Gene Expression Analysis.

DNA arrays ⇒ expression levels or amount of mRNA produced in the cell of 'n' genes, under many time points & conditions.

⇓

genes switched on and switched off in the cell.

$[I]_{n \times m}$ = expression matrix.

$= [I_{ij}]_{\forall}$

I_{ij} ⇒ Intensities of hybridization signals as provided by DNA array (for i^{th} gene at j^{th} time instant & conditions).

⇒ Intensities are transformed & normalized. (Not discussed here).

Expression pattern of gene i ⇔ i^{th} row of I.

$n \times m$ expression matrix ⟶ $n \times n$ distance matrix $d = (d_{ij})$.

Clustering algorithms groups genes with similar expression patterns into clusters with the hopes that these clusters correspond to groups of functionally related genes.

Clustering principle:

- Homogeneity: d_{ij} small if i & j belong to same group.
- Separation: d_{ij} large if they belong to diff. cluster.

* Hierarchical Clustering (d, n)

1. Form n clusters with single element.
   Let S_c = set of initial clusters. V = {vertex set corresponds to clusters}.
2. While (there is more than 1 cluster)
   {
   2.1 Find the two closest clusters C_1 & C_2.
   2.2 Merge C_1 & C_2 to C with $|C_1| + |C_2|$ elements.
   2.3 Compute distances of C with other clusters.
   2.4 Add a vertex, corresponds to C whose children are C_1 & C_2.
   2.5 Remove C_1 & C_2 from the list & add C to the list.
   }

Diff. distances measures:

$$d_{min}(C^*, C) = \min_{x \in C^*, y \in C} d(x, y)$$

$$d_{av}(C^*, C) = \underset{x \in C^*, y \in C}{\text{avg.}} \{ d(x, y) \}$$

? $$d_H(C^*, C) = \frac{d_H(C^*, C_1) + d_H(C^*, C_2) - d_H(C_1, C_2)}{2}$$

* K-means clustering.

Let $\chi = \{x_1, x_2, \ldots, x_k\}$ be k-centers.

Given a data point v, let the squared error distortion be:

$$d(v, \chi) = \min_{1 \le i \le k} d(v, x_i)$$

Let $V = \{v_1, v_2, \ldots, v_n\}$

$$\therefore d(V, \chi) = \frac{\sum_{i=1}^{n} (d(v_i, \chi))^2}{n}$$

= avg. squared error distortion.

k-means clustering problem:

Given n data points, find k center points minimizing the avg. squared error distortion.

→ no efficient (polynomial) alg. known yet.

Lloyd k-means clustering algorithm (k)

1. Choose arbitrary k points as cluster representatives. : $\{x_1, x_2, \dots x_k\}$
2. while (partitions converges or fluctuations are small)
   {
   2.1. Assign each data point to the cluster c_i corresponding to the closest cluster representative x_i $(1 \le i \le k)$
   2.2. $\forall i$, $x_i = \dfrac{\sum_{v_j \in c_i} v_j}{|c_i|}$
   }

Variation over distance measure:

$$d(v, x) = \sum_{i=1}^{n} d(v_i, x) \Rightarrow \text{K-Median Prob.}$$

$$d(v, x) = \max_{1 \le i \le n} d(v_i, x) \Rightarrow \text{K-center Prob.}$$

While Lloyd alg. is fast, it can significantly rearrange every cluster in every iteration. A conservative approach is to move only one element between clusters in every iteration (the most suitable replacement is preferred which decreases mainly the cost fn. or distance).

* Greedy Approach for K-Means Clustering:

Let P denotes a partitioning into k partitions.
Cost (P) = associated distortion measure.
$P_{i \to c}$ = Partition obtained from P by moving v_i to c.

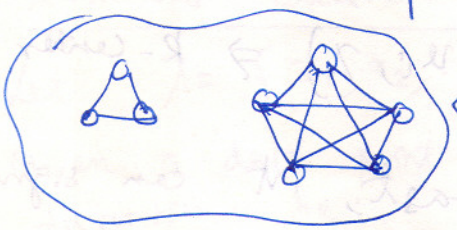
$\therefore$ improvement $\Rightarrow \Delta(i \to c) = \text{cost}(P) - \text{cost}(P_{i \to c}) > 0$

Greedy method takes that move which maximises this maximal improvement. Stop iterating after the improvement becomes zero or ~~small~~.

* Clustering through clique finding / transformation.

Complete graph: K_n (undirected) with n nodes with every two vertices connected by an edge.

Clique graph: Every connected component is a complete graph.



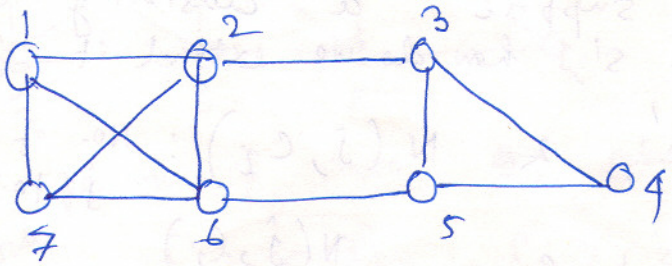
← clique graph.

