Using DP for hierarchical discretization of continuous attributes

Amit Goyal (31st March 2008)

Reference

- Ching-Cheng Shen and Yen-Liang Chen. A dynamic-programming algorithm for hierarchical discretization of continuous attributes. *In European Journal of Operational Research 184 (2008) 636-651 (ElseVier).*

Overview

- What is Discretization?
- Why need Discretization?
- Issues involved
- Traditional Approaches
- DP solution

Background

- Discretization
  - reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals.
- Concept hierarchies
  - reduce the data by collecting and replacing low level concepts (such as numeric values for the attribute age) by higher level concepts (such as young, middle-aged, or senior).

Why need discretization?

- Data Warehousing and Mining
  - Data reduction
  - Association Rule Mining
  - Sequential Patterns Mining
- In some machine learning algorithms like Bayesian approaches and Decision Trees.
- Granular Computing

Discretization Issues

- Size of the discretized intervals affect support & confidence

  $\{Refund = No, (Income = \$51,250)\} \rightarrow \{Cheat = No\}$

  $\{Refund = No, (60K \leq Income \leq 80K)\} \rightarrow \{Cheat = No\}$

  $\{Refund = No, (0K \leq Income \leq 1B)\} \rightarrow \{Cheat = No\}$

  - If intervals too small
    - may not have enough support
  - If intervals too large
    - may not have enough confidence
- Loss of Information (How to minimize?)
- Potential solution: use all possible intervals
- Too many rules!!!

Common Approaches

- Manual
- Equal-Width Partition
- Equal-Depth Partition
- Chi-Square Partition
- Entropy Based Partition
- Clustering

Simple Discretization Methods: Binning

- Equal-width (distance) partitioning:
  - It divides the range into *N* intervals of equal size: uniform grid
  - if *A* and *B* are the lowest and highest values of the attribute, the width of intervals will be: $W = (B-A)/N$.
  - The most straightforward
- Equal-depth (frequency) partitioning:
  - It divides the range into *N* intervals, each containing approximately same number of samples

Chi-Square Based Partitioning

- X^2 (chi-square) test

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

- The larger the X^2 value, the more likely the variables are related
- Merge: Find the best neighboring intervals and merge them to form larger intervals recursively

Entropy Based Partition

- Given a set of samples S, if S is partitioned into two intervals S1 and S2 using boundary T, the entropy after partitioning is

$$E(S,T) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2)$$

- The boundary that minimizes the entropy function over all possible boundaries is selected as a binary discretization.
- The process is recursively applied to partitions obtained until some stopping criterion is met

