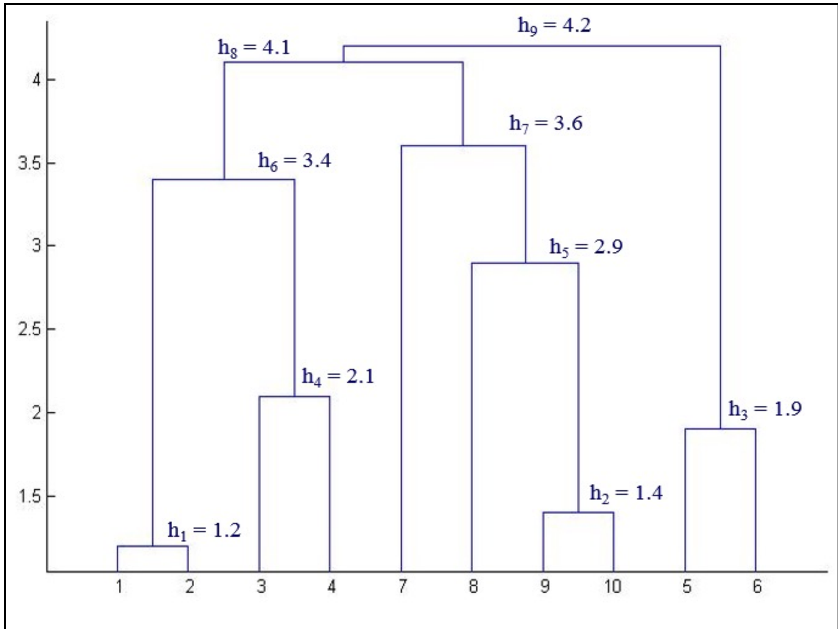
- SLINK
  - Pointer representation(The pair of π , λ functions)
    - Used to improve performance of hierarchical clustering
    - Allows a new object to be inserted in an efficient way
    - A pair of functions which contain information on a dendrogram
- Main idea of SLINK
  - Use Pointer representation(π, λ) to specify N objects
  - Only access the part-row values (M)

SLINK-Pointer representation

- Pointer representation
  - $\pi(i)$: ($i \in [1, N] \rightarrow [1, N]$)
    - The last object of the clusters which is merged with *i*
    - *(sort of cluster id defined as the largest id in the cluster)*
  - $\lambda(i)$: ($i \in [1, N] \rightarrow [0, \infty]$)
    - The smallest distance at which *i* is no longer the last (i.e., the highest numbered) object in its cluster

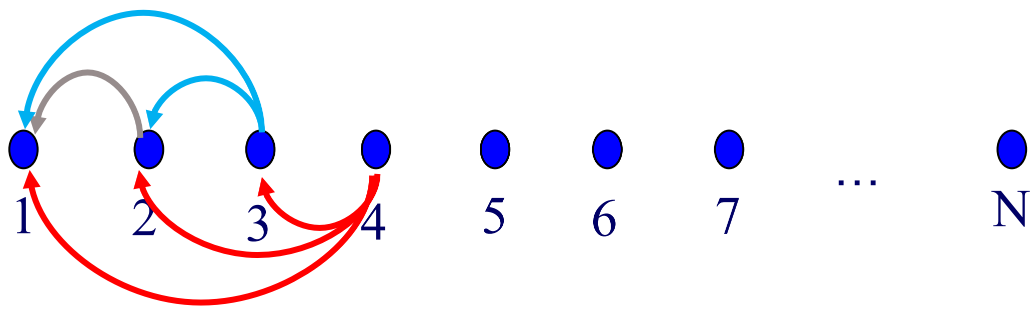
Example: (i : object processing sequence)

i	1	2	3	4	5	6	7	8	9	10
$\pi[i]$	2	4	4	10	6	10	10	10	10	10
$\lambda[i]$	1.2	3.4	2.1	4.1	1.9	4.2	3.6	2.9	1.4	∞



SLINK-Part-row values

- Part-row values
  - $M(j)$ $(j \in [1, N\text{-}1] \rightarrow [0, \infty], j < i)$
  - Example
    - (when i = 2): 2-1(M(1))
    - (when i = 3): 3-1(M[1)), 3-2(M(2))
    - (when i = 4): 4-1(M(1)), 4-2(M(2)), 4-3(M(3))



