The Development of Hashing Indexing Technique in Case Retrieval

Mohamad Farhan Mohamad Mohsin
UUM College of Arts & Sciences
Universiti Utara Malaysia
Kedah, Malaysia
farhan@uum.edu.my

Norita Md Norwawi
Faculty of Science & Technology
Universiti Sains Islam Malaysia
Bangi, Selanor
norita@usim.edu.my

Maznie Binti Manaf
Fakulti Sains Komputer dan Matematik
Universiti Teknologi Mara (Kelantan)
Machang, Kelantan
maznie@kelantan.uitm.edu.my

Mohd Helmy Abd Wahab
Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn
Johor, Malaysia
helmy@uthm.edu.my

Abstract—**Case-based reasoning (CBR) considers previous experience in form of cases to overcome new problems. It requires many solved cases in case base in order to produce a quality decision. Since today, database technology has allowed CBR to use a huge case storage therefore the case retrieval process also reflects the final decision in CBR. Traditionally, sequential indexing method has been applied to search for possible cases in case base. This technique is worked fast when the number of cases is small but it consumes more time to retrieve when the number of data grows in case base. To overcome the weakness, this study researches the non-sequential indexing called hashing as an alternative to mine large cases and faster the retrieval time in CBR. Hashing indexing searches a record by determines the index using only an entry's search key without traveling to all records. This paper presents the review of a literature and early stages of the integration hashing indexing method in CBR. The concept of hashing indexing in case retrieving process, the model development, and the preliminary algorithm testing result will be discussed in this paper.**

Keywords- hashing indexing, sequential indexing, case retrieval, temporal data

I. INTRODUCTION

Case-based reasoning (CBR) is rapidly growth and widespread across in many domain. It has been applied in many fields such as medical for diagnostic and therapeutic task, image retrieval, treatment, planning, and tutoring [1,2]. As a data mining technique, CBR considers previous experience in form of cases to understand and overcome new problems. To derive conclusion, it executes four steps that are retrieve the most similar cases, reuse the retrieved cases to solve the problem, revise the reused solution, and finally retain the revised experience in case base for future decision making. The concept of CBR is the similarity between old and new case therefore it requires many previous cases in case base in order to provide a good decision [3]. To achieve that, it relies heavily on the quality of old cases but to obtain a quality case is difficult [4].

Nowadays, the capability of database technology to store gigantic records has allowed CBR to have a huge number of cases in case base. Because of the big cases, many researchers have undertaken study on case retrieval mainly on the case indexing technique for faster retrieval time.

The most common technique for similarity retrieval in CBR is the k-nearest neighbor (k-NN) [6]. According to [5], the k-NN is not an efficient method to minimize retrieval time particularly at case similarity retrieval. They suggested the indexing technique which is commonly used in database application that can help speed up retrieval and optimize accessibility of data. Indexing is a technique used to speed up access of stored data. By storing the data in the right way, it helps the system to find items or element without having to scan the entire data set [7]. There are several techniques for case retrieval which include indexing schemes such as kd-trees, Fish-and-Sink approach, the CRASH memory model, Case Retrieval Nets [8, 9] and Hashing [10].

Generally, there are two types of indexing structures which are sequential and non-sequential indexing. Traditionally, sequential indexing has been applied to search for possible cases in case base. Through sequential technique, cases are retrieved case by case following a sequence until the most similar case is matched. It works fast when the number of cases is small but it will consume more time to retrieve when there are huge cases in case base.

Therefore, this study researches the non-sequential indexing called hashing as an alternative to cater large cases and faster the retrieval time in CBR. Hashing indexing searches a record by determining the index using only an entry's search key without traveling to all records [10]. It utilizes small memory, faster retrieval time, and easier to

978-1-4244-6716-7/10/$26.00 © 2010 IEEE

code compared to other indexing technique like data structure [11]. This paper presents the review of a literature and early stages of the integration of hashing indexing method in CBR in order to improve the performance of case retrieval. Then, hashing indexing performance will be compared with sequential method in term of retrieval time and accuracy.

