CS 6824: Modules

T. M. Murali

February 22, 27, and March 1, 2018
Student Presentations

- Wiring cost optimisation and cost-efficiency trade-offs
  Scott Clark, Omar Faruqi, Cameron Rader

- Degree-based measures of centrality
  Branden Arnold, Mostafa Elmary, Madison Wilkins

- Betweenness centrality
  Aidan Barton, Jose Canahui, Jordan Kuhn

- Rich clubs
  Team Valkyrie: Shane Davies, Heidi Tubbs, Tianna Woodson

- Overlapping modules
  Team Wildcards: Kavin Aravind, Tom Evans, Rishi Pulluri

- Growth connectomics: generative models for brain networks
  William Edmisten, Ethan Gallagher, Sophia Sheikl
Schedule of Meetings and Presentations

- Each group meets me for 60–90 minutes one week before practice presentation.
  - Goal is to discuss details of presentation.
  - Come prepared: read your section, find relevant papers, have a talk outline, ask me questions.
- Each group meets me for 60–90 minutes about one week before actual presentation.
- Office hours for these meetings: 10am-12pm on Tuesdays and Thursdays, after Spring break, and by appointment.
Schedule of Meetings and Presentations

- Each group meets me for 60–90 minutes one week before practice presentation.
  - Goal is to discuss details of presentation.
  - Come prepared: read your section, find relevant papers, have a talk outline, ask me questions.
- Each group meets me for 60–90 minutes about one week before actual presentation.
- Office hours for these meetings: 10am-12pm on Tuesdays and Thursdays, after Spring break, and by appointment.

<table>
<thead>
<tr>
<th>Topic</th>
<th>First meeting</th>
<th>Practice</th>
<th>Second meeting</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiring cost optimisation</td>
<td>Mar 13</td>
<td>Mar 20</td>
<td>Apr 3</td>
<td>Apr 10</td>
</tr>
<tr>
<td>Degree-based centrality</td>
<td>TBD</td>
<td>Mar 22</td>
<td>Apr 5</td>
<td>Apr 12</td>
</tr>
<tr>
<td>Betweenness</td>
<td>Mar 20</td>
<td>Mar 27</td>
<td>Apr 10</td>
<td>Apr 17</td>
</tr>
<tr>
<td>Rich clubs</td>
<td>Mar 22</td>
<td>Mar 29</td>
<td>Apr 12</td>
<td>Apr 19</td>
</tr>
<tr>
<td>Overlapping Modules</td>
<td>Mar 27</td>
<td>Apr 3</td>
<td>Apr 17</td>
<td>Apr 24</td>
</tr>
<tr>
<td>Growth connectomics</td>
<td>Mar 29</td>
<td>Apr 5</td>
<td>Apr 19</td>
<td>Apr 26</td>
</tr>
</tbody>
</table>
Plan after Spring Break

- Invited presentation by Heidi Theussenn from Smith Career Center (March 13)
- No class on March 15
- Practice presentations (March 20 to April 5)
- Presentations (April 10 to Apr 26)
- No class on May 1
Summary of Course Thus Far

- Clustering coefficient is a local measure of graph density.
- Small world property captures global features of graph density.
Summary of Course Thus Far

- Clustering coefficient is a local measure of graph density.
- Small world property captures global features of graph density.

Are there intermediate notions of graph density?

- We have already considered components, shortest paths, cliques, and cores.
- We have also seen two specific types of modules: cliques and $k$-cores.
Why Modules?

Why should (brain) networks be modular?
Do modules exist in brain networks?
How do we define modules and find them?

No, because all nodes have roughly the same degree.
No, although other small-world networks can contain modules.
But the brain is indeed modular: organ, hemispheres, coarse divisions, lobes, cytoarchitectural areas, nuclei, etc.

Modularity and hierarchical organisation offer several advantages: evolvability, flexibility, adaptability, and complexity.
Why Modules?