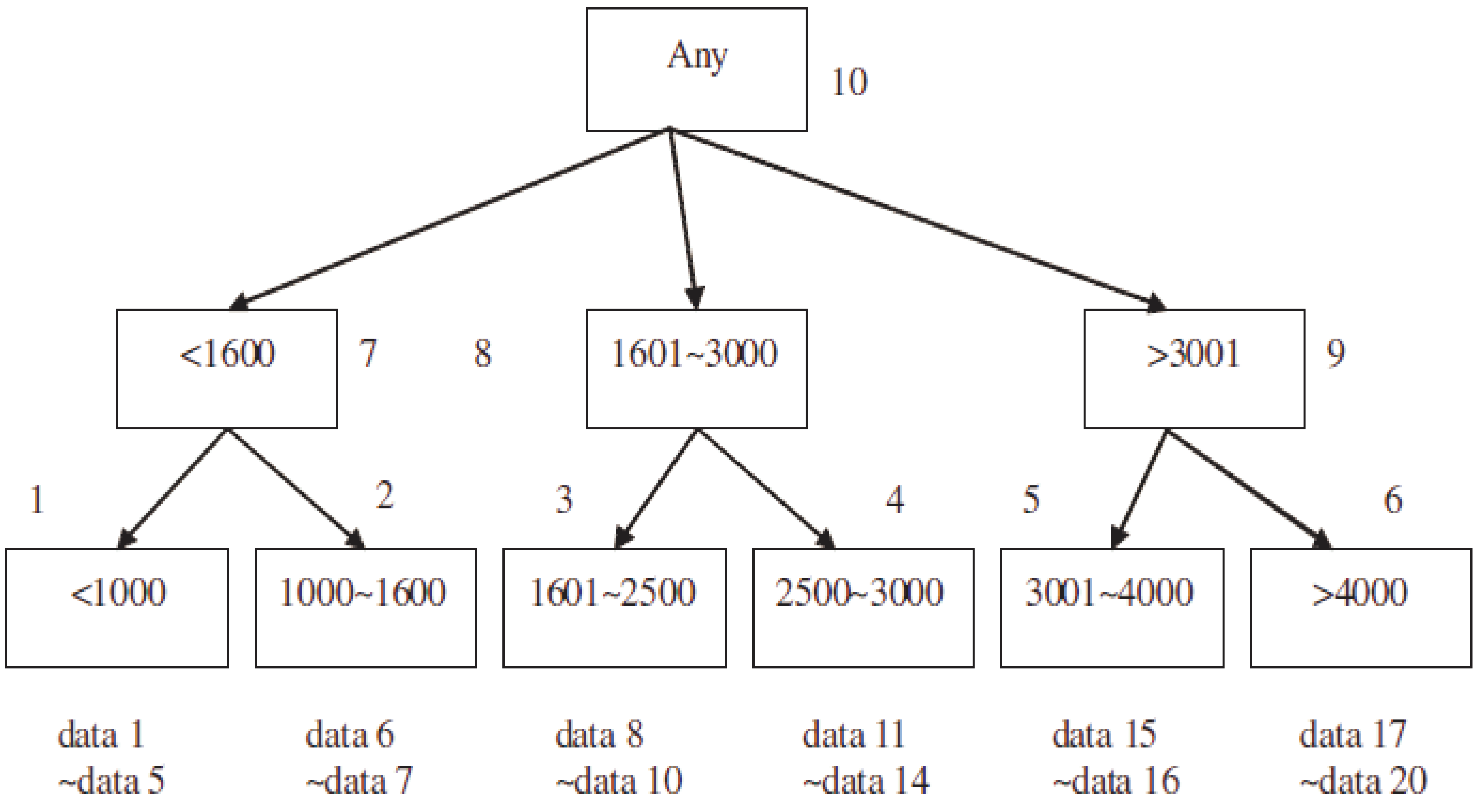
Clustering

- Partition data set into clusters based on similarity, and store cluster representation (e.g., centroid and diameter) only
- Can be very effective if data is clustered but not if data is "smeared"
- Can have hierarchical clustering and be stored in multi-dimensional index tree structures
- There are many choices of clustering definitions and clustering algorithms

Notations

- *val(i)*: value of ith data
- *num(i)*: number of occurrences of value val(i)
- *R*: depth of the output tree
- *ub*: upper boundary on the number of subintervals spawned from an interval
- *lb*: lower boundary

Example



R = 2, lb = 2, ub = 3

Problem Definition

Given parameters *R*, *ub*, and *lb* and input data *val(1)*, *val(2)*, …, *val(n)* and *num(1)*, *num(2)*, … *num(n)*, our goal is to build a minimum volume tree subject to the constraints that all leaf nodes must be in level *R* and that the branch degree must be between *ub* and *lb*

Distances and Volume

- Intra-distance of a node containing data from data *i* to data *j*

$$\text{intradist}(i, j) = \sum_{x=i}^{j} (val(x) - mean(i, j)) * num(x)$$

- Inter-distance b/w two adjacent siblings; first node containing data from i to u, second node containing data from u+1 to j

$$\begin{aligned}\text{interdist}(i, j, u) &= \beta \times (val(u+1) - val(u)) \times \text{totalnum}(i, j) \\ &= \beta \times (val(u+1) - val(u)) \times (\text{totalnum}(\mathrm{i}, \mathrm{u}) + \text{totalnum}(\mathrm{u}+1, \mathrm{j})) \\ &= \text{interdist}^{\mathrm{L}}(i, u) + \text{interdist}^{R}(u+1, j)\end{aligned}$$