This paper is organized as follows. Section 2 outlines the literature of hashing indexing technique. Then, the integration of hashing indexing technique in CBR is discussed in section 3. It will be followed by a discussion on the model development of the study in Section 4. Section 5 reports the preliminary algorithm testing and final sections conclude this work.

II. HASHING INDEXING

Hashing indexing is commonly used in database application during data retrieval. The idea of hashing is to map each records to a hash key (x) using hash function; $H(x)$. $H(x) = i$ whereby i is an index to the hash table (HT) and $HT = [1 \ldots n]$. HT represents an array of size n. The $H(x)$ will take a search key and produces an integer type index representing each case in HT. After that, the case can be directly retrieved at respective address from the HT. The address or search key, x is generated from the function $H(x) = x \bmod n$ whereby x is the search key, n is the table size and *mod* is the modulo operator.

In practice, $H(x)$ can map two or more addresses into HT. The $H(x)$ is capable to store more than two items in the same array location in HT or tends to open other address. This occurrence is called collision and the item involved are often called as synonyms [12]. An example of hashing indexing technique which is adopted from [13] is shown in Fig 1.

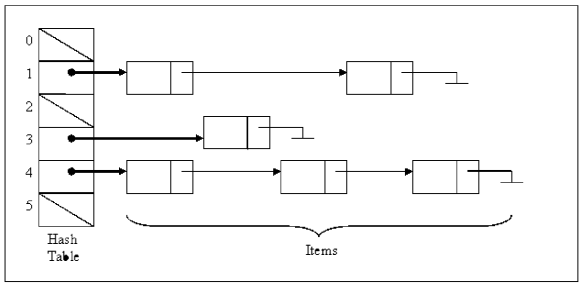

Figure 1. Hasing indexing technique

One of the HT limitations is when the records become full. It will start working very badly unless separate chaining which capable to handle collision is used. This is the reason why [12] suggested that HT should never be allowed to get full. To determine either HT is full, the ratio of the number entry located in HT need to be calculated. The ratio is known as load factor. Generally, HT size should be automatically increased and the records in the table should be rehashed when the ratio of table is reached 0.7 (70% full) or 0.8 (80% full) for open addressing [10, 12].

Recently, many applications utilized hashing mechanism to solve specific problem such as in programming that use HT to keep track of declared variables in source code [10, 13, 12]. HT is an ideal application for this kind of problem because only two operations are performed *insert* and *find*; identifiers are typically short, so the $H(x)$ can be computed quickly. In this application, most searches are successful.

Another common use of HT is in game programs. As the program search through different lines of play, it keeps track of positions that it has encountered by computing $H(x)$ based on the position (and storing its move for the position). If the same position recurs, usually by a simple transposition of moves, the program can avoid expensive recalculation.

III. INTEGRATING HASHING INDEXING IN CASE RETRIEVAL

From literatures, hashing indexing can faster the retrieval time and minimizes the usage of computer resources; therefore it has high possibility to integrate hashing technique with CBR since CBR also performs case retrieval from case base. Fig. 2 depicts the concept of this technique and Fig. 3 is a sequential indexing method. Sequential indexing is a conventional technique practiced in CBR's case retrieval.