⇒ Each component is a string cluster.

Complete Subgraph: If $V' \subset V$ in a graph $G(N, E)$ induces a subgraph which is complete, the subgraph is complete.

Clique: A clique in a ~~subgraph~~ graph is a maximal complete subgraph.

i.e. a complete subgraph which is not contained inside any other ~~com~~ complete subgraph.



$\{1, 7, 6\}$ ⇒ Not a clique but a complete subgraph.

Cliques are: $\{1, 2, 6, 7\}$; $\{3, 4, 5\}$.

Expression matrix → Distance Matrix → Distance graph

$\{I_{v,s}\}$ $\{d_{i,j}\}$ $\{E(i,j) \text{ if } d_{i,j} < \theta\}$

↑ threshold.

Errors in expression data and the absence of a "universal good threshold" θ often results in distance distance graphs that do not quite form clique graphs.

Corrupted clique problem:

Determine the smallest number of edges that need to be added or removed to transform a graph into a clique graph.

→ NP-hard. → Need for approx. soln.

Two ~~algo~~ approximate algorithms:

- Parallel Classification with cores (PCC)
- Cluster Affinity Search Technique (CAST)

- Parallel Classification with Cores (PCC) Alg.:

(Amir Ben-Dor et. al. (1999))

Let $S' \subset S$, suppose a clustering $(C_1, C_2, \ldots C_k)$ is available on S', how do we extend it for S?

Let $j \in S - S'$, & $N(j, C_i)$: No. of edges from j to C_i.

$\therefore$ affinity j to $C_i = \dfrac{N(j, C_i)}{|C_i|}$

Assign j to that cluster which has maximum affinity.

~~PCC (S, G, k) : S is a distance graph, k no. of clusters.~~

PCC (S, G, k)

S: Set of n elements (say genes, forming vertices of G)

G: Dist graph

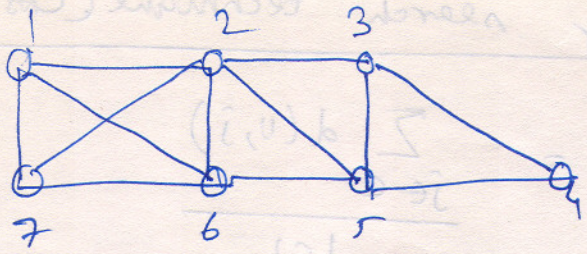
k: no. of clusters.

1. Randomly select $S' \subset S$ & $S'' \subset S - S'$ s.t. $|S'| \approx \log\log n$ & $|S''| \approx \log n$

2. For all k partition in S'; obtain extended partition in S, through two stages of extensions ($S' \to S'' \to S - (S' \cup S'')$). & choose the one which has ~~minimum clustering~~ maximum score (i.e. the no. of edges required to add or remove from G, to get a clique graph as per the partition).

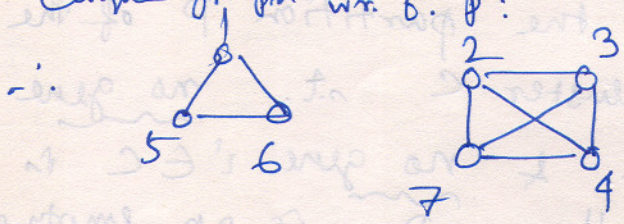
Eg,

Say, G =



Let a partn P into 2 partns be say as follows: $\{1, 5, 6\}$, $\{2, 3, 7, 4\}$.

Clique graph wr.t. P:



No. of edges to be added or removed in G:

For addition: $\{(1,5), (4,7), (2,4), (3,7)\}$

+

For deletion: $\{(1,2), (1,7), (7,6), (2,5), (3,5), (4,5)\}$

$= 10$

$\therefore$ score (P) $= 10$.

Time complexity:

No. of partitions in $S' = k^{|S'|} = k^{\log \log n} = (\log n)^{\log_2 k}$

$= (\log n)^c$

In each iteration $O(n^2)$ operations for extension and score computation.

$\therefore$ Total Time complexity: $O(n^2 (\log n)^c)$

where $c = \log_2 k$

- **cluster affinity search technique (CAST)**

Let, $$d(v, C) = \frac{\sum_{j \in C} d(v, j)}{|C|}$$

Given a threshold θ, i is close to C, if $d(i, C) < \theta$ else, i is distant.

CAST iteratively builds the partition P of the set S by finding a cluster C s.t. no gene $i \notin S$ is close to C & no gene $i \in C$ is distant to C. Initially P is an empty set.

```
CAST (S, G, θ)
1. P = φ
2. while (S ≠ φ)
   {
   2.1. v = vertex of maximal degree in G.
   2.2. C ← {v};
   2.3. while (∃ a close gene i ∉ C  or  ∃ distant gene j ∈ C)
        {
        2.3.1. Find the closest gene i ∉ C & add to C.
        2.3.2. Find the farthest (distant) gene i ∈ C & remove it.
        }
   2.4  Add C to P
   2.5. S = S − C; & update G (distance graph) accordingly.
   }
```

There is no performance guarantee & it may not converge at all. But, in general it gives good performance.