- Volume of a tree is the total intra-distance minus total inter-distance in the tree

Theorem

- The volume of a tree = the intra-distance of the root node + the volumes of all its sub-trees - the inter-distances among its children

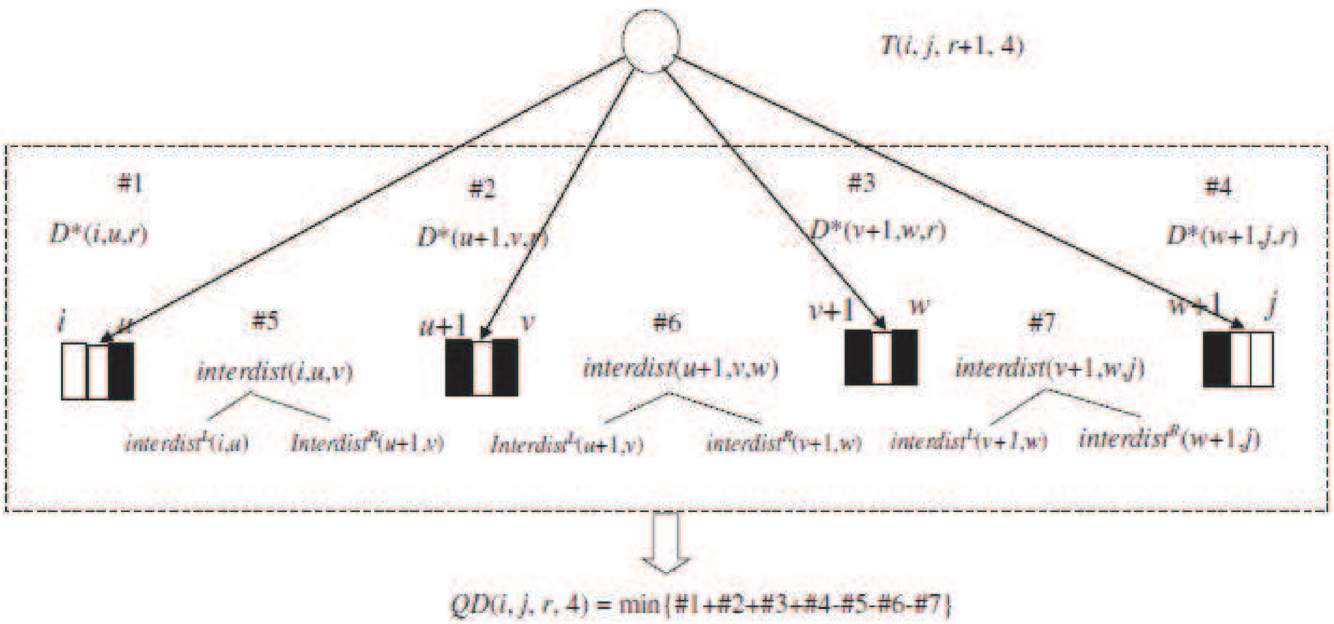
Notations

- $T^*(i,j,r)$: the minimum volume tree that contains data from data i to data j and has depth r
- $T(i,j,r,k)$: the minimum volume tree that contains data from data i to data j, has depth r, and whose root has k branches
- $D^*(i,j,r)$: the volume of $T^*(i,j,r)$
- $D(i,j,r,k)$: the volume of $T(i,j,r,k)$

Notations Cont.