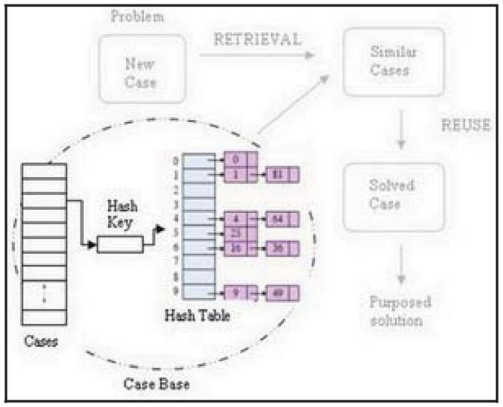

Figure 2. Integration of hashing indexing teachine in CBR

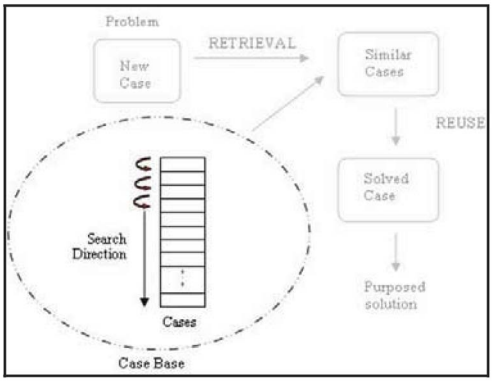

Figure 3. Integration of seqeuntial indexing teachine in CBR

In the proposed hash model, the separate chaining or close addressing is chosen to resolve collisions. Through this method, specific location is allowed to store more than one value called bucket, b. A good $H(x)$ is should be fast to compute, minimize collisions, and distribute entries uniformly through the HT.

In this study, three search keys, x are defined. The x are mean of average rainfalls (m), change water level (ΔWL), and combining mean average rainfall and change water level ($\overline{R_m}$ ^ ΔWL) as written in (1). Different x are used to determine which x will produces better result mainly in high accuracy and low time retrieval.

$$H(x) = \begin{cases} \overline{R_m} \bmod n \\ \Delta WL \\ \overline{R_m} \ \hat{} \ \Delta WL \end{cases} \quad (1)$$

Where n is the table size, $\overline{R_m}$ refers Eq (2), ΔWL refers Eq. (3)

Eq. 2 shows the formula to calculate the combination of mean average rainfall and change water level key, $\overline{R_m}$ and Eq 3 is a formula to calculate change water level key (ΔWL).

$$\overline{R}_m = \frac{\overline{R}_{t-1} + \overline{R}_{t-2}}{2} \quad (2)$$

where $\overline{R_{t-1}}$ is the average rainfall at time t -1, $\overline{R_{t-2}}$ is the average rainfall at time t -2, t is the time index

$$\Delta WL_t = \frac{WL_{t-1} + WL_{t-2}}{WL_{t-2}} \quad (3)$$

where $WL_{t-1}$is the average rainfall at time t -1, WL_{t-2} is the average rainfall at time t -2, t is the time index

Moreover, every types of x will have different size of hash table or called bucket, b. The number of b is depends on the type of its x. For example the change of water level (ΔWL) has three types of water level, which are *Alert*, *Warning* and *Danger* [15]. Therefore, ΔWL will have three buckets. Table 1 shows the ΔWL key, the number of bucket, and the range of case ΔWL. From table I, figure 4 represents the bucket arrangement of ΔWL.

TABLE I. TYPE OF CHANGE WATER LAVEL (ΔWL)AND THE NUMBER OF BUCKETS

Bucket (b)	Type of water level (ΔWL)	Range of ΔWL / m
0	Alert	$x \leq 0.0034$
1	Warning	$0.0034 < x < 0.0061$
2	Danger	$x \geq 0.0061$

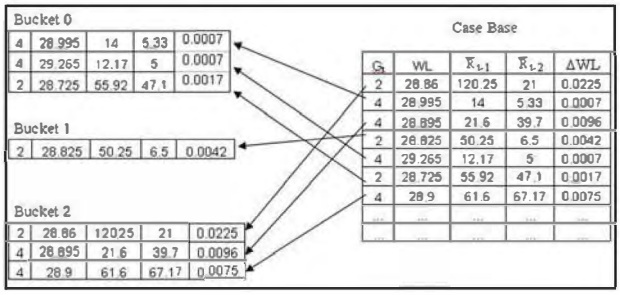

Figure 4. The Bucket Arrangement Using ΔWL key

For mean of average rainfall (m) key, it has four buckets which represent type of rainfall; there are *Light*, *Moderate*, *Heavy* and *Very Heavy*. Table II elaborates the type of rainfall while Fig. 5 represents the bucket arrangement of m.

