TELKOMNIKA, Vol.10, No.2, June 2012, pp. 335~342
ISSN: 1693-6930
accredited by DGHE (DIKTI), Decree No: 51/Dikti/Kep/2010

Review of Sequential Access Method for Fingerprint Identification

G. Indrawan*, B. Sitohang, S. Akbar
Data & Software Engineering Research Division
School of Electrical Engineering and Informatics, Bandung Institute of Technology
Jl. Ganesha No. 10 Bandung, West Java, Indonesia, Ph. +6222 2502260, Fax +6222 2534222
e-mail: gdindrawan@gmail.com*, benhard@stei.itb.ac.id, saiful@informatika.org

Abstrak

Identifikasi sidik jari waktu-nyata pada umumnya menggunakan arsitektur mesin komputasi spesifik untuk mengoptimalkan faktor kecepatan. Berfokus pada kinerja kecepatan yang lebih baik dari identifikasi sidik jari pada mesin komputasi umum, penyelidikan dilakukan pada metode akses sekuensial identifikasi sidik jari, dengan struktur data yang dirancang untuk bekerja dengan pemrosesan paralel. Berdasarkan hipotesis, pemrosesan paralel berbasis teknologi prosesor multi-core, mampu memberikan hasil lebih cepat tanpa mengurangi akurasi. Jika prosesor multi-core terdeteksi, beberapa proses akan berjalan simultan pada beberapa pasangan pencocokan sidik jari untuk menghasilkan nilai kesamaannya masing-masing. Eksperimen mengkonfirmasi kinerja kecepatan identifikasi sidik jari menggunakan metode akses sekuensial dengan pemrosesan paralel lebih baik dibandingkan tanpa pemrosesan paralel. Untuk kedua strategi, meskipun menggunakan pemrosesan paralel mengkonfirmasi hasil yang lebih cepat, eksperimen menunjukkan waktu pencarian O(n) masih bergantung secara linier pada jumlah sidik jari dalam database. Menghindari tren waktu pencarian tersebut, berdasarkan hipotetis, memerlukan strategi pemanfaatan metode akses langsung.

Kata kunci: *akurasi, identifikasi, kecepatan, pencocokan, sidik jari*

Abstract

Real time fingerprint identification is usually equipped with specific computation machine architecture to optimize speed factor. Focusing on achieving better speed performance of fingerprint identification on common computation machine, a disquisition was conducted on sequential access method for fingerprint identification, with its underlying data structure designed to work with parallel processing. Hypothetically, parallel processing based on multi-cores processor technology, can give faster result without reducing accuracy. If multi core processor was detected, simultaneous processes would run on fingerprint matching-pairs to find its similarity score, respectively. Experiment confirms that speed performance of fingerprint identification using sequential access method with parallel processing outperforms the one without parallel processing. For both strategy, even though using parallel processing confirms faster result, experiment shows that searching time O(n) still linearly depends on number of fingerprints in database. Avoiding such searching time trend, hypothetically, need strategy of direct access method utilization.

Keywords*: accuracy, fingerprint, identification, matching, speed.*

1. Introduction

A fingerprint identification system recognizes an individual by searching the entire enrolment templates in database for a match. It conducts one-to-many comparison/matching to establish if the individual is present in the database and if so, returns the identifier of the enrolment reference that matched. In an identification system, the system establishes a subject's identity (or determines that the subject is not enrolled in the system database) without the subject having to claim an identity [1]. This general concept applied to others biometric identification system, like palm print [2].

Figure 1 shows daily activities of a fingerprint identification system that consists of two main stages, i.e. enrolment and matching. On large database (could be several million fingerprints), identification demands real time result, so it is important to optimize its speed with respect to its accuracy by tuning up all of those main stages. Previously, we have

Received January 21, 2012; Revised April 30, 2012; Accepted May 7, 2012

reportedspeed optimization of fingerprint feature extraction in [3], which play important role during ten-print batch processing of enrolment andmatching.