Slink – pseudocode

1. Set $\Pi(n+1)$ to $n+1$, $\Lambda(n+1)$ to ∞
2. Set $M(i)$ to $d(i, n+1)$ for $i = 1, \ldots, n$
3. For i increasing from 1 to n

    if $\Lambda(i) \geqslant M(i)$

        set $M(\Pi(i))$ to min $\{M(\Pi(i)), \Lambda(i)\}$

        set $\Lambda(i)$ to $M(i)$

        set $\Pi(i)$ to $n+1$

    if $\Lambda(i) < M(i)$

        set $M(\Pi(i))$ to min $\{M(\Pi(i)), M(i)\}$

4. For i increasing from 1 to n

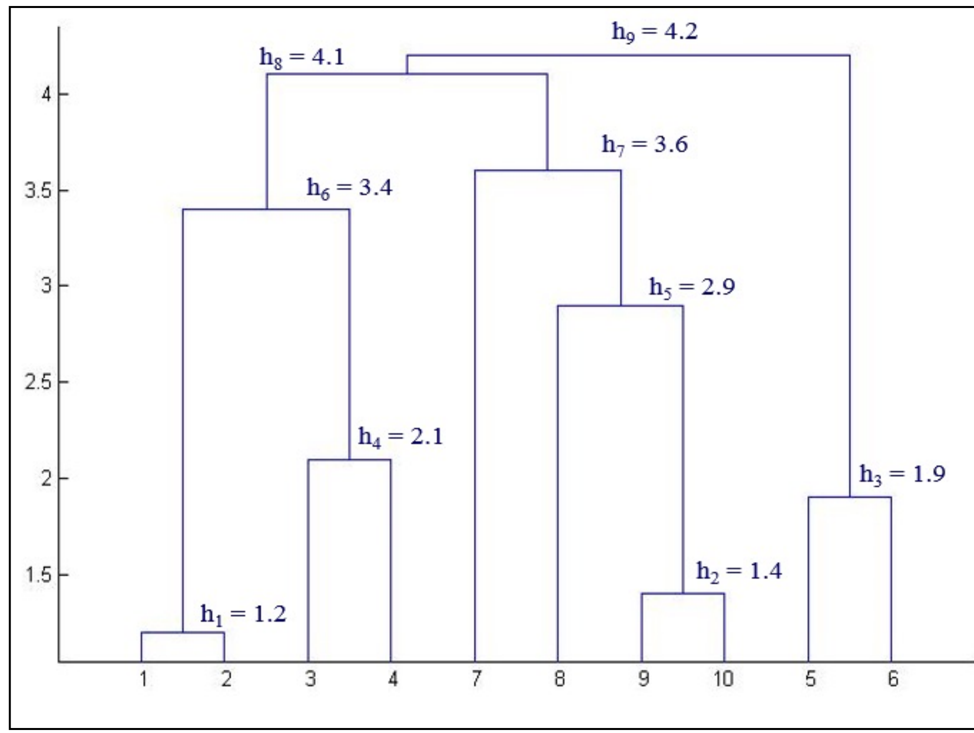
    if $\Lambda(i) \geqslant \Lambda(\Pi(i))$

        set $\Pi(i)$ to $n+1$

SLINK-Example

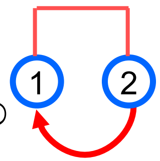
- Here, D is a distance matrix and the values are **just** used to show the part-row values in the following example. In SLINK, there is no need to calculate D.

$$
\begin{array}{cc}
 & \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\
D = & \begin{pmatrix}
0 & 1.2 & 5 & 5 & 4.2 & 7 & 9 & 7.6 & 11 & 4.3 \\
1.2 & 0 & 3.4 & 4.1 & 5 & 6 & 4.1 & 6.4 & 5.3 & 4.5 \\
5 & 3.4 & 0 & 2.1 & 6 & 6.2 & 4.6 & 9 & 11.3 & 22 \\
5 & 4.1 & 2.1 & 0 & 11 & 5 & 13 & 4.1 & 4.3 & 5.5 \\
4.2 & 5 & 6 & 11 & 0 & 1.9 & 7 & 9 & 5.5 & 4.3 \\
7 & 6 & 6.2 & 5 & 1.9 & 0 & 7.5 & 5.6 & 6.3 & 4.5 \\
9 & 4.1 & 4.6 & 13 & 7 & 7.5 & 0 & 3.6 & 8 & 10 \\
7.6 & 6.4 & 9 & 4.1 & 9 & 5.6 & 3.6 & 0 & 4.9 & 2.9 \\
11 & 5.3 & 11.3 & 4.3 & 5.5 & 6.3 & 8 & 4.9 & 0 & 1.4 \\
4.3 & 4.5 & 22 & 5.5 & 4.3 & 4.5 & 10 & 2.9 & 1.4 & 0
\end{pmatrix}
\end{array}
$$