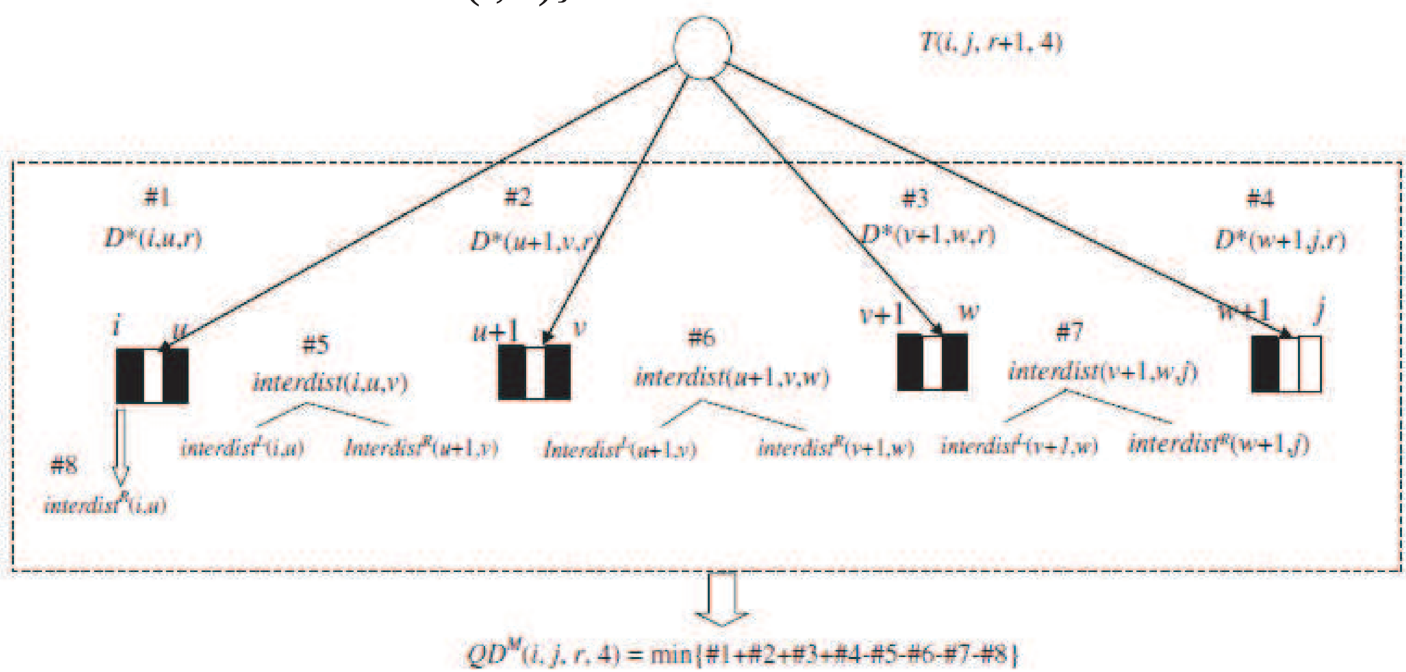
$$QD(i,j,r,k):\ \min\Big\{\sum_{v=1}^{k}(\textit{the volume of } v^{th}\textit{ node}) -$$

$$\sum_{v=1}^{k-1}(\textit{the}\text{ interdistance }\textit{b/w } v^{th}\textit{ node and } (v+1)^{th}\textit{ node})\Big\}$$