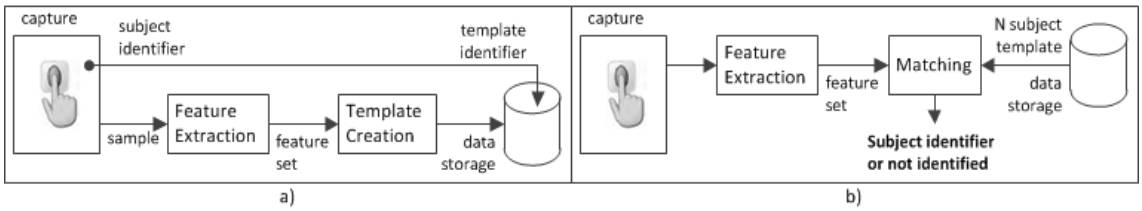

Figure 1. Fingerprint identification system: a) enrolment; b) matching

The identification process can be speeded up by reducing the number of comparisons that are required to be performed. Sometimes, information about sex, race, age, and other data related to the individual are available and the portion of the database to be searched can be significantly reduced; however, this information is not always accessible (e.g., criminal identification based on latent fingerprints) and, in a general case, intrinsic information of the fingerprint samples has to be used for an efficient retrieval.Searching involves lookup or indexing operations (finding the *value* associated with a *key*) using *index*. Prudent use of *index* can make searching faster by eliminating the need to sort (almost always the ultimate goal of sorting is to organize a search) and thus reducing I/O cost [4]. Searching involves sequential or direct access of data. Sequential access is the concept of accessing (or reading) records from a table in sequential order, i.e. from the top to bottom, one after another. Direct access is the concept of accessing (or reading) specific records from a table in no particular order by specifying which row(s) to be read. The row(s) value possibly comes from processing of specific data representation, like quad-tree, kd-tree, and range-tree [5].

Although hypothetically, direct access of fingerprint identification method would outperforms its sequential access method, it is however still important to know some aspects of sequential access method as a base for improvement. Sequential access method for finger print identificationis the most natural process of fingerprint identification thatbasically conducts one-to-many matching between input fingerprintand enrolled fingerprints in the database.Because of method's simplicity, method's most natural process, and method's strong foundation for improvement, a disquisition of this method was taking place based on [6], [7] and using [8] as a basic framework. While [6] gives translation- and rotation- variant features (i.e. minutiae's absolute coordinate and absolute orientation), [7] gives translation-invariant and rotation-variant features in star configuration (i.e. center-minutiae-to-neighbor-minutiae edge length, and center-minutiae's absolute orientation), [8] gives translation- and rotation- invariant features in star configuration (i.e. center-minutiae-to-neighbor-minutiae edge length, center-minutiae's relative orientation, and neighbor-minutiae's relative orientation), this paper observed parallel processing design that was implemented on [8] to gives improvement on fingerprint identification speed without reducing its accuracy.

2. Research Method

Sequential access method for fingerprint identification uses fingerprint's local features (minutiae-based only) without pre-selection stage (clasification process to produce pre-defined classes). Research design on this method involves:

a. Global data structure used by enrolment stage (Figure 2).
b. Sequential access method run above designated global data structure (Figure 3).
c. The algorithm and its underlying main object (Figure 4).
d. Derivative featurescomputation and comparison(Figure 5).
e. Similarity score mechanism from pairing traversal process (Figure 6).
f. Monte-Carlo-based experiment at Result and Discussion (Figure 7, Figure 8, Table 1).

2.1. Global Data Structure at Enrolment Stage

Enrolment uses main *Abstract Data Type* (ADT) called *Person* which is primarily a way to group multiple fingerprints belonging to one person. *Person* consists of *Fingerprint* object that

contains basic information about the fingerprint, i.e. *Image* that is used to perform template extraction and *Template* that is used for identification. *Image* is in raw image format that must be set before generating valid *Template*. The format of this image is a simple raw 2D array of bytes. Every byte represents shade of gray from black (0) to white (255). Template is an abstract model of the fingerprint and once was generated, Image property can be set to null to save space. Figure 2 shows memory space requirement for template contains fingerprint intrinsicinformation. Templates are better than fingerprint images, because they require less space and they are easier to match than images.

Person object is designed to be easy to serialize in order to be stored in binary-format (BLOB) under *Person* attributein database.This binary-format attribute indexed with numeric-format ID attribute (Figure 2). Here, we need to determine what *key*-like information can be used to construct data structure for searching. Additional requirement, this information should be translation- and rotation- invariant for reliability factor related to identification accuracy. Global structure data at Figure 2 do not contains translation- and rotation- invariant information (consider as primitive feature), but that information can be used to generate translation- and rotation- invariant *key*-like information (consider as derivative feature).