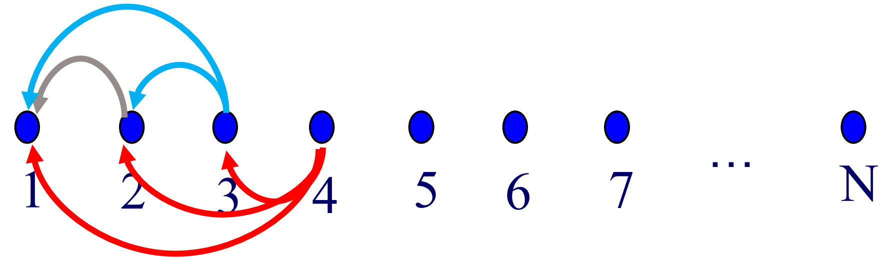


SLINK-Example

- 1st object:
  - $\pi(1) = 1, \lambda(1) = \infty$ 1
- 2nd object:
  - Initialization:
    - $\pi(2) = 2, \lambda(2) = \infty$
    - $M(1) = 1.2$

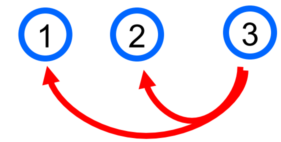


  - Update:
    - 1 case: $\lambda(1) > M(1) \rightarrow (\infty > 1.2)$
      - $M(\pi(1)) = \min(M(\pi(1)), \lambda(1))$
        $M(1) = \min(1.2, \infty) = 1.2$
      - $\lambda(1) = M(1) = 1.2, \pi(1) = 2$
  - Final status:
    - $\pi(1) = 2, \lambda(1) = 1.2$
    - $\pi(2) = 2, \lambda(2) = \infty$

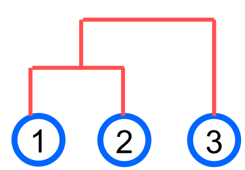


1. Set $\Pi(n+1)$ to $n+1$, $\Lambda(n+1)$ to ∞
2. Set $M(i)$ to $d(i, n+1)$ for $i = 1, \ldots, n$
3. For i increasing from 1 to n
   if $\Lambda(i) \geqslant M(i)$
   set $M(\Pi(i))$ to min $\{M(\Pi(i)), \Lambda(i)\}$
   set $\Lambda(i)$ to $M(i)$
   set $\Pi(i)$ to $n+1$
   if $\Lambda(i) < M(i)$
   set $M(\Pi(i))$ to min $\{M(\Pi(i)), M(i)\}$
4. For i increasing from 1 to n
   if $\Lambda(i) \geqslant \Lambda(\Pi(i))$
   set $\Pi(i)$ to $n+1$

SLINK-Example

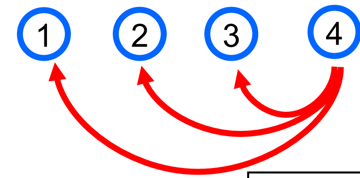


- 3^{rd} object:
  - Initialization:
    - $\pi(3) = 3, \lambda(3) = \infty$
    - $M(1) = 5, M(2) = 3.4$
  - Update:
    - 1 Case: $\lambda(1) < M(1) \rightarrow (1.2 < 5)$
      - $M(\pi(1)) = \min(M(\pi(1)), M(1))$
$M(2) \quad = \min(M(2), M(1)) = 3.4$
    - 2 Case: $\lambda(2) > M(2) \rightarrow (\infty > 3.4)$
      - $M(\pi(2)) = \min(M(\pi(2)), \lambda(2))$
$M(2) \quad = \min(M(2), \infty) = 3.4$
      - $\lambda(2) = M(2) = 3.4, \pi(2) = 3$
  - Final status:
    - $\pi(1) = 2, \lambda(1) = 1.2$
    - $\pi(2) = 3, \lambda(2) = 3.4$
    - $\pi(3) = 3, \lambda(3) = \infty$



