



THE HONG KONG
POLYTECHNIC UNIVERSITY

香港理工大學

Pao Yue-kong Library
包玉剛圖書館

Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact lbsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.



THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

A System for Document Image Classification

(1996.12/1997.12)

Supervisor: Paul Bao, Keith Chan

Student Name: Huang Xudong



ABSTRACT

In this thesis, the system for Document Image Classification is discussed in detail with three parts, i.e., document segmentation, and block classification and document classification. In fact, only document segmentation and document classification is used. The block classification does not need. The reason for it has been analyzed clearly. For the segmentation, the Fuzzy Segmentation Method (FSM) for the analysis of document images is developed. Based on the uses of the fuzzy set theory, the fuzzy blank and fuzzy black blocks are defined. The idea of λ -cut sets is utilized when determining the vertical and horizontal thresholds in the segmentation process. With the definition of fuzzy sets, such a process can be made automatically in an adaptive manner. For the block classification, a system for automated pattern analysis and classification (APACS) method is applied to the block classification. The algorithm, features, and selection of the samples for using this method are specified. Furthermore, the part of affirming the feature of the blocks is omitted. The document classification algorithm in the third part employs the features of each block as parameters. This can reduce the error of recognition. The results are very satisfied. For the document classification, the Branch-and-Bound Technique is adopted to match two Attributed Random Graphs (ARGs) of a document. A new evaluation function of the Branch-and-Bound algorithm is proposed. ARG construction, decision tree construction and using branch-and-bound to match two ARGs are elaborated in detail. The document classification is successfully transferred to how to match the two ARGs. The experimental results are quite satisfied.

TABLE OF CONTENTS

ABSTRACT	ii
ACKNOWLEDGEMENT	v
Chapter 1 Introduction	1
1.1 Problem Statement.....	2
1.2 Overview of the results	3
1.2.1 Segmentation	3
1.2.2 Block Classification.....	5
1.2.3 Document Classification	9
1.3 Fuzzy Methodology.....	10
Chapter 2 Technique for Pattern Recognition	14
2.1 Statistical Pattern Recognition.....	14
2.2 Structural Pattern Recognition.....	14
2.3 Pattern Representation.....	14
2.4 Graph Matching and Graph Isomorphism.....	15
2.5 Attributed Random Graph.....	16
Chapter 3 Fuzzy Segmentation for Document Image Analysis	17
3.1 Introduction to the segmentation of document classification.....	17
3.2 The Fuzzy Segmentation Method (FSM)	20
3.3 Automatic Determination of Thresholds.....	22
3.4 Features of the FSM.....	25
3.5 Paragraph Block	26
3.6 Experimental Results.....	27
3.7 Summary.....	27
Chapter 4 Block Classification of Document	31
4.1 Introduction.....	31
4.2 Block Classification Using ACAPS.....	34
4.3 The Algorithm of ACAPS.....	35
4.4 The features of using the ACAPS	37
4.5 The Implementation of using the ACAPS.....	38
Chapter 5 Document Classification Using Branch-and-Bound Technique Based on ARG Matching	41
5.1 Introduction.....	41
5.2 Construct Attribute Random Graph.....	41
5.3 Construct Decision Tree	42

5.4 ARG matching Use Branch-and-Bound Technique.....	42
5.5 Application Algorithm for Matching ARG.....	44
5.6 Whole Procession of DIC.....	45
5.7 Experimental Results.....	45
5.8 Conclusion	47
Chapter 6 Document Classification Based on Advanced ARG	
Matching.....	54
6.1 Shortcoming of the whole procession	54
6.2 Construct Attribute Random Graph.....	54
6.3 Construct Decision Tree	55
6.4 ARG matching Use Branch-and-Bound Technique.....	56
6.5 Application Algorithm for Matching ARG.....	57
6.6 Experimental Results.....	58
Chapter 7 Conclusion	64
7.1 Summary.....	64
7.2 Future Work.....	65
Bibliography	67
Appendix A Shell Commands for Testing and Training in UNIX	
Appendix B All Combinations' Testing and Training Results	
Appendix C Representative Samples	
Appendix D Part Source Code in C++	
Appendix E The Pattern and Process Result Files	

ACKNOWLEDGEMENT

Dear Madam/Sir:

I am very thankful and honorary for you reading this clumsy and unskilled thesis. From Nov. 1995 to Dec. 1997, I am as a Research Student for MPhil Degree in Department of Computing, Hong Kong Polytechnic University. Undertaking a research project entitled "Simulating the Behavior of Software Modules from their Specifications" for one year because my supervisor Dr. Wang Yabo leave there. In that time, I had almost completed the project. I am very thank for Dr. Keith Chan, and Dr. Paul Bao given me a chance to continue my research work and given me much more support. With their help, I had completed this project entitled "Document Image Classification by Layout Analysis (Rename to "A System for Document Image Classification")". In this period, I did mainly research work and used the C++ in UNIX to simulate the project. The following is summary for my work within one year:

- 1) Survey of the previous techniques about pattern recognition in document image classification.
- 2) Use the Fuzzy Set Theory for document segmentation.
- 3) Using Advanced ARG match ARGs.
- 4) I think the block classification needn't in the document image classification (i.e. we needn't tell the text, graph, horizontal line, and vertical line.) Advanced ARG algorithm employs the features of each block as parameters to match the ARGs. According to this idea to practice, the performance of whole system is prompted and it reduces the error of recognition.
- 5) In answer to my supervisor, I do my best to survey the methods for block classification. Especially for APACS, the information about it in details is analyzed. I think this method is not very good for the block classification in the document classification.
- 6) All of the almost researched content above uses C++ in UNIX to simulate it.
- 7) I employed more than 80 scanned image file as samples to class them into 12 classes to analysis the result of the testing.
- 8) I completed this thesis on time.

I am very thank for you again for given me so much help and advice. Our system had much improved currently but it is hard to avoid that the system has some deficiency. I would continue to perfect it if I have a chance.

Sincerely,

Xudong Huang

Chapter 1 Introduction

Despite the integration of computerized information handling techniques, recent surveys have shown that well over 90% of business information are still held on paper [35-37]. As the paper volume grows, problems with misfiles, lost files, processing errors and delays, out-of-file conditions, record duplication, etc., are expected to become more serious. In order to improve productivity, reduce costs and increase responsiveness to customer needs, many organizations, in particular, many financial institutions and government agencies, have explored the possibility of handling papers electronically using *Document Image Processing* (DIP) systems [38-40].

Among the many suppliers that market such systems, there are IBM with *Image Plus* [35,41-43], Olivetti with *File Net* [36,38,44-51,60], Philips with *Megadoc* [36-37,41,45,48-49,60], Xionics with *DIP-X* [53,56-58], Wang with *WIIS* [45,53,57,60], CACL with *Catalyst* [53,59], Toshiba with *Tosfile* [48], Racal with *REOS* [45,53,57,60], Ingenium with *Archea* [53], NEC with *Neofile* [48], Kodak with *KIMS* [36,60], Laser Data with *Laser View* [54], ACS with *Hyparchiv* [58], DSL with *INFOplus* [53], Kofax with *Kipp* [52], Sanyo with *Sanfile* [48], Micro Dynamics with *MARS* [60], Unisys with *IIPS* [60], Integrated Documentics with *IMSOFT* [53], Matsushita with *Panafite* [48], Realstream with *The Origin* [53] and ROCC with *ROCCImage* [55]. These systems support almost all the functions needed for a total online document storage and retrieval solution: image capturing, indexing, archiving, viewing, printing, and, for some of them, querying, reporting, document and user tracking, electronic annotation, distribution, security and query management, etc. It should be noted, however, that even with such capabilities, not all office procedures could be fully automated. The responsibility of classifying the optically scanned documents is, for example, still left to the users.

Since there is a need, in many offices, to group documents into classes according to their types (letters, memos, forms, manuals, invoices, technical reports, magazines, journals, etc.), sources and destinations (the receiver, the sender, etc.) and/or their processing requirements (reply to inquirer, requests for repairs, purchase orders, etc.) for the purpose of indexing and work assignment etc., DIP systems that are able to classify document images automatically could be very valuable.

Although reliable optical character recognition (OCR) methods could be used for this task, the process is usually rather slow and expensive. Based on the observation that office documents belonging to the same class have similar layout conventions [69] and that a human is generally able to classify office documents by a perceptive point of view [61], a DIP system is proposed here to perform document classification by automatic layout analysis rather than OCR. This system is capable of acquiring a set of relevant and invariant layout descriptions of each document class from preclassified digitized images so that, based on these descriptions, documents whose class membership is unknown, can be correctly classified.

With the use of a proven inductive inference technique [62-68], the proposed system is able to find the similarities between layout styles of documents belonging to the same class and the differences between that of different classes even in the absence of any a priori knowledge of document structures. It is capable of dealing with both symbolic and numerical data in the inference process and can, for this reason, overcome the problems facing many pattern recognition and artificial intelligence approaches. Furthermore, the proposed system has the advantage that it can be easily implemented on making the training and classification process particularly efficient. Once a document is classified with this system, the recognition of its key components such as the logo, the sender, the destination, etc. In case of business letters or the name, the address, the telephone fields, etc. in case of application forms can be made easier.

