Compressing Fingerprint Templates by Solving an Extended Minimum Label Spanning Tree Problem*

Andreas Chwatal* Günther Raidl* Olivia Dietzel

*Vienna University of Technology
Favoritenstraße 9–11/1861, 1040 Vienna, Austria
chwatal|raidl@ads.tuwien.ac.at

1 Introduction

In a currently ongoing project, we investigate a new possibility for compressing fingerprint template data. The application background is to encode fingerprint information with watermarking techniques e.g. in images of identification cards as an additional security feature. Since the amount of information that can be stored by means of watermarking is very limited, extraordinary compression mechanisms are required in order to achieve reasonably small errors when finally checking fingerprints against the encoded templates.

Having a scanned fingerprint image, traditional image processing techniques are applied for determining its *minutiae*, which are points of interest such as bifurcations, crossover, and ridge endings. Fingerprint matching algorithms are usually based on these minutiae data [3]. Typically, 15 to 60 minutiae are extracted from a single fingerprint, and for each we obtain as attributes its type, x and y coordinates, and an angle. The task we focus on here is the selection of a pre-specified number k of all minutiae in combination with their lossless encoding in a highly compact way.

More formally, we consider as raw data n d-dimensional points (vectors) $V = \{v_1, \dots, v_n\}$ from a discrete domain $\mathbb{D} = \{0, \dots, \tilde{v}^1 - 1\} \times \dots \times \{0, \dots, \tilde{v}^d - 1\}$ corresponding to our minutiae data ($d = 4$ in our case). The domain limits $\tilde{v}^1, \dots, \tilde{v}^d \in \mathbb{N}$ represent the individual sizes and resolutions of the d dimensions.

In our compression approach we select a subset of k of these n points and connect them by a directed spanning tree (outgoing arborescence). Each arc of this tree represents the relative geometric position of its end point in dependence of its starting point. In addition, we use a small set of specially chosen *template arcs*. Instead of storing for each tree arc its length in any of the d dimensions, we encode it more tightly by a reference to the most similar template arc plus a correction vector from a small domain. Thus, the set of template arcs acts as a codebook. In order to achieve a high compression rate, we optimize the selection of encoded points, the tree structure, and the set of template arcs at the same time by a GRASP [2] approach. The domain for the correction vector is pre-specified, while the number of template arcs is the objective to be minimized.

*This work is part of a cooperation with the Fraunhofer Institute for Production Systems and Design Technology, Berlin. It is further supported by the European RTN ADONET under grant 504438.

Having solved this problem, we finally store as compressed information the template arc set and the tree. The latter is encoded by traversing it with depth-first search; at each step we write one bit indicating whether a new arc has been traversed to reach a new node or backtracking along one edge took place. When following a new arc, a reference to its template arc plus the (small) correction vector are additionally written.

2 Problem Definition

We now define the optimization problem we need to solve in a more formal way. Given is a complete directed graph $G = (V, A)$ with the node set V corresponding to our n points and arc set $A = \{(i, j) \mid i, j \in V \wedge i \neq j\}$. A solution to our problem consists of

- a vector of arcs $R = (r_1, \ldots, r_m) \in \mathbb{D}^m$ of arbitrary size m representing the set of template arcs,
- a rooted, outgoing tree $T = (V_T, A_T)$ with $V_T \subseteq V$ and $A_T \subset A$ connecting precisely $|V_T| = k$ nodes,
- a template arc index $\kappa_{(i,j)} \in \{1, \ldots, m\}$ for each tree arc $(i, j) \in A_T$,
- and a correction vector $\delta_{(i,j)} \in D'$ for each tree arc $(i, j) \in A_T$ from a pre-specified, small domain $\mathbb{D}' = \{0, \ldots, \tilde{\delta}^1 - 1\} \times \ldots \times \{0, \ldots, \tilde{\delta}^d - 1\}$, $\mathbb{D}' \subseteq \mathbb{D}$.

The crucial condition for a solution to be feasible is:

$$v_j = (v_i + r_{\kappa_{(i,j)}} + \delta_{(i,j)}) \bmod \tilde{v} \qquad \forall (i, j) \in A_T, \tag{1}$$

i.e. each tree arc must correspond to the sum of its referenced template arc plus its correction vector. The modulo-calculation is performed for simplicity in order to always stay within a finite ring. In this way, there is also no need for negative values in template arcs or correction vectors. The objective value of a solution is its number of template arcs, i.e. m.

3 Preprocessing

We approach the problem by first deriving a larger set R_{cand} of meaningful *candidate template arcs* and then select a smallest possible subset for which a feasible solution exists by optimization. The candidate set is derived in such a way that obviously redundant template arcs are effectively avoided. For an arbitrary arc $r \in \mathbb{D}$, let $Z(r) \subseteq A$ be the subset of arcs from A for which r can be used as representative so that corresponding correction vectors from the limited domain $\mathbb{D}'$ exist in order to fulfill condition (1). In particular, an arc r is considered redundant and not included in R_{cand} when it cannot be used for any arc in A, i.e. $Z(r) = \emptyset$, or it is dominated by some other arc r', i.e. $Z(r) \subset Z(r')$. Furthermore, from all arcs with exactly the same $Z(r)$, only one needs to be kept. We developed a dynamic programming procedure using a k-d tree data structure for efficiently determining R_{cand}.

With R_{cand} now available, our problem is related to the NP-hard *minimum label spanning tree* (MLST) problem introduced in [1]. Differences are, however, that we have to consider the directed case, multiple labels for each arc (corresponding to the suited candidate template arcs), and that not all but only k nodes need to be connected. We call this problem *Minimum Labeling k-Node Spanning Subtree Problem (k-MLNSS)*.

4 Metaheuristic Approaches

Based on the idea of the MVCA heuristic for the classical MLST problem in [1], we developed a greedy construction heuristic for our problem. Starting from an empty graph and an empty set of labels we iteratively add a label with its corresponding arcs to the graph until the graph contains a feasible k-node arborescence. The decision of which label to take next is based on the calculation of upper bounds on the numbers of further labels required to obtain a feasible arborescence. Advanced data structures are used in this procedure in order to avoid repeated time-consuming depth-first searches. The method runs in relatively short time; the quality of the results is, however, only moderate. Significantly better solutions, i.e. higher compression rates, can be achieved by a GRASP approach utilizing the construction heuristic. Hereby the next label to take is always chosen in a randomized fashion among a restricted set of candidates consisting of the best labels according to the determined bounds. A subsequent local search procedure tries to substitute some labels by others with the aim to reduce the number of required labels. Also, we modeled the problem as an integer linear program and were able to solve small to moderate instances to provable optimality. While this approach usually is too time-expensive for practical use, it could be verified that the GRASP metaheuristic is able to find optimal or near optimal solutions in most cases.

5 Summary and Outlook

Concerning the achieved compression ratios, the whole approach seems to be highly promising. However, more investigations on larger sets of test instances and with different minutiae matching algorithms are necessary in order to quantify how much fingerprint minutiae data can be compressed in order to still achieve reasonably low false match rates and false non-match rates. Furthermore, we are currently also pursuing other metaheuristics such as evolutionary algorithms for solving the k-MLNSS problem and variants of the described model.

References

[1] R.-S. Chang and S.-J. Leu. The minimum labeling spanning trees. *Information Processing Letters*, 63(5):277–282, 1997.

[2] T. Feo and M. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

[3] A. K. Jain and D. Maltoni. *Handbook of Fingerprint Recognition.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[4] S. O. Krumke and H.-C. Wirth. On the minimum label spanning tree problem. *Inf. Process. Lett.*, 66(2):81–85, 1998.