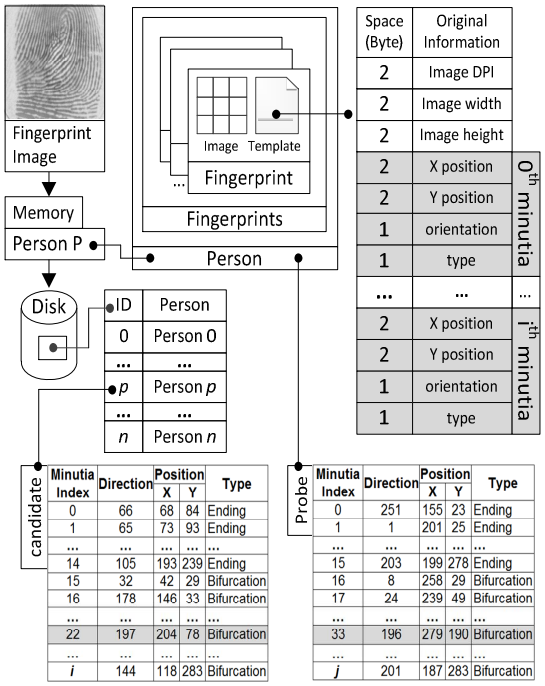

Figure 2. Global data structure of enrolment

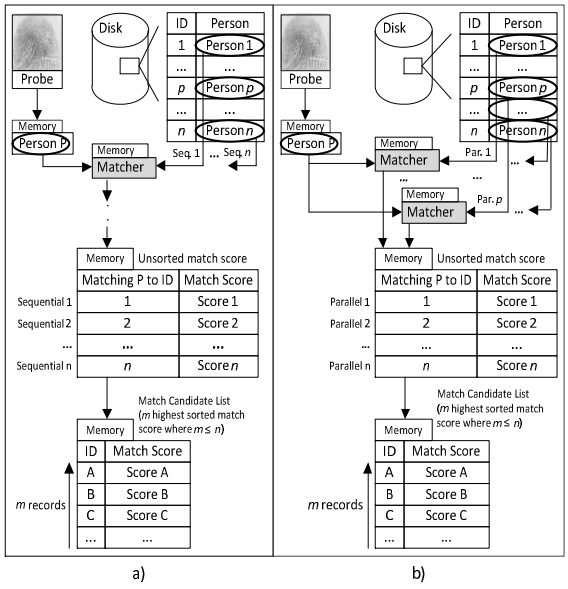

Figure 3. Sequential access method for identification: a) basic approach; b) improvement approach

2.2. Sequential Access Method at Identification Stage

Figure 3a shows basic approach of sequential access method for fingerprint identification. The matching is a kind of associative array with similarity measure. Probe fingerprint on input is compared sequentially with database's candidate fingerprints stored in the associative array, candidates are sorted by result of the similarity measure function, and the closest match is returned. Similarity measure function (matching algorithm) computes similarity score that represents degree of similarity between two templates.

Enhancement of basic approach (Figure 3b) utilizes *p* cores of processor so there will be *p* concurrent tasks (parallel process) on *p* fingerprint matching-pairs to produce *p*similarity scores, respectively. A fingerprint matching-pair consists of probe fingerprint and a candidate fingerprint from database to be match with.

2.3. The Algorithm

Identification stage at Figure 3 uses algorithm (Figure 4) that works integrated with its main *Abstract Data Type* called *Matcher*. *Matcher* consists of several objects, i.e. *ProbeIndex*

and *EdgeTable* to store derivative features belong to probe and candidate fingerprint, respectively. The others objects, i.e. *MinutiaPairing*, *EdgeLookup*, and *PairSelector* work together during *edge* pairing process to produce pair of the longest paths belong to probe and candidate fingerprint (Figure 6a). Traversal of candidate's longest path will give best similarity score as indicator for best similarity to the probe fingerprint. During traversal, some parameters values saved by *MinutiaPairing*, similar edges belong to probe and candidate saved by *EdgeLookup*, and decision for next minutiae pair to be travelled handled by *PairSelector*.