Why should (brain) networks be modular?
Do modules exist in brain networks?
How do we define modules and find them?

- Do E-R graphs contain modules?
Why Modules?

Why should (brain) networks be modular?
Do modules exist in brain networks?
How do we define modules and find them?

- Do E-R graphs contain modules? No, because all nodes have roughly the same degree.
Why Modules?

Why should (brain) networks be modular?
Do modules exist in brain networks?
How do we define modules and find them?

- Do E-R graphs contain modules? No, because all nodes have roughly the same degree.
- Do W-S graphs contain modules?
Why Modules?

Why should (brain) networks be modular?
Do modules exist in brain networks?
How do we define modules and find them?

- Do E-R graphs contain modules? No, because all nodes have roughly the same degree.
- Do W-S graphs contain modules? No, although other small-world networks can contain modules.
- But the brain is indeed modular: organ, hemispheres, coarse divisions, lobes, cytoarchitectural areas, nuclei, etc.

- Modularity and hierarchical organisation offer several advantages: evolvability, flexibility, adaptability, and complexity.
Finding modules or clusters formed by a set of objects is a widely studied problem.

Long history in mathematics, statistics, and computer science.

Module $\equiv$ Cluster $\equiv$ Community.
Definition of Clustering

Given a set of \( n \) objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.
Definition of Clustering

Given a set of \( n \) objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

How do we measure how similar or close two objects are?
Definition of Clustering

Given a set of \( n \) objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

- How do we measure how similar or close two objects are?
- How many subsets?
Definition of Clustering

Given a set of \( n \) objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

- How do we measure how similar or close two objects are?
- How many subsets?
- How do we compare two different partitions?
Measuring Similarity of Objects

- Assume each object specified by a list of values, e.g., $x, y, z$ coordinates indicating voxel position in an fMRI image.
Assume each object specified by a list of values, e.g., $x$, $y$, $z$ coordinates indicating voxel position in an fMRI image.

Distance between two objects $p$ and $q$ is $d(p, q)$.

Euclidean metric: $d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$. 

- Euclidean, Manhattan distances are metrics.
- Correlation, dot product are not metrics.
Assume each object specified by a list of values, e.g., $x, y, z$ coordinates indicating voxel position in an fMRI image.

Distance between two objects $p$ and $q$ is $d(p, q)$.

- **Euclidean metric:** $d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$.
- **Manhattan metric:** $d(p, q) = \sum_i |p_i - q_i|$.

Other distances: normalised dot product, K-L divergence, relative entropy, Pearson’s correlation.

**Metrics obey triangle inequality:**

$$d(p, q) + d(q, r) \geq d(p, r)$$

- **Euclidean**, **Manhattan distances** are metrics.
- **Correlation**, **dot product** are not metrics.
Measuring Similarity of Objects

- Assume each object specified by a list of values, e.g., $x, y, z$ coordinates indicating voxel position in an fMRI image.
- Distance between two objects $p$ and $q$ is $d(p, q)$.
  - Euclidean metric: $d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$.
  - Manhattan metric: $d(p, q) = \sum_i |p_i - q_i|$.
- Other distances: normalised dot product, K-L divergence, relative entropy, Pearson’s correlation.
Presentations Hierarchical clustering MST

Measuring Similarity of Objects

- Assume each object specified by a list of values, e.g., $x$, $y$, $z$ coordinates indicating voxel position in an fMRI image.
- Distance between two objects $p$ and $q$ is $d(p, q)$.
- Euclidean metric: $d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$.
- Manhattan metric: $d(p, q) = \sum_i |p_i - q_i|$.
- Other distances: normalised dot product, K-L divergence, relative entropy, Pearson’s correlation.
- Metrics obey triangle inequality: $d(p, q) + d(q, r) \geq d(p, r)$.
  - Euclidean, Manhattan distances are metrics.
  - Correlation, dot product are not metrics.
Hierarchical Clustering

- Attempt to recursively find sub-modules within modules.
- Natural way to “zoom into” areas of interest.
- Represent using a tree or dendrogram.
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 

![Diagram showing hierarchical clustering process]
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.