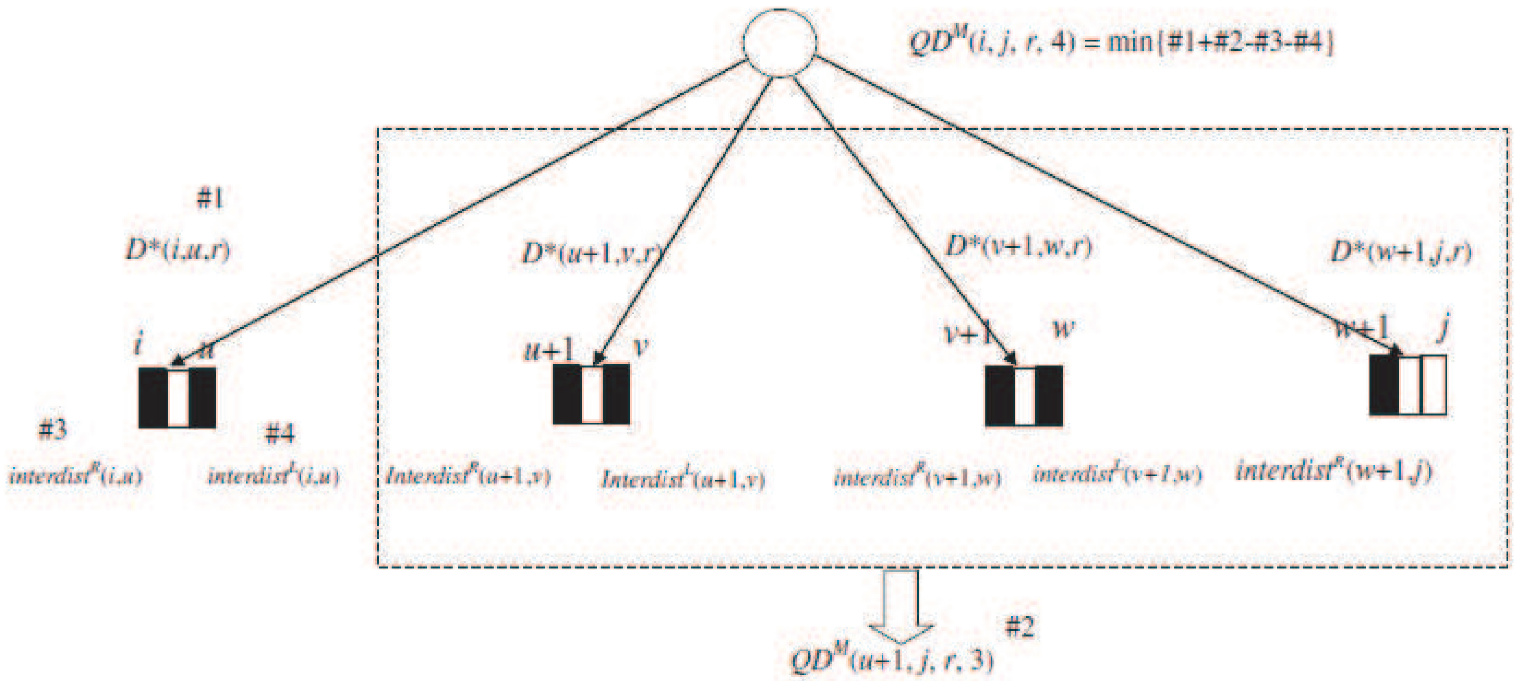
Notations Cont.

$$QD^{M}(i,j,r,k):\ \min\{\sum_{v=1}^{k}(\textit{the volume of } v^{th} \textit{ node}) - \sum_{v=1}^{k\text{-}1}(\textit{the } \text{interdistance } b/w\ v^{th} \textit{ node and } (v+1)^{th} \textit{ node}) - \text{interdistance}^{R}(i,u)\}$$