TABLE II. TYPE OF RAINFALL AND THE NUMBER OF BUCKETS

Bucket (b)	Type of Rainfall (m)	Range of Rainfall / mm
0	Light	$x \leq 11$
1	Moderate	$11 < x < 32$
2	Heavy	$32 < x < 62$
3	Very Heavy	$x \geq 62$

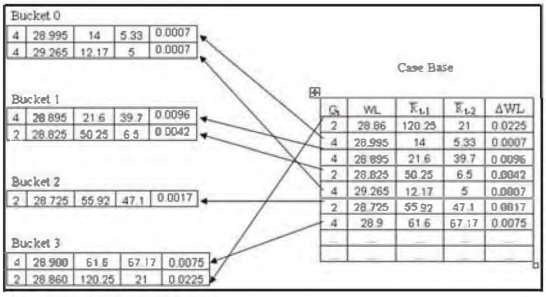

Figure 5. The Bucket Arrangement Using m's search key

Generally, the modified hashing indexing algorithm in CBR involves two main tasks that are storing new cases and retrieving a case. Process flow in Fig.6 represents the process of storing a new case in case base. It starts with calculation of the $H(x)$ and then store the current cases in bucket.

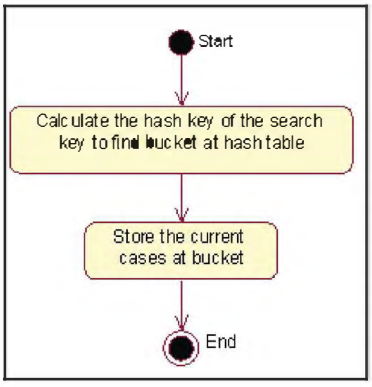

Figure 6. Storing a new case

The case retrieving process based on CBR's hashing indexing is shown in Fig. 7. It starts with calculating the hash key and map to the HT. The result will be retrieved by finding the *x* after entering a new case. The *H(x)* formula is used to find the address in *HT*. Finally, the similarities of cases in similar bucket will be calculated to get the predicted result.

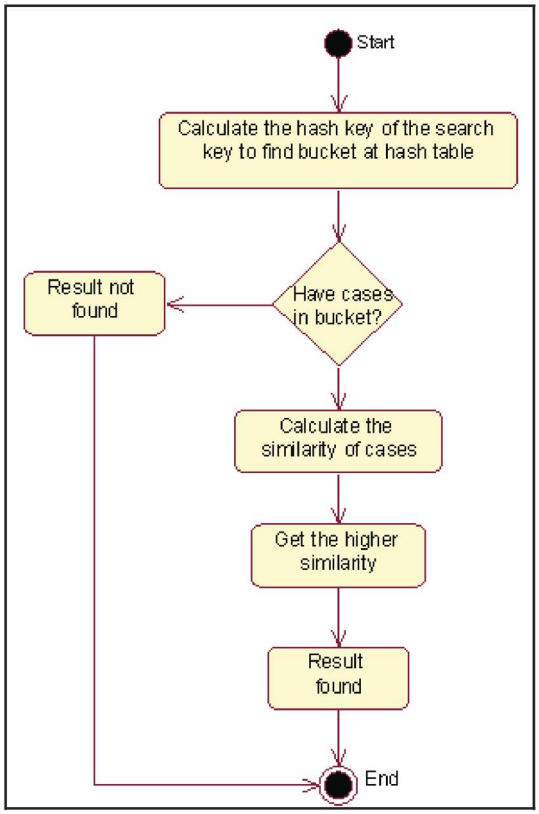

Figure 7. *Retrieving Cases in Buckets at Hash Table*

IV. MODEL DEVELOPMENT

This section will explain the model development of the study. This study is an experimental approach which is divided in two phases. First phase is algorithm development and the second phase is mining phase. Fig. 8 illustrates the experimental design in this study.