2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 

![Diagram illustrating the hierarchical clustering algorithm with points and connections representing clusters merging over time.](image-url)
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 

![Diagram of hierarchical clustering](image-url)
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 

![Diagram of hierarchical clustering process]
Hierarchical Clustering Algorithm

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$. 

![Hierarchical clustering algorithm diagram]
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   - Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   - Merge $C_i$ and $C_j$.
Hierarchical Clustering Algorithm

- Bottom-up clustering algorithm.

1. Start with every object in its own cluster.
2. Repeat
   ▶ Let $C_i$ and $C_j$ be the clusters “nearest” each other.
   ▶ Merge $C_i$ and $C_j$.
3. until all the objects are in one cluster.
Measuring Distance between Clusters

- How do we measure distance between two clusters $C_i$ and $C_j$?

\[ d_{\text{min}}(C_i, C_j) = \text{distance between closest pair of objects.} \]

\[ d_{\text{max}}(C_i, C_j) = \text{distance between farthest pair of objects.} \]

\[ d_{\text{mean}}(C_i, C_j) = \text{average of distances between all pairs of objects.} \]

\[ d_{\text{centroid}}(C_i, C_j) = d(\mu_i, \mu_j), \text{where } \mu_i \text{ is the centroid of } C_i. \]

Methods are called minimum linkage, maximum linkage, mean linkage, and centroid linkage clustering, respectively.

Computing $d_{\text{min}}$, $d_{\text{max}}$, $d_{\text{avg}}$ takes $O(n_i n_j)$ time.

Computing $d_{\text{mean}}$ takes $O(n_i + n_j)$ time.
Measuring Distance between Clusters

- How do we measure distance between two clusters $C_i$ and $C_j$?

$d_{\text{min}}(C_i, C_j) = \text{distance between closest pair of objects.}$

$d_{\text{max}}(C_i, C_j) = \text{distance between farthest pair of objects.}$

$d_{\text{mean}}(C_i, C_j) = \text{average of distances between all pairs of objects.}$

$d_{\text{centroid}}(C_i, C_j) = d(\mu_i, \mu_j)$, where $\mu_i$ is the centroid of $C_i$.

Methods are called minimum linkage, maximum linkage, mean linkage, and centroid linkage clustering, respectively.

Computing $d_{\text{min}}$, $d_{\text{max}}$, $d_{\text{avg}}$ takes $O(n^2)$ time.

Computing $d_{\text{mean}}$ takes $O(n_i + n_j)$ time.
Measuring Distance between Clusters

- How do we measure distance between two clusters $C_i$ and $C_j$?
- $d_{\text{min}}(C_i, C_j) =$ distance between closest pair of objects.
Measuring Distance between Clusters

- How do we measure distance between two clusters $C_i$ and $C_j$?
- $d_{min}(C_i, C_j) =$ distance between closest pair of objects.
- $d_{max}(C_i, C_j) =$ distance between farthest pair of objects.
Measuring Distance between Clusters

- How do we measure distance between two clusters $C_i$ and $C_j$?
- $d_{\text{min}}(C_i, C_j) =$ distance between closest pair of objects.
- $d_{\text{max}}(C_i, C_j) =$ distance between farthest pair of objects.
- $d_{\text{mean}}(C_i, C_j) =$ average of distances between all pairs of objects.
How do we measure distance between two clusters $C_i$ and $C_j$?

- $d_{min}(C_i, C_j) =$ distance between closest pair of objects.
- $d_{max}(C_i, C_j) =$ distance between farthest pair of objects.
- $d_{mean}(C_i, C_j) =$ average of distances between all pairs of objects.
- $d_{centroid}(C_i, C_j) = d(\mu_i, \mu_j)$, where $\mu_i$ is the centroid of $C_i$. 

