GENERALITY AND COMPUTATIONAL COST

Azriel Rosenfeld
Computer Science Center
University of Maryland
College Park, Maryland 20742
U.S.A.

The purpose of this note is pedagogical. It discusses how one can reduce the computational cost of applying a set of operators (or predicates) by breaking them up into combinations of commonly occurring, simpler ones. This can be thought of as a process of generalization, in the sense that the common, simple operators are more "general" than the original, more complex ones. We are thus suggesting that even when one has a priori knowledge of a specialized nature (i.e., that the complex operators are applicable), it may still be desirable to use generalized operators in order to reduce computational cost.

To illustrate this idea, suppose that we want to apply a set of predicates $P_1,\ldots,P_n$ to an input I, and suppose that the cost of applying predicate P_i is (proportional to) the cardinality $|S_i|$ of its set of support $S_i \subseteq I$. Thus the total cost of applying the P's is $\sum_{i=1}^{n}|S_i|$. For example, applying P_i might involve a template-matching process, where P_i is true iff. a perfect match to the template is found in I. Here I could be an image, or a string (where the "template" is the right-hand side of a rule in a grammar), or a graph (where the "template" is a subgraph). In what follows, we will use the image/template metaphor.

Suppose now that there exists a set of subtemplates $Q_1,\ldots,Q_m$ such that, for $1 \leq i \leq n$, P_i is a concatenation of n_i of the Q_j's. The cost of applying the Q_j's to I is $\sum_{j=1}^{m}|T_j|$, where T_j is Q_j's set of support. If we store the match positions in a new array I', then to test for P_i, we need only apply a template of cardinality n_i to I'. Thus testing for all the P_i's costs $\sum_{i=1}^{n}n_i$, and the total cost of the two-step matching process is $\sum|T_j| + \sum n_i$.

Under what circumstances is the two-step cost less than the brute-force cost $\sum|S_i|$ of applying the P_i's directly? We claim that this depends on the degree to which the Q's "generalize" the P's — i.e., on how few Q's are needed to construct all the P's. For concreteness, suppose that all the Q_j's have the same support size $|T_j| = r$, and that each P_i consists of the same number $n_i = s$ of Q_j's. Thus each P_i has support size $|S_i| = rs$, and the costs of the brute-force and two-step approaches are nrs and mr+ns, respectively. If there are few Q's, they must be used repeatedly, and we have m << ns (m=ns would mean that each Q is used only once); thus mr+ns will be much smaller than nrs. The fewer Q's we need, the greater a saving mr+ns is over nrs. Thus the more we can generalize the P's, the lower the computational cost.

This template example is certainly not a universal one. It would be desirable to extend this type of analysis to other situations. (On the advantages of hierarchical matching in the graph/subgraph case see Barrow et al. (1972).) However, our example does illustrate the idea that it may be advantageous to use generalized rather than specialized knowledge (see Zucker et al. (1975)), because this can lead to savings in computational cost.

References

Barrow, H. R., A. P. Ambler, and R. M. Burstall, Some techniques for recognizing structure in pictures, in S. Watanabe, ed., Frontiers of Pattern Recognition, Academic Press, N. Y., 1972, 1-29.

Zucker, S. W., A. Rosenfeld, and L. S. Davis, General-purpose models: expectations about the unexpected, Proc. 4IJCAI, 1975, 716-721.

The support of the National Science Foundation under Grant MCS-76-23763 is gratefully acknowledged, as is the help of Mrs. Shelly Rowe in preparing this note.