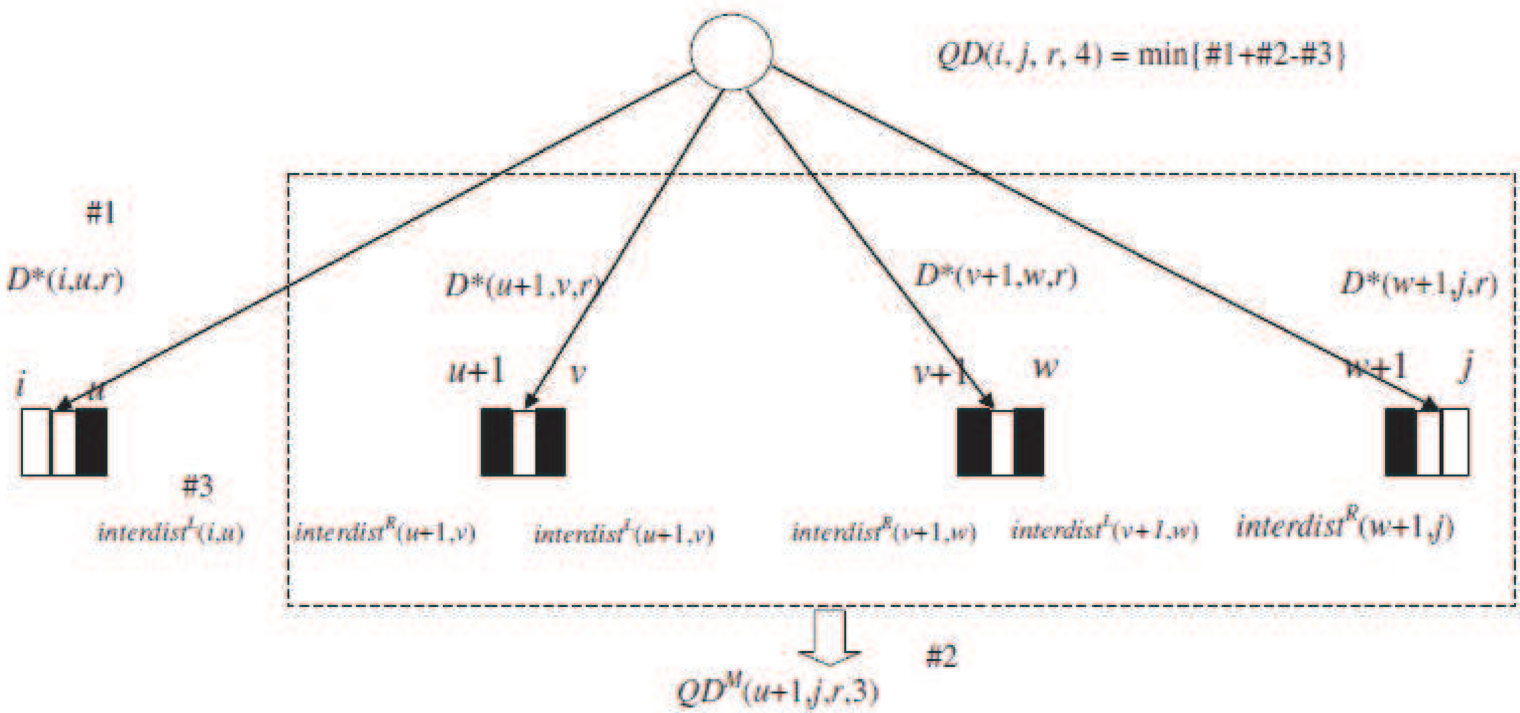
Algorithm

$$QD^M(i,j,r,k) = \min_{i \le u < j}\{D^*(i,u,r) + QD^M(u+1,j,r,k-1) - \text{interdist}^R(i,u) - \text{interdist}^L(i,u)\}$$