Methods are called minimum linkage, maximum linkage, mean linkage, and centroid linkage clustering, respectively.

Computing $d_{min}$, $d_{max}$, $d_{avg}$ takes $O(n_i n_j)$ time.
Computing $d_{mean}$ takes $O(n_i + n_j)$ time.
How do we measure distance between two clusters $C_i$ and $C_j$?

- $d_{\text{min}}(C_i, C_j) =$ distance between closest pair of objects.
- $d_{\text{max}}(C_i, C_j) =$ distance between farthest pair of objects.
- $d_{\text{mean}}(C_i, C_j) =$ average of distances between all pairs of objects.
- $d_{\text{centroid}}(C_i, C_j) = d(\mu_i, \mu_j)$, where $\mu_i$ is the centroid of $C_i$.

Methods are called minimum linkage, maximum linkage, mean linkage, and centroid linkage clustering, respectively.

- Computing $d_{\text{min}}, d_{\text{max}}, d_{\text{avg}}$ takes $O(n_i n_j)$ time.
- Computing $d_{\text{mean}}$ takes $O(n_i + n_j)$ time.
Running Time of Hierarchical Clustering

1. Start with every object in its own cluster.
2. Repeat
   - Let $D_i$ and $D_j$ be the clusters “nearest” each other.
   - Merge $D_i$ and $D_j$.
3. until all the objects are in one cluster.
Running Time of Hierarchical Clustering

1. Start with every object in its own cluster.
2. Repeat
   - Let $D_i$ and $D_j$ be the clusters “nearest” each other.
   - Merge $D_i$ and $D_j$.
3. until all the objects are in one cluster.

- Assume computing distance between two objects takes $O(1)$ time.
- Store all $O(n^2)$ inter-object distances.
- At each iteration, compute distance between every pair of clusters: takes $O(n^2)$ time in total.
- There are $n$ iterations, so overall running time is $O(nn^2) = O(n^3)$. 
Hierarchical Clustering Result
Hierarchical Clustering Result
Hierarchical Clustering Result
Hierarchical Clustering Result
Definition of Clustering

Given a set of \( n \) objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

- How do we measure how similar or close two objects are?
Definition of Clustering

Given a set of $n$ objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

- How do we measure how similar or close two objects are?
- How many subsets?
**Definition of Clustering**

Given a set of $n$ objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

- How do we measure how similar or close two objects are?
Definition of Clustering

Given a set of $n$ objects, find the best partition of the objects into subsets such that each subset contains objects that are similar/close to each other.

- How do we measure how similar or close two objects are?
- How do we compare two different partitions?
Example of Clustering
Example of Clustering
Example of Clustering
Example of Clustering
Formalising the Clustering Problem

- Let $U$ be the set of $n$ objects labelled $p_1, p_2, \ldots, p_n$.
- For every pair $p_i$ and $p_j$, we have a distance $d(p_i, p_j)$.
- We require $d(p_i, p_i) = 0$, $d(p_i, p_j) > 0$, if $i \neq j$, and $d(p_i, p_j) = d(p_j, p_i)$.
Formalising the Clustering Problem

- Let $U$ be the set of $n$ objects labelled $p_1, p_2, \ldots, p_n$.
- For every pair $p_i$ and $p_j$, we have a distance $d(p_i, p_j)$.
- We require $d(p_i, p_i) = 0$, $d(p_i, p_j) > 0$, if $i \neq j$, and $d(p_i, p_j) = d(p_j, p_i)$.
- Given a positive integer $k$, a $k$-clustering of $U$ is a partition of $U$ into $k$ non-empty subsets or "clusters" $C_1, C_2, \ldots, C_k$. 
Formalising the Clustering Problem

