A Fast 'Parsing' Algorithm for Conceptual Clustering

N. Sanjay Jaishankar
#14-A, 10th Cross, 3rd Main,
Hanumanth Nagar, Bangalore 560019, Karnataka, India

1. Introduction
The notion of conceptual clustering was introduced by Michalski in 1980 [1] where he used concepts at a more physical level such as (color = 'red') and (height = 'tall') and so on. Shankar et. al. [2, 3] explored the possibility of using a Knowledge Base in the Clustering process. They used concepts at one or more levels higher than the physical level. Shekar [4] suggests an algorithm having an exponential time complexity. This paper suggests a method which uses only integers during the clustering process. However, the algorithm based on the method has not been included here due to paucity of space. Refer to [4] for the terms Concept, Cohesion Forest, N-trees, Q-trees and Objects used in this paper.

2. The idea
The idea involves reorganizing the Knowledge Base (KB) used in [2, 3], and the presentation of a method for clustering, which uses KB so re-organized. The crux of the reorganization (see figures) involves associating every node in Cohesion Forest (CF) with an unique integer. This association is made in a 'breadth-first-traversal' fashion for all trees in CF. Formally, the following rules govern the association scheme (also referred as 'numbering'). (1) Each parent and its children are associated with consecutive integers, (2) A parent is associated with an integer smaller than any of its children, (3) Siblings can be numbered in any order, provided they confirm to (1) and (2), and (4) This 'consecutive numbering scheme applies only to parents and their respective children, viz. Two nodes without a common parent, need not have consecutive integers associated with them.

Two other factors are considered in the re-organization, (1) Concepts involving 'OR' conditions(s) (as dictated by tuple 'c' of a non-terminal node) are converted into a series of concepts involving only 'AND' conditions [4], and (2) The N-trees and Q-trees present in the CF are 'merged', i.e., the N-trees are modified to also include the information hither to present in the Q-trees

3. The method
The method is fairly straight-forward. It is evident (Fig. b) that the presence of concepts 5,6,7 imply the presence of node 4, and the presence of 2,3,4 in turn imply the presence of 1. Such integer ranges which imply the presence of a parent node at every level in all trees is maintained in KB. Hence, whenever a sequence like 2,3,4 is encountered, it is replaced by the parent, i.e., 1. The proposed algorithm also yields overlapped clusters [4]. We show below a sample input and its output (see [4] for detail).

Input: (x1,blade,cutting),(x2,pencil,marking), (x3,eraser, erasing),(x4,brush,brushing),(x5,soap,soaping),(x6,water, wettening)

Output: [C1: {x1,x2,x3,x4,x5,x6}; D1:C1-(writing OR shaving/1,4}(1,2)]

The above output indicates that all the objects x1 to x6 can be put together in a single cluster which can execute the concepts Writing OR Shaving/1,4, but not both at any given instant of time.

4. Performance
Compile time: The method works best if there are no OR concepts to be converted to AND concepts. At worst the time complexity becomes exponential.
Run time: The best case occurs when the input generates no overlapped clusters. The time complexity is O(n). It may be shown that the time complexity is less than 2n, based on elemental comparisons. At worst, the complexity is O(k.n!), where n = total number of nodes in KB and k = total number of input objects.

Reference: On request