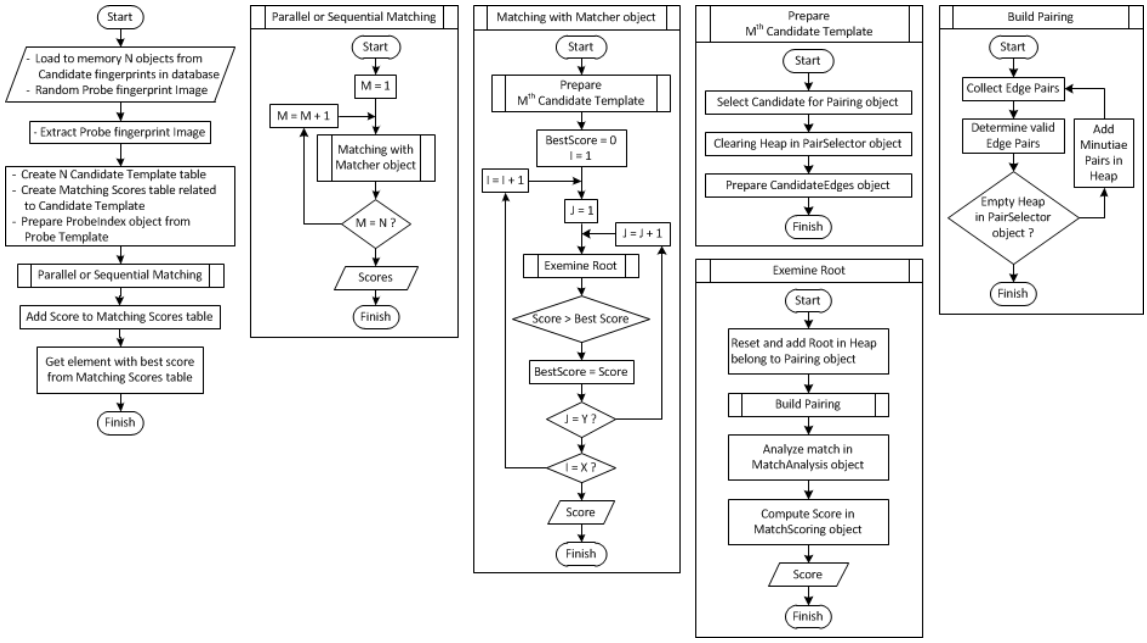

Figure 4. Algorithm of sequential access method for fingerprint identification

Several things are important to note regarding this algorithm:

a. Small size translation-androtation-variant minutia information (primitive features) are persisted in database, i.e. its location (*x*,*y*) and orientation *t*, as shown by Figure 2.
b. Relative big size translation-androtation-invariant minutia information (derivative features) from primitive features are compute in dynamic run-time memory for identification, i.e. its length of minutiae-edge *d*, and relative angle between minutiae – minutiae-edge (β_1 and β_2), as shown by Figure 5b.
c. Derivative features are constructed at *EdgeTable*belong to Matcher object. One table for the probe fingerprint (encapsulated by *ProbeIndex*) and one table for candidate fingerprint to be matched against.
d. There will be global computation time *O*(*n*) for identification where *n*is number of candidate fingerprints in database. For each matching between a probe and candidate fingerprint, there will be local computation time *O*(*i***j*) where *i* is number of records of probe's minutia information (primitive features) and *j* is number of records of candidate's primitive features, as shown by Figure 2.
e. The output will be pairing of the longest paths which are similar between probe path and its counterpart candidate path,as shown by pair of green paths at Figure 6a. These longest paths have its own *root* (consists of pair of minutiae index, one from probe and one from candidate), shown by gray record at Figure 2, as final objective of fingerprint identification. Traversal of the longest candidate path, start from its *root*, accumulate a similarity score (from computation of some parameters taken during traversal).

2.4. Derivative Features Computation

Figure 5b illustrates computation of translation-androtation- invariant minutia information (derivative features) from primitive features. Two minutiae connected by line construct edge. From left edge of Figure 5b, first minutia *k* is in the upper right and is depicted by the dot

representing location (x_k,y_k) and the arrowed line pointing down representing orientation t_k. A second minutia *j* is in the lower left with orientation pointing down and to the left.

To account for relative translational position, the distance d_{kj} is computed between the two minutia locations. This derivative feature will remain relatively constant between corresponding points on two different finger impressions regardless of how much translation and rotating may exist. Additional derivative featuresis angle between each minutia's orientation and the intervening line between both minutiae. This way, these angles remain relatively constant to the intervening line regardless of how much the fingerprint is rotated. In Figure5b, the angle θ_{kj} of the intervening line between minutia *k* and *j* is computed by taking the arctangent of the slope of the intervening line. Angles β_k and β_j are computed relative to the intervening line by incorporating θ_{kj} and each minutia's orientation *t*.