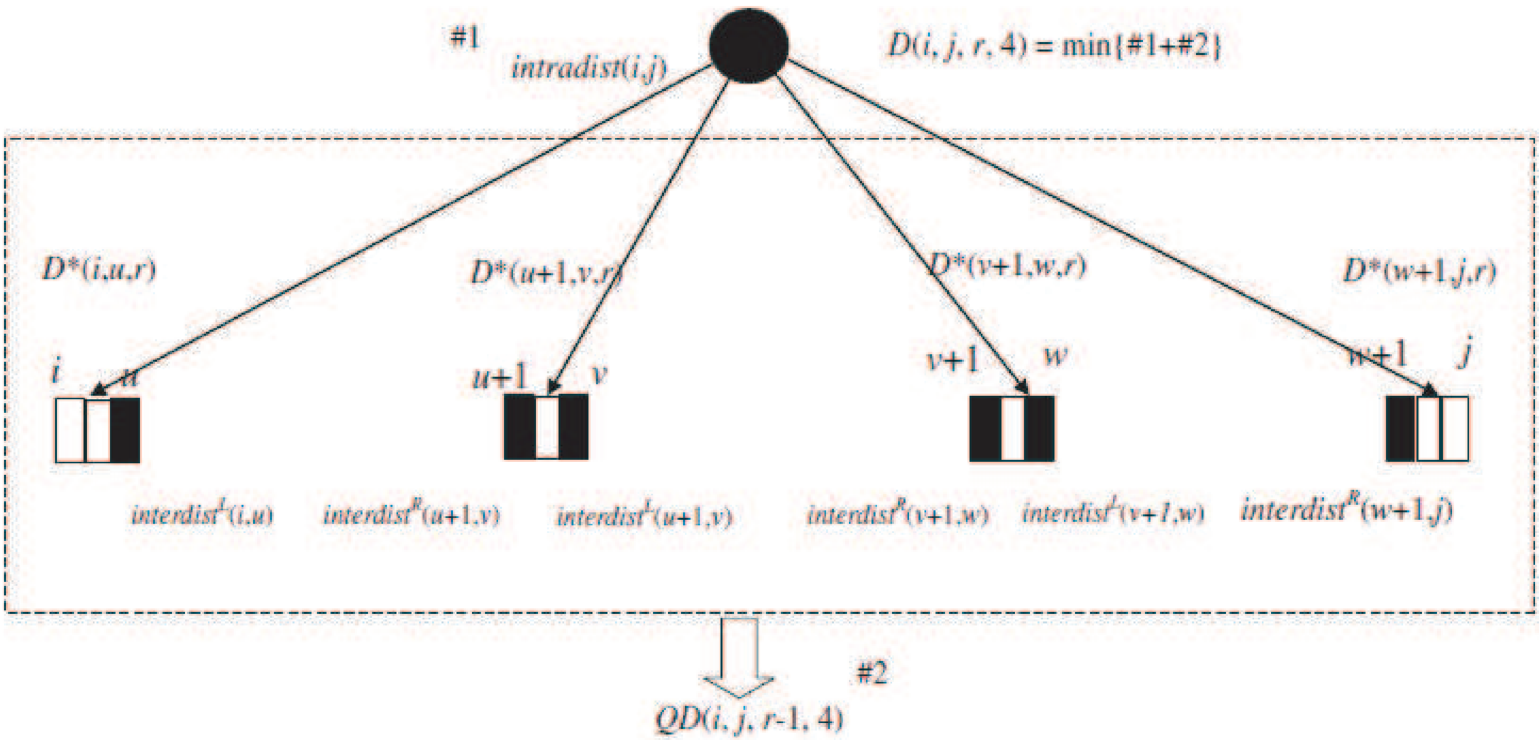
Algorithm Cont.

$$QD(i,j,r,k) = \min_{i \le u < j}\{D^*(i,u,r) + QD^M(u+1,j,r,k-1) - \text{interdist}^L(i,u)\}$$



Algorithm Cont.

$$D(i,j,r,k) = \text{intradist}(i,j) + QD(i,j,r-1,k)$$



The complete DP algorithm

$$QD^M(i,j,r,k) = \min_{i \le u < j}\{D^*(i,u,r) + QD^M(u+1,j,r,k-1) - \text{interdist}^R(i,u) - \text{interdist}^L(i,u)\}$$

$$QD(i,j,r,k) = \min_{i \le u < j}\{D^*(i,u,r) + QD^M(u+1,j,r,k-1) - \text{interdist}^L(i,u)\}$$

$$D(i,j,r,k) = \text{intradist}(i,j) + QD(i,j,r-1,k)$$

$$D^*(i,j,r) = \min_{lb \le k \le rb}\{D(i,j,r,k)\}$$

Steps

- Base Case (r=0):
  - $D^*(i,j,0) = intradist(i,j)$
  - $QD(i,j,0,1) = intradist(i,j)$
- For k = 2 to *ub*
  - Compute $QD^M(i,j,0,k)$
  - Compute $QD(i,j,0,k)$
- For r = 1 to R
  - Compute $D(i,j,r,k)$
  - Compute $D^*(i,j,r)$
  - Compute $QD^M(i,j,r,1)$
  - Compute $QD^M(i,j,r,k)$
  - Compute $QD(i,j,r,k)$