A. Algorithm Developemt

The algorithm development phase focuses on the algorithm modification. This phase covers three steps which are design development, implementation and testing. In the design development, two approaches: sequential indexing and hashing indexing technique are designed and integrated into CBR using Microsoft Visual C++. After that, the model will be tested. The aim of the testing is to check the accurateness of the hash table and the similarity calculation during mining. The result of the testing is reported in section V (Preliminary algorithm testing)

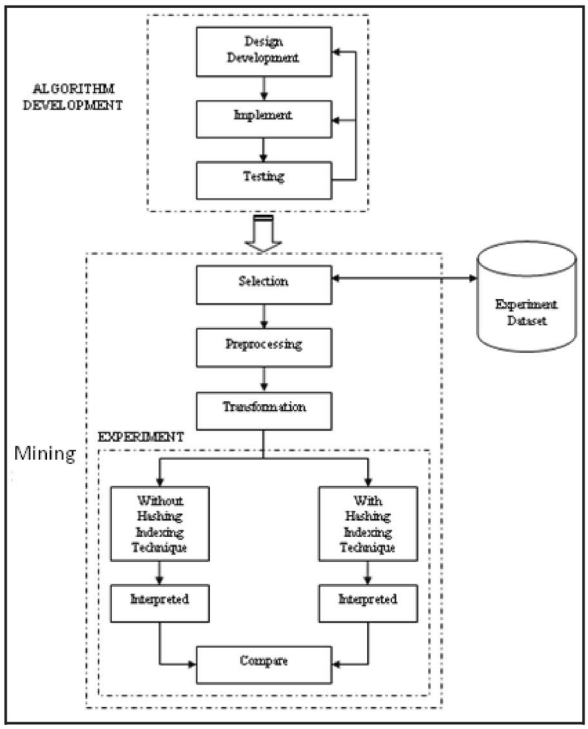

Figure 8. Model Developemnt

B. Mining Phase

Mining refers to the process of extracting knowledge using CBR algoithm. There are two main process in mining that are preparing data for mining and knoweldge extraction.

1) Preparing data for mining

In this part, it will cover three steps that are data selection, data pre-processing, and data transformation. The aim of this process is to clean and prepare the data before presenting it into the CBR mining system.

The experiment dataset set used in this study is a 15 attributes' temporal dataset called Timah Tasoh Dam dataset. It comprises the historical hydrological data of daily Timah Tasoh dam operation in Perlis, Malaysia in the year 1997-2005. The preliminary observation on the raw dataset, some attributes are not related to study and certain values were missing. Therefore, the dataset are pre-processed using temporal data series approach which was adopted from [14,15].

During data preprocessing, only relevant attributes will be selected. Out of 15 attributes, 4 attributes are chosen that are current water level, average rainfall, current change water level, and current gate. Those attributes represents reservoir water level, rainfall measurement from 6 telemetry stations (Padang Besar, Tasoh, Lubuk Sireh, Kaki Bukit, Wang Kelian, and Guat Jentik) and the number of spillway gates. Spillway gate refers to a structure that makes it possible for

the excess water to be released from the dam. Timah Tasoh has six gates and normally the water will be released using Gate 2, Gate 4, and Gate 6 depends on the situation. The selection is made using sliding window technique which is adopted from [14, 15]. After that, the data are re-scaled into a suitable representation to increase mining speed and minimize memory allocation. Table III is a sample of clean data which is ready for mining using CBR's hashing indexing and CBR's sequential indexing model. Based on the table, the current water level (WL_t), average rainfall at t-1 and t-2, current change water level (ΔWL_t) and current gate (G_t) are the final input to be mined using CBR.

TABLE III. CASE BASE SAMPLE FOR MINING

Gate	Water Level	Average Rainfall (t-1)	Average Rainfall (t-2)	Change Water Level
2	29.275	7.33	5.375	0.0007
2	28.86	120.25	21	0.0225
2	29.145	28.125	9.5	0.0059
4	29.025	22.75	11	0.001
4	28.825	50.25	6.5	0.0042
6	28.46	3.5	10.1	0.0018

2) *Knowledge Extraction*

In this part, Timah Tasoh Dam dataset will be mined. Both models-hashing and sequential indexing model will use similar case base and their performance will compared. The evaluation is based on two metrics that are accuracy and retrieval time.