For each pair-wise minutia comparison, an entry is made into an edge table as shown by Figure 5c, i.e. consists of { d_{kj}, β_1, β_2}, where β_1 = max (β_k, β_j) and β_2 = min (β_k, β_j). So that in left edge illustration as shown by Figure 6b, $\beta_1 = \beta_k$ and $\beta_2 = \beta_j$. Entries are stored in the *Edge Table* belong to Matcher object. In ascending order of distance, the table is trimmed at the point in which a maximum distance and a maximum neighbour threshold are reached. Making these measurements between pairs of minutiae, an edge table must be constructed for each probe fingerprint and every candidate fingerprint we wish to match with.

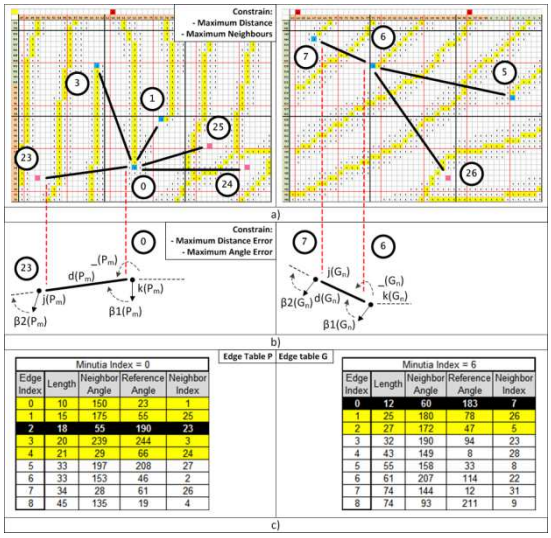

Figure 5. Intrinsic information from portion of the same fingerprint with slight different impressions: a) left: probe, right: candidate; b) derived translation- and rotation- invariant information of minutiae points; c) sample of fingerprint *key*-like information in tabular form where yellow records related to displayed entries at a) and black records indicate similar entries

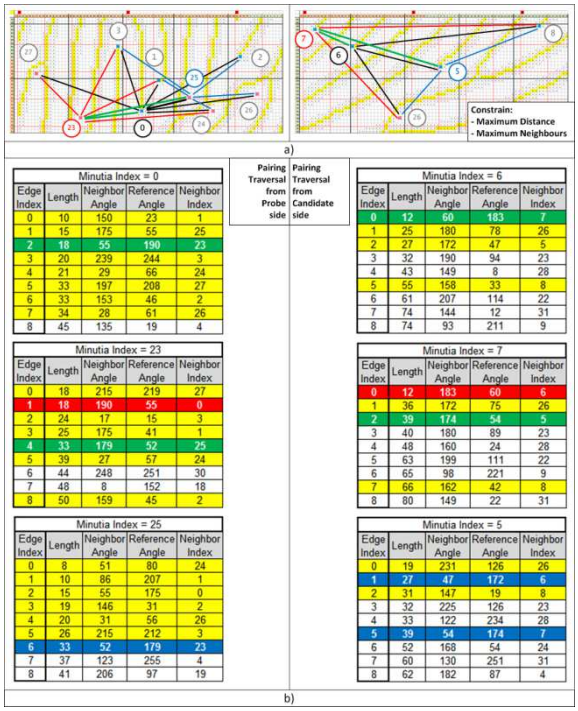

Figure 6. Pairing traversal: a) the longer the green path, the higher the similarityscore; b) pairing traversal information in tabular form

2.5. Derivative Features Comparison

Derivative feature comparison takes the minutia edge tables from two separate fingerprints and look for similar entries between the two tables. Figure5a shows part of two impressions of the same fingerprint with slight differences in both translation and rotation. The left print represents a probe impressionin which all its minutiae have been pair-wised computed with relative measurements stored in edge table P with sample (Figure 5c).

The relative measurements computed from the particular pair of minutia (Figure 5b) have been stored as the m^{th} entry in table *P*, denoted P_m. The notation of individual values stored in the table are represented as lookup functions on a given table entry. For example, the *index* of the upper right minutia is stored in table entry P_m and is referenced as $k(P_m)$, while the

distance between the two minutiae is also stored in table entry $P_m$and is referenced as $d(P_m)$. The right print represents a candidate impressionfrom database, and uses similar notation, except that all its pair-wise minutia comparisons have been stored in table G, and the measurements made on the two corresponding minutia in the candidate print have been stored in table entry G_n.