1.1 Problem Statement

A document comprises a number of components such as letterhead, logo, text, etc. Office documents belonging to the same class have similar layout conventions. Given a set of documents, they can be classified into different classes according to their types (letters, memos, forms, manuals, invoices, technical reports, magazines, journals, etc.) For example, a letter must contain a letter head, and address, text area, and a signature; a form must contain a number of empty blocks for users to fill in the details; an article must contain columns of text blocks and maybe be with some pictures or diagrams.

Assuming that there are common properties within groups, (e.g. magazine and manuals may have a large proportion of text blocks, letters and memos must have an addressee), and that there are some differences within groups, (e.g. text blocks in

magazine are being displayed in columns while manuals are not being separated into columns; or the position of addressee for letters and memos are different, etc.)

Based on this layout, documents can be classified according to their types (letters, memos, reports, journals, etc.), their sources and destinations (the orders, etc.) for the purpose of indexing and work assignment, etc. Document Image Processing (DIP) system is therefore can be developed to perform document classification by automatic layout analysis.

The major objective of this project is to develop a system that is able to classify a document after layout analysis. Through the development of the project, different approaches on pattern recognition (e.g. Structural Pattern Recognition) will be studied techniques on image processing (such as image retrieval knowledge base, matching algorithms) will be implemented.

1.2 Overview of the results

According to the survey, in general, the whole procession of the document image classification is consisted of three stages, i.e. document segmentation, block classification, and document classification. In this section, the document segmentation, and the block classification are surveyed. The idea of solution for the document classification is presented.

1.2.1 Segmentation

The first step in determining the layout structure of a document page from its binary image is block segmentation. At this stage, the images are partitioned into several, usually rectangular, regions each of that is referred to as a block. The partitioning of an image is based on local and global visual properties of its various regions [3]. Of the many techniques, two of them are particularly suitable for block segmentation involving binary images. The first is the Run-Length Smearing (RLS) method [1], and the second is the Recursive Projection Profile Cuts (RPPC) method [2].

For each row and column in the image, the RLS algorithm merges any two black pixels that are less than a certain threshold apart into a continuous stream of black pixels leaving the white pixels unchanged. This procedure is first applied row-by-row and then column-by-column, yielding two distinct bit maps. The two results are then

combined by applying a logical AND to each pixel location. The resulting RLS image contains a "smear" wherever printed material appears on the original image [16].

Contrary to the RLS algorithm, the RPPC method cuts a document image recursively into rectangular blocks. At each step of the recursive process, the projection profile is computed along both horizontal and vertical directions (a projection along a line parallel to, say the x-axis, is simply a sum of all the pixel values along that line) [3]. A "local" peak detector is then applied to the horizontal and vertical "profiles" to detect local peaks (corresponding to thick black or white gaps) at which the cuts are placed [17].

The effectiveness of the two segmentation methods were compared experimentally and it was concluded that, for getting small blocks such that each block includes just a text line in the text area, the RLS method is better than the RPPC method [3]. This is because the RLS smearing process is done within each text line, and not between text lines. So the blocks which contain just one text line can be obtained directly by using RLS once. On the other hand, the RPPC cuts have to be done several times to make each text line into a block. If large blocks corresponding to, say, paragraphs are needed, then the RPPC method is better than the RLS method. A merging algorithm has to follow the RLS method so as to merge blocks corresponding to single text lines into blocks corresponding to paragraphs.

It should be noted that the blocks, which the RLS method identifies, might not be rectangular. If all blocks need to be rectangular, an algorithm for finding the bounding rectangle has to be applied after executing the RLS procedure. Since, for our application, the image would be better separated into large enough blocks so that a text block can contain a whole paragraph, the RPPC method is used.

Both the RPPC and the RLS methods are sensitive to document skew with respect to the raster scanning direction of the scanner. Skew detection is, therefore, necessary. In the proposed DIP system, this is done by iterative examining small angle projections from normal direction and determining the one that gives the steepest variations of the projection profile [18,19].

The block segmentation decomposes a document image into rectangular blocks each of which includes one of text, horizontal or vertical lines, graphics, or pictures. Several techniques for block segmentation have been developed [12,13,14]. A constrained run-length smoothing algorithm [14] is used to segment a document into

areas of text, lines, and pictures. The graphics, except solid horizontal or vertical lines, is categorized into the same class as pictures. A rule-based block segmentation [12] consists of smearing the document image via the run-length smoothing algorithm, calculating the locations and statistical properties of connected components, and filtering out the image. The Bley algorithm [15] decomposes a connected component into subcomponents which makes more complex in the recognition process and which is sensitive to text font and size variations. A robust algorithm for block segmentation [13] which uses the Hough transformation to group connected components together into logical character strings to be discriminated from the graphics, is relatively independent of changes in text's font, size, and string orientation.