- Let $U$ be the set of $n$ objects labelled $p_1, p_2, \ldots, p_n$.
- For every pair $p_i$ and $p_j$, we have a distance $d(p_i, p_j)$.
- We require $d(p_i, p_i) = 0$, $d(p_i, p_j) > 0$, if $i \neq j$, and $d(p_i, p_j) = d(p_j, p_i)$.
- Given a positive integer $k$, a $k$-clustering of $U$ is a partition of $U$ into $k$ non-empty subsets or “clusters” $C_1, C_2, \ldots C_k$.
- The spacing of a clustering is the smallest distance between objects in two different subsets:

$$\text{spacing}(C_1, C_2, \ldots C_k) = \min_{1 \leq i, j \leq k, i \neq j, \ p \in C_i, q \in C_j} d(p, q)$$
Formalising the Clustering Problem

- Let $U$ be the set of $n$ objects labelled $p_1, p_2, \ldots, p_n$.
- For every pair $p_i$ and $p_j$, we have a distance $d(p_i, p_j)$.
- We require $d(p_i, p_i) = 0$, $d(p_i, p_j) > 0$, if $i \neq j$, and $d(p_i, p_j) = d(p_j, p_i)$.
- Given a positive integer $k$, a $k$-clustering of $U$ is a partition of $U$ into $k$ non-empty subsets or “clusters” $C_1, C_2, \ldots C_k$.
- The spacing of a clustering is the smallest distance between objects in two different subsets:

$$\text{spacing}(C_1, C_2, \ldots C_k) = \min_{1 \leq i, j \leq k, i \neq j, p \in C_i, q \in C_j} d(p, q)$$

Clustering of Maximum Spacing
Given a set $U$ of objects, a distance function $d : U \times U \rightarrow \mathbb{R}^+$, and a positive integer $k$,
compute a $k$-clustering of $U$ whose spacing is the largest over all possible $k$-clusterings.
Example of Clustering
Algorithm for Clustering of Maximum Spacing

Intuition: greedily cluster objects in increasing order of distance.

Let \( C \) be a set of \( n \) clusters, with each object in \( U \) in its own cluster.

Process pairs of objects in increasing order of distance.

1. Let \((p, q)\) be the next pair with \( p \in C_p \) and \( q \in C_q \).
2. If \( C_p \neq C_q \), add new cluster \( C_p \cup C_q \) to \( C \), delete \( C_p \) and \( C_q \) from \( C \).

Stop when there are \( k \) clusters in \( C \).

Same as Kruskal's algorithm but do not add last \( k - 1 \) edges in MST.
Algorithm for Clustering of Maximum Spacing

Intuition: greedily cluster objects in increasing order of distance.

1. Let $C$ be a set of $n$ clusters, with each object in $U$ in its own cluster.
2. Process pairs of objects in increasing order of distance.
   - Let $(p, q)$ be the next pair with $p \in C_p$ and $q \in C_q$.
   - If $C_p \neq C_q$, add new cluster $C_p \cup C_q$ to $C$, delete $C_p$ and $C_q$ from $C$.
3. Stop when there are $k$ clusters in $C$.

Same as Kruskal's algorithm but do not add last $k-1$ edges in MST.
Algorithm for Clustering of Maximum Spacing

Intuition: greedily cluster objects in increasing order of distance.

Let $C$ be a set of $n$ clusters, with each object in $U$ in its own cluster.

Process pairs of objects in increasing order of distance.

1. Let $(p, q)$ be the next pair with $p \in C_p$ and $q \in C_q$.
2. If $C_p \neq C_q$, add new cluster $C_p \cup C_q$ to $C$, delete $C_p$ and $C_q$ from $C$.

Stop when there are $k$ clusters in $C$.