Black records at Figure 5c indicate similar entries. Three tests are conducted to determine if table entries P_m and $G_n$are similar. The first test checks to see if the corresponding distances are within a specified tolerance:Δ_d ($d(P_m)$, $d(G_n)$) $<T_d$.The last two tests check to see if the relative minutia angles are within a specified tolerance:Δ_β ($\beta_1(P_m)$, $\beta_1(G_n)$) $<T_\beta$, and Δ_β ($\beta_2(P_m)$, $\beta_2(G_n)$) $<T_\beta$.

Figure 6 shows part of process to calculate similarity score of a candidate fingerprint compared to the probe fingerprint. Maximum similarityscore produced by accumulatingsimilarityscore parameters during traversal of the longest path (maybe discontinuous path) of candidate fingerprint. Pair of green paths at Figure6a actually is not pair of the longest paths but it illustrates the process to find it at probe and candidate fingerprint. These longest paths have its own root -- consists of probe's minutia *index* and candidate's minutia *index*, shown by gray record at Figure2, as final objective of fingerprint matching.

Figure6b shows sample of pairing traversal information start from minutia pair (0,6) until minutia pair (25, 5). Yellow records show displayed edge at Figure6a (not all entries at edge table Figure6b shown at Figure 6a). Green, red, and blue records are similar edge pair (each from probe's edge and candidate's edge) belong to its minutia pair process during traversal. Green records also special entries that constructs pair of the longest path, each longest path for probe and candidate fingerprint.

2.5. Similarity Score

Similarity score between probe fingerprint and one candidate fingerprint from database, is obtained through traversal of pairing of the longest paths, each path belongs to the probe and candidate fingerprint to be match with (Figure 6). The longer the pair of these longest paths, the higher the similarity between those both fingerprints.

There are two conducted comparisons related to the similarity score, i.e.:

a. Finding pairing of the longest path through comparison with others pairing of the longest path found during iteration on probe and one candidate fingerprint.
b. After finding pairing of the longest path with the best similarity score (represent the best similarity between probe and one candidate fingerprint),that score put in unsorted associative array with Person ID as a *key*, and Similarity (Match) Score as the *value* (Figure3). Person ID represents *index* (location) of a candidate fingerprint in database and Match Score indicate candidate's best similarity score obtained during pairing traversal with probe fingerprint. The highest score in associative array become the strong similar candidate among others candidates fingerprint in database.

Some parameters for traversal score computation of the pairing of the longest path are minutiae pair count, correct type minutiae pair count, and supported minutiae pair count. The last two parameters are used for strict filtering of minutiae pair count forbetter deal with false matches that cause problems with low quality fingerprints.

3. Results and Discussion

Monte Carlo analysis was conducted on experiment, that rely on repeated random sampling to compute the results. The analysis is most suited for calculation by a computer and tends to be used when it is infeasible to compute an exact result with a deterministic algorithm [9].The analysis follows this procedure: 1) define a domain of possible inputs; 2) generate inputs randomly from a probability distribution over the domain; 3) perform a deterministic computation on the inputs; 4) aggregate the results.

In this procedure the domain of inputs is set A of database 1 (fingerprint image collected by using small-size and low-cost optical sensors) of fingerprint verification contest (FVC) 2000 [10], 2002 [11], and 2004 [12], each with 300x300, 388x374, and 640x480 pixel size. Then each database was set up with 100, 400, and 800 data population (every set A of FVC database has maximum value of 800 data population). So we have domain of inputs come from nine different databases (differ from FVC year and number of data population).

Experiment was conducted for each of nine data bases. We generate arbitrary number of random inputs (a hundred should be enough) per database then perform a computation on each input (searching/matching itself on database). Finally, we aggregate the results to obtain our final result per database, as shown by Table 1, i.e., the approximation of average matching time (*Avg*), sample standard deviation (*σ*), minimum matching time (*Min*), and maximum matching time (*Max*).

Sample standard deviation was calculated using formula with Bessel's correction [13] below(the use of $N - 1$ instead of N):

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2} \quad (1)$$

where $\{x_1, x_2, \ldots, x_N\}$ are the observed values (matching time) of the sample items and $\bar{x}$ is the mean value (*Avg*) of these observations. To get an accurate approximation for those all values, this procedure has two other common properties of Monte Carlo method. First, the inputs should truly be random. Second, there should be a large number of inputs.We conducted this Monte-Carlo-based experiment on sequential access method algorithm without parallel processing (single core) and with parallelprocessing (multi-cores) on Microsoft® Windows™ 7 machine with RAM 4 GB and Intel® Core™i5-2430M (Quad Cores)@ 2.20 GHz.