In order to measure the accuracy, the algorithm is tested using various data partition by taking cases in case based as a test set. The measurements are adopted from [15] as shown in Table IV. This is due to the fact that the real datasets consists of unbalanced data where the number of occurrences of event is lower as compared to non-event occurrence.

TABLE IV. PERFORMACE MATRCIS ADOPTED FROM NORWAWI (2004)

Measurement	Meaning
t_p	True positive. Number of event correctly predicted
f_p	False positive. Number of predicted event but in actual non-even
t_n	True negative. Number of non-event correctly predicted
f_n	False negative. Number of predicted non-even but in actual even
Sensitivity (*Sen*)	The accuracy of correctly predicted event $= \frac{t_p}{t_p + f_n}$
Specificity (*Spec*)	The accuracy of correctly predicted non-event $= \frac{t_n}{t_n + f_p}$
Total Accuracy (*Acc*)	The ratio of total correct prediction $= \frac{t_p + t_n}{t_n + f_n + t_p + f_p}$

Second measurement is retrieval time which refers to time taken to search for the similarity case from case base. The time is tested by selecting one case from case base and the selected case will be measured for both hashing and sequential technique. The retrieval time will be recorded five times before calculate the average. A special loop is used to perform the task as shown in coding in Fig 9.

```
Double start = clock();

//dummy looping
for (i=0; i<900000; i++){
      for(i=0; i<column; i++){
            for (j=0; j<row; j++){
                  oldCases[i][j];
}}}

/***************************************
    CBR ENGINE (get the global similarity)
*/

Double end = clock();

//calculate retrieval time
cout << Time Taken :: " << (end-start)/60 << millisecond
```

Figure 9. Algorithm to calculate retrival time

V. PRELIMINARY ALGORITHM TESTING

This is an early stage of the project and this section represents the testing of the hashing and sequential indexing models. The aim of the testing is to check the capability both models can generate an accurate hash table and similarity calculation before starts mining the Timah Tasoh Dam dataset.

To check the accuracy of the hash table, the buckets of each case are listed and the test cases in case base will be map to the listed bucket. If the value of actual bucket and suggested bucket are equal, then the models can be concluded have generated an accurate result. Table V represents the sample of the test case and Table VI shows the accuracy of hash table using average rainfall (*m*) search key. From the table, the actual bucket and suggested bucket are similar. This shows the model generates an accurate hash table.

TABLE V. TEST CASE SAMPLES

Cases No	Gate	WL	$\overline{R}$ t-1	$\overline{R}$ t-2	ΔWL
1	2	28.860	120.25	21.00	0.0225
2	4	28.995	14.00	5.33	0.0007
3	4	28.895	21.60	39.70	0.0096
4	2	28.825	50.25	6.50	0.0042
5	4	29.265	12.17	5.00	0.0007
6	2	28.725	55.92	47.10	0.0017
7	4	28.900	61.60	67.17	0.0075

TABLE VI. THE TEST RESULT OF HASH TABLE ACCURACY USING AVERAGE RAINFAKK (M) KEY

Cases No	Suggested Bucket	Actual Bucket	Accurate
1	3	3	Yes
2	0	0	Yes
3	1	1	Yes
4	1	1	Yes
5	0	0	Yes
6	2	2	Yes
7	3	3	Yes

** Yes means >95% accurate for cases in the right bucket of hash table

The second test is the accurateness of the case similarity calculation. From the testing, the model has shown it can generate an accurate result since the similarity of the test is more than 95%. Table VII shows the test result in term of case similarity.

TABLE VII. THE TEST RESULT OF MODEL ACCURACY IN TERM OF CASE SIMILARYTY

Cases No	Suggested Open Gate	Actual Open Gate	Similarity %
1	2	2	95.7
2	4	4	98.0
3	4	4	95.2
4	2	2	97.4
5	4	4	98.9
6	2	2	97.3
7	4	4	99.1

Since both testing has shown a good result, it can be concluded that the modified models-hashing indexing and sequential indexing in CBR are ready for mining the experiment data.