1. Set $\Pi(n+1)$ to $n+1$, $\Lambda(n+1)$ to ∞
2. Set $M(i)$ to $d(i, n+1)$ for $i = 1, \ldots, n$
3. For i increasing from 1 to n
   - if $\Lambda(i) \geqslant M(i)$
     - set $M(\Pi(i))$ to min $\{M(\Pi(i)), \Lambda(i)\}$
     - set $\Lambda(i)$ to $M(i)$
     - set $\Pi(i)$ to $n+1$
   - if $\Lambda(i) < M(i)$
     - set $M(\Pi(i))$ to min $\{M(\Pi(i)), M(i)\}$
4. For i increasing from 1 to n
   - if $\Lambda(i) \geqslant \Lambda(\Pi(i))$
     - set $\Pi(i)$ to $n+1$

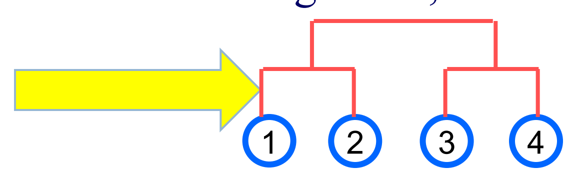
SLINK-Example



- **4th object:**
  - Initialization:
    - $\pi(4) = 4, \lambda(4) = \infty$
    - $M(1) = 5, M(2) = 4.1, M(3) = 2.1$
  - Update:
    - 1 Case: $\lambda(1) < M(1) \rightarrow (1.2 < 5)$
      - $M(\pi(1)) = \min(M(\pi(1)), M(1))$
        $M(2) \quad = \min(M(2), M(1)) = \min(4.1, 5) = 4.1$
    - 2 Case: $\lambda(2) < M(2) \rightarrow (3.4 < 4.1)$
      - $M(\pi(2)) = \min(M(\pi(2)), M(2)) \ = \min(M(3), M(2))$
        $M(3) \quad = \min(2.1, 4.1) = 2.1$
    - 3 Case: $\lambda(3) > M(3) \rightarrow (\infty > 2.1)$
      - $M(\pi(3)) = \min(M(\pi(3)), \lambda(3)) = \min(M(3), \lambda(3))$
        $M(3) \quad = \min(2.1, \infty) = 2.1$
      - $\lambda(3) = M(3) = 2.1, \pi(3) = 4$
  - After case update status:
    - $\pi(1) = 2, \lambda(1) = 1.2$
    - $\pi(2) = 3, \lambda(2) = 3.4$
    - $\pi(3) = 4, \lambda(3) = 2.1$
    - $\pi(4) = 4, \lambda(4) = \infty$

1. Set $\Pi(n+1)$ to $n+1$, $\Lambda(n+1)$ to ∞
2. Set $M(i)$ to $d(i, n+1)$ for $i = 1, \ldots, n$
3. For i increasing from 1 to n
   if $\Lambda(i) \geqslant M(i)$
     set $M(\Pi(i))$ to min $\{M(\Pi(i)), \Lambda(i)\}$
     set $\Lambda(i)$ to $M(i)$
     set $\Pi(i)$ to $n+1$
   if $\Lambda(i) < M(i)$
     set $M(\Pi(i))$ to min $\{M(\Pi(i)), M(i)\}$
4. For i increasing from 1 to n
   if $\Lambda(i) \geqslant \Lambda(\Pi(i))$
     set $\Pi(i)$ to $n+1$

Here, assume we only have 4 objects, then after cluster rearrangement,



$\pi(1) = 2, \lambda(1) = 1.2$
$\pi(2) = 4, \lambda(2) = 3.4$
$\pi(3) = 4, \lambda(3) = 2.1$
$\pi(4) = 4, \lambda(4) = \infty$