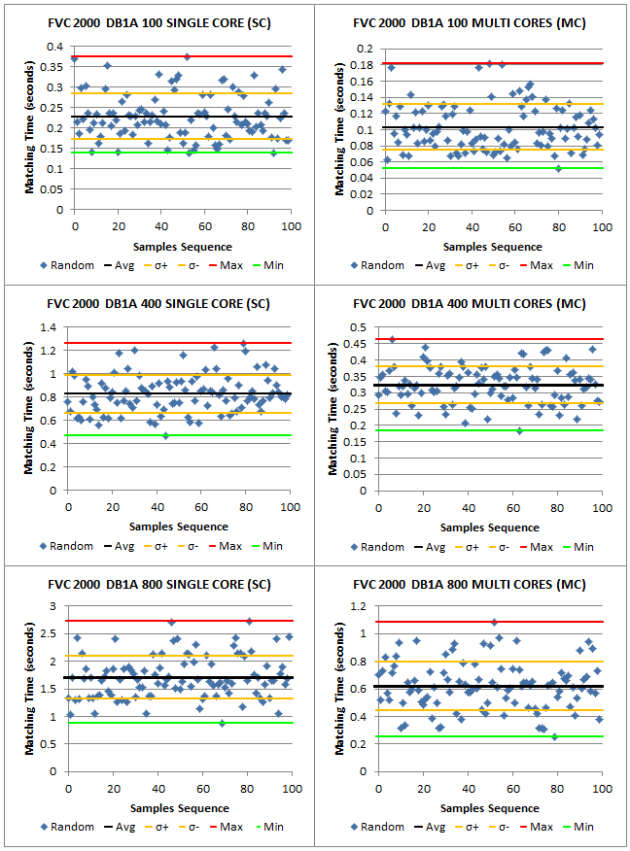

Figure 7. Monte Carlo analysis for sequential accessed method with single core and multi-coreson FVC 2000 database

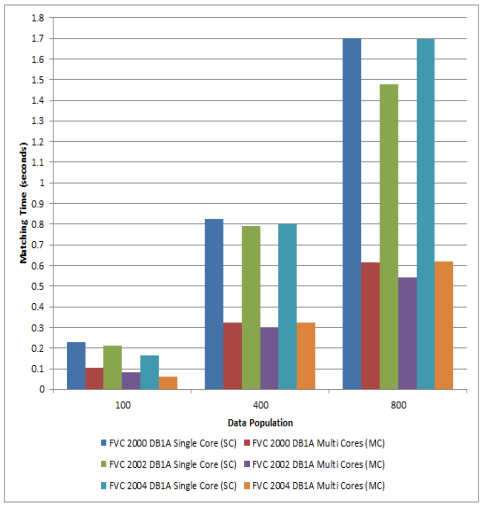

Figure 8. Speed performance comparison of sequential accessed method with single core and multi-cores of nine databases

Table 1. Parametersapproximation of sequential accessed method with single core and multi-cores of nine databases

Database \ Population	100				400				800			
	Avg	σ	Min	Max	Avg	σ	Min	Max	Avg	σ	Min	Max
FVC 2000 DB1A SC	0.23	0.06	0.14	0.37	0.82	0.16	0.47	1.26	1.70	0.39	0.87	2.73
FVC 2000 DB1A MC	0.10	0.03	0.05	0.18	0.32	0.06	0.18	0.46	0.62	0.18	0.25	1.08
FVC 2002 DB1A SC	0.21	0.04	0.13	0.32	0.79	0.16	0.44	1.26	1.48	0.30	0.54	2.30
FVC 2002 DB1A MC	0.08	0.01	0.05	0.13	0.30	0.07	0.18	0.49	0.54	0.13	0.23	0.87
FVC 2004 DB1A SC	0.17	0.04	0.07	0.28	0.80	0.23	0.43	1.48	1.70	0.44	0.85	3.62
FVC 2004 DB1A MC	0.06	0.02	0.02	0.12	0.33	0.10	0.15	0.58	0.62	0.17	0.30	1.40

Figure 7 shows detail of Monte Carlo analysis for sequential accessed method of fingerprint identification without and with parallel processing on FVC 2000 database. It shows approximation of *Avg*, sample standard deviation range (from *σ*- to *σ*+), *Min*, and *Max*. Complete results for other databases (FVC 2002 and FVC 2004) were shown on Table 1.Figure 8clearly shows that for all nine databases, sequential accessed method of fingerprint

identification with parallel processing outperforms the one without parallel processing. It also shows that even though using parallel processing gives faster result, in general searching time $O(n)$ is still dependsheavily on number of n fingerprints in database. Without parallel processing, searching time $O(n)$ is almost linearly depends on number of n fingerprints in three different tested database (FVC 2000, FVC 2002, and FVC 2004), while using parallel processing by quad core processor, searching time $O(n)$ is in sub linear trend ($o(n)$) where time increasing is only about one-third to a half of searching time $O(n)$ without parallel processing.