VI. CONCLUSION

This on-going research consists of two stages. The first stage is designing the hash indexing algorithm in case retrieval function. The second stage is to compare the hashing indexing case retrieval with sequential case retrieval in term of accuracy and retrieval time. To achieve that, temporal dataset named Timah Tasoh Dam dataset will be selected as a case study. This project is purposely conducted to offers an alternative technique for case base representation and case retrieval. The finding is aimed can assist future miner to mine cases faster, obtain better accuracy and minimize the computer resources usage.

REFERENCES

[1] Yang, Z., Matsumura, Y., Kuwata, S., Kusuoka, H., & Takeda, H. (2003). Similar Cases Retrieval From the Database of Laboratory Test Results. Journal of medical systems (J. med. syst.). Vol. 27, No. 3: pp. 271-282

[2] Schmidt, R., & Gleri, L. (2000). Case-based Reasoning for Medical Knowledge-based Systems. International Journal of Medical Informatics, Vol .64, pp 355.

[3] Armengol, E., Ontanon, S., Plaza, E. (2002). Explaining Similarity in CBR. *Artificial Intelligence Review.* Vol. 24, No. 2.

[4] Rong, P., Qiang, Y., Sinno, J.P. (2007). Mining competent case bases for case-based reasoning. Artificial Intelligence, Elsevier Science Publishers Ltd. Vol. 171 .

[5] Kang, S. H. & Lau, S. K. (2002). Intelligent Knowledge Acquisition with Case-Based Reasoning Techniques. Retrived December, 2009 from:http://www.dsl.uow.edu.au/publications/techreports/kang03intelligent.pdf.

[6] Mantaras, R. L. D., Mcsherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M. L., Cox, M. T., Forbus, K., Keane, M., Aamodt, A., & Watson, I. (2005). Retrieval, Reuse, Revision, and Retention in Case-Based Reasoning. The Knowledge Engineering Review. Vol. 20, No. 3: pp.215-240.

[7] Andre-Jonsson, H. (2002). *Indexing Strategies for Time Series Data.* Linkopings Universitet, Linkoping, Sweden.

[8] Lenz, M. (1996). Case Retrieval Nets Applied to Large Case Bases. Proc. 4th German Workshop on CBR, Berlin.

[9] Lenz, M., Burkhard, H., & Burckner, S. (1996). Applying Case Retrieval Nets to Diagnostic Tasks in Technical Domains. Proceedings of the Third European Workshop on Advances in Case-Based Reasoning , pp: 219 – 233.

[10] Carrano, F. M., & Savitch, W. (2003). Data Structures and Abstractions with Java, Paerson Education, Inc:USA.

[11] Griebel, M., & Zumbusch, G. (1998). Hash-Storage Techniques for Adaptive Multilevel Solvers and Their Domain Decomposition Parallelization. Proceedings of Domain Decomposition Methods 10, DD10.

[12] Maurer, W. D., & Lewis, T. G. (1975). Hash Table Methods. ACM Computing Surveys (CSUR), Vol. , No 1, pp:5-19.

[13] Darus, N. M., Yusof, Y., Mohd, H., & Baharom, F. (2003). *Struktur Data Dan Algoritma Menggunakan Java* (First Edition ed.). Selangor, Malaysia: Pearson Prentice Hall.

[14] Hassin, M. H. B. M., Norwawi, N. B. M., & Aziz, A. B. A. (2006). *Sliding Window as a Segmentation Technique for Temporal Data Preprocessing.* Proceeding of the Regional Computer Science Postgraduate Conference (ReCSPC '06).

[15] Norwawi, N. B. M. (2004). Computational Recognition - Primed Decision Model based on Temporal Data Mining Approach for Reservoir Flood Control. PHD Thesis, Universiti Utara Malaysia, Malaysia