Same as Kruskal’s algorithm but do not add last $k - 1$ edges in MST.
Algorithm for Clustering of Maximum Spacing

- Intuition: greedily cluster objects in increasing order of distance.
Algorithm for Clustering of Maximum Spacing

- Intuition: greedily cluster objects in increasing order of distance.
- Let \( C \) be a set of \( n \) clusters, with each object in \( U \) in its own cluster.
- Process pairs of objects in increasing order of distance.
  - Let \((p, q)\) be the next pair with \( p \in C_p \) and \( q \in C_q \).
  - If \( C_p \neq C_q \), add new cluster \( C_p \cup C_q \) to \( C \), delete \( C_p \) and \( C_q \) from \( C \).
- Stop when there are \( k \) clusters in \( C \).
Algorithm for Clustering of Maximum Spacing

- Intuition: greedily cluster objects in increasing order of distance.
- Let $\mathcal{C}$ be a set of $n$ clusters, with each object in $U$ in its own cluster.
- Process pairs of objects in increasing order of distance.
  - Let $(p, q)$ be the next pair with $p \in C_p$ and $q \in C_q$.
  - If $C_p \neq C_q$, add new cluster $C_p \cup C_q$ to $\mathcal{C}$, delete $C_p$ and $C_q$ from $\mathcal{C}$.
- Stop when there are $k$ clusters in $\mathcal{C}$.
- Same as Kruskal’s algorithm but do not add last $k - 1$ edges in MST.
What is the spacing of the Algorithm’s Clustering?

- Let $C$ be the clustering produced by the algorithm.
- What is $\text{spacing}(C)$?
What is the spacing of the Algorithm’s Clustering?

- Let $C$ be the clustering produced by the algorithm.
- What is \( \text{spacing}(C) \)? It is the cost of the \((k - 1)\)st most expensive edge in the MST. Let this cost be \( d^* \).
Why does the Algorithm Work?

Let $C'$ be any other clustering.

We will prove that $\text{spacing}(C') \leq d^*$. 
Why does the Algorithm Work?

- Let $C'$ be any other clustering.
- We will prove that spacing$(C') \leq d^*$. 
spacing($C'$) \leq d^*: Intuition
$\text{spacing}(C') \leq d^*: \text{ Intuition}$
\[ \text{spacing}(C') \leq d^*: \text{Intuition} \]
\[ \text{spacing}(C') \leq d^*: \text{ Intuition} \]
\textbf{spacing}(C') \leq d^*

- There must be two objects \( p_i \) and \( p_j \) in \( U \) in the same cluster \( C_r \) in \( C \) but in different clusters in \( C' \):
spacings C′) ≤ d∗

- There must be two objects p_i and p_j in U in the same cluster C_r in C but in different clusters in C': spacings C′) ≤ d(p_i, p_j).
spacing($C'$) $\leq d^*$

- There must be two objects $p_i$ and $p_j$ in $U$ in the same cluster $C_r$ in $C$ but in different clusters in $C'$: $\text{spacing}(C') \leq d(p_i, p_j)$. But $d(p_i, p_j)$ could be $> d^*$.
- Suppose $p_i \in C'_s$ and $p_j \in C'_t$ in $C'$. 
There must be two objects $p_i$ and $p_j$ in $U$ in the same cluster $C_r$ in $C$ but in different clusters in $C'$: spacing($C'$) $\leq d(p_i, p_j)$. But $d(p_i, p_j)$ could be $> d^*$.

Suppose $p_i \in C'_s$ and $p_j \in C'_t$ in $C'$.

All edges in the path $Q$ connecting $p_i$ and $p_j$ in the MST have length $\leq d^*$.

In particular, there is an object $p \in C'_s$ and an object $p' \not\in C'_s$ such that $p$ and $p'$ are adjacent in $Q$.

$d(p, p') \leq d^* \Rightarrow$ spacing($C'$) $\leq d(p, p') \leq d^*$.

Figure 4.15 An illustration of the proof of (4.26), showing that the spacing of any other clustering can be no larger than that of the clustering found by the single-linkage algorithm.