It can be analyzed the cause of the experiment result above (that also two main disadvantages of sequential access method) which are lied on its two big sequential accesses conducted during identification process, i.e.:

a. Sequential access to the whole n candidate fingerprints in database during matching probe fingerprint. Even though highest similarity score has already obtained from candidate fingerprint at very first location of database, the algorithm must still examine the rest of candidate fingerprints in database.
b. Sequential access to the whole minutia pairs [i,j] during matching probe to each candidate in database, where i is number of probe's minutia and i is number of candidate's minutia. Even though pair of the longest path (Figure 7a) has already obtained from the very first minutia pairs [0,0], the algorithm must still examine the rest of minutia pairs [i,j].

4. Conclusion

We have observed sequential access method forfingerprint identification without and with parallel processing. Theexperiment gives empiric resultsthat in general searching time $O(n)$ still dependsheavily on number of n fingerprint in database, i.e. in linear and sub linear fashion, respectively.With this result, we can predict what would happened if database contains millions number of fingerprints. The algorithm becomes unreliable with long time response without using parallel processing, and with parallel processing there must be some specific hardware implementation that should be avoided without doing algorithm optimization first.To overcome such kind of disadvantages of sequential access method for fingerprint identification, future work on designing and implementing of direct access method with various efficient representation of data structure, need to be observed.

References

[1] Maltoni D, Maio D, Jain AK, Prabhakar S. Handbook of Fingerprint Recognition.Second Edition.London: Springer -Verlag. 2009.
[2] Putra IKGD, Erdiawan. High Performance Palmprint Identification System Based On Two Dimensional Gabor. *TELKOMNIKA: Indonesian Journal of Electrical Engineering*. 2010; 8(3): 309-318.
[3] IndrawanG, Sitohang B, and Akbar S.*Parallel Processing for Fingerprint Feature Extraction*. International Conference on Electrical Engineering and Informatics (ICEEI). Bandung. 2011.
[4] Snell GP.*Think FAST! Use Memory Tables (Hashing) for Faster Merging*. SAS Users Group International (SUGI) 31 Proceedings. San Francisco. 2006.
[5] Samet H. The Design and Analysis of Spatial Data Structures. New York: Addison-Wesley. 1990.
[6] WatsonCI,Garris MD., Tabassi E, Wilson CL, McCabe RM, Janet S, Ko K. *User's Guide to Export Controlled Distribution of NIST Biometric Image Software (NBIS-EC)*. National Institute of Standards and Technology. 2001.
[7] Ratha NK, Pandit VD, Bolle RM,Vaish V. *Robust Fingerprint Authentication Using Local Structural Similarity*.Proc. Workshop on Applications of Computer Vision. 2000: 29–34.
[8] R. Vazan. SourceAFIS- Open Source Automatic Fingerprint Identification System onSourceForge. http://sourceforge.net/projects/sourceafis/
[9] Anderson HL. Metropolis, Monte Carlo and the MANIAC.Los Alamos Science. 1986: 14: 96–108.
[10] Maio D, Maltoni D, CappelliR, Wayman JL, Jain AK. FVC2000: Fingerprint Verification Competition.*IEEE Transactions on Pattern Analysis Machine Intelligence*.2002; 24(3):402-412.
[11] Maio D, Maltoni D, Cappelli R, Wayman JL, Jain AK. *FVC2002: Second Fingerprint Verification Competition*.Proc. 16th Int'l Conf. Pattern Recognition.2002;811-814.
[12] Maio D, Maltoni D, Cappelli R, Wayman JL, Jain AK. *FVC2004: Third Fingerprint Verification Competition*.Proc. Int'l Conf. Biometric Authentication.2004; 1-7.
[13] Montgomery DC, Runger GC. Applied Statistics and Probability for Engineers.Third Edition.New Jersey: Wiley and Sons. 2003.