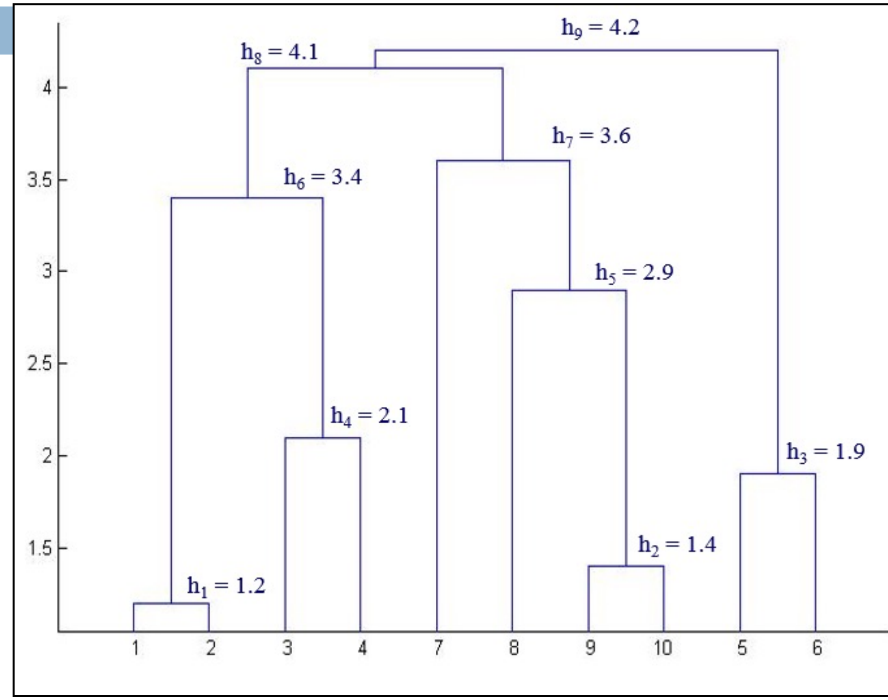
SLINK-Hierarchy Extraction

Example:

i	1	2	3	4	5	6	7	8	9	10
$\pi[i]$	2	4	4	10	6	10	10	10	10	10
$\lambda[i]$	1.2	3.4	2.1	4.1	1.9	4.2	3.6	2.9	1.4	∞

Order by λ

m	1	2	3	4	5	6	7	8	9	10
i	1	9	5	3	8	2	7	4	6	10
$\pi[i]$	2	10	6	4	10	4	10	10	10	10
$\lambda[i]$	1.2	1.4	1.9	2.1	2.9	3.4	3.6	4.1	4.2	∞



When m=1: ① + ②

When m=2: 10 not exists in [2], so ⑨ + ⑩

When m=3: 6 not exists in [2, 10], so ⑤ + ⑥

When m=4: 4 not exists in [2, 10, 6], so ③ + ④

When m=5: 10 exists in [2,10,6,4], so (⑨ + ⑩) + ⑧

when m=6: 4 exists in [2,10,6,4,10], so (③ + ④) + (① + ②)

…

Complexity analysis for SLINK

- SLINK:
  - Time complexity: $O(N^2)$ to find π, λ
  - Space complexity: in fact 3N (π, λ, M)

Item	General single linkage	Slink
Time complexity	$O(N^3)$	$O(N^2)$
Space complexity	$O(N^2 + \text{overhead})$	O(3N)

Yet to be Investigated

- *How to make this parallel ?*

HPC Lab Project

- Single Linkage Hierarchical Clustering
- To Be Delivered:
  - Sequential implementation of the algorithm discussed
  - Parallel implementation
    - Multi-threaded or Distributed or GPU
  - Report discussing performance figures of the proposed parallel implementation
    - Varying threads/cores/processors
    - Varying parallelism strategy…
- Score:
  - up to 3 additional points
- Bonus:
  - SIMD Vectorization
  - Cache analysis

HPC Lab Project

- The report should be short, ~5 pages
- I expect to find the following flow:
  - I attacked the problem with strategy A
    - Data layout and parallel decomposition
  - I expected to find these results
  - Experimental results are different
  - I tried to understand why
  - This lead me to strategy B
  - (… repeat …)

HPC Lab Project

- You may deliver your code and report via moodle, a link to your git repo for the source is ok, 3 days before the exam date
- C++ coding guidelines:
  - https://lefticus.gitbooks.io/cpp-best-practices/content/03-Style.html
- Markdown can be used for the report
  - https://bitbucket.org/tutorials/markdowndemo
- English writing:
  - https://faculty.washington.edu/heagerty/Courses/b572/public/StrunkWhite.pdf
  - https://www.publishingcampus.elsevier.com/websites/elsevier_publishingcampus/files/Skills%20training/Elements_of_Style.pdf
  - http://services.unimelb.edu.au/__data/assets/pdf_file/0009/471294/Using_tenses_in_scientific_writing_Update_051112.pdf
  - https://pingpong.chalmers.se/public/pp/public_courses/course08583/published/1510227352918/resourceId/4156227/content/Zobel%20-%20Writing%20for%20computer%20science%203rd%20edition.pdf

References

- Introduction to Data Mining (Second Edition), Kumar et al, Chap. 7. Cluster Analysis: Basic Concepts and Algorithm.
- Slides: https://github.com/jackyust/SLINK_CLINK/blob/master/SLINK_CLINK.ppt
- Original paper in Moodle

The End!