1.2.2 Block Classification

After segmentation, each block is classified into several basic classes: halftone image, text, graphics, vertical and horizontal lines. In order to accomplish such tasks, it is necessary to first extract appropriate features from each block. Simultaneously with component coloring, the following measurements are taken [17]: total number of black pixels in the segmented block, minimum $x - y$ coordinates of a block and its $x - y$ lengths, total number of black pixels in the original block, and number of horizontal white-black transitions in the original image block, etc. Given these measurements, several features are computed [1, 17, 20, 21]. These features include: the height of a block (H), its eccentricity (E), the ratio of the number of black pixels to the area of the surrounding rectangle (S), and the mean horizontal length of the black runs in the original data from the block (R). A block is considered to be text if it is a textured stripe of mean height H_m and mean black-run length R_m . The distribution of values in the $R - H$ plane derived from sample documents are observed to determine the discriminative function. Low R and H values represent regions containing text. To determine the threshold values of R and H that define the text region in the $R - H$ plane, an adaptive method is used. This method estimates H_m , R_m and the standard deviation of R and H . These values are then used to classify the various blocks by using the following pattern classification scheme that assumes linear separability:

- Text if $R < C_{21}R_m$ and $H < C_{22}H_m$;
- Horizontal solid black lines if $R > C_{21}R_m$ and $H < C_{22}H_m$;

- Graphic and Halftone images if $E > 1/C_{23}$ and $H > C_{22} H_m$; and
- Vertical solid black lines if $E < 1/C_{23}$ and $H > C_{22} H_m$.

The constants C_{ij} are determined by heuristic rules from examining the $R - H$ plane plot of typical document and the values of R_m and H_m .

In the preceding section, the mixed-mode (mixture of text, graphics and pictures) document is segmented into blocks, each of which contains the single-mode content. This section presents the block classification algorithm to classify the blocks into one of the text, horizontal or vertical line, graphics and picture classes. Most existing block classification techniques are based on the discrimination of statistical local or global features. The commonly used features, such as block height and aspect ratio, are elementary for extracting the mostly popular text blocks. However, they are not sufficient in classifying the mixed -mode document blocks. Therefore, more complex features are needed to achieve the higher reliability.

Since the text/image blocks tend to cluster in space with respect to some features, a threshold or a discriminative function is selected for separation. A two-dimensional plane consisting of mean value of the block height versus run length of the block mean black pixel is established to classify document blocks into text, image, horizontal line, and vertical line [1]. A rule-based classification uses the features such as height, aspect ratio, density, perimeter, and perimeter/width ratio [12]. A newspaper classification method creates the black-white pair run-length matrix to derive three features: short run emphasis, long run emphasis, and extra long run emphasis for clustering [3]. The distribution of the features used is dependent on the character's font and size and the image resolution. The inappropriately chosen threshold can lead into misclassification.

Text Analysis Recognition

(1) Text Analysis Recognition

There are two main types of analysis that are applied to text in documents. One is optical character recognition (OCR) to derive the meaning of the characters and words from their bit-mapped images. The other is page layout analysis to discover formatting of the text and, from that, to derive meaning associated with the positional and functional blocks in which the text is located. These operations may be performed separately, or the results from one analysis may be used to aid or correct those from

the other. OCR methods are usually distinguished as being applicable to either machine-printed or handwritten character recognition. Page layout analysis techniques are applied to machine-printed or handwritten text occurring within delineated blocks on a printed form.

(2) Skew estimation

A *text line* is a group of characters, symbols, and words that are adjacent, relatively close to each other, and through which a straight line can be drawn (usually with horizontal or vertical orientation). The dominant orientation of the text lines in a document page determines the *skew angle* of that page. Therefore, analysis steps such as OCR and page layout analysis most often depend on an input page with zero *skew*, it is important to perform skew estimation and correction before these steps. Also, since a reader expects a page displayed on a computer screen to be upright in normal reading orientation, skew correction is normally done before scanned pages are displayed. There are three categories of skew estimation techniques based on their approaches:

- Projection profile methods:

Projection profile methods are popular for skew detection. A *projection profile* is a histogram of the number of ON-pixel values accumulated along parallel sample lines taken through the document. The profile may be at any angle, but often it is taken horizontally along rows or vertically along columns.

The most straightforward use of the projection profile for skew detection is to compute the skew at a number of angles close to the expected orientation [22]. For each angle, a measure is made of the variation in the bin height along the profile, and the one with the maximum variation gives the skew angle. One modification to the projection profile method was proposed by Baird [19] to improve the speed and accuracy of skew detection. In a faster, though less accurate, method of approximating the skew angle, shifts in projection profiles are measured. [24]. For text lines that are approximately horizontal, the document is divided into equal-width vertical strips that each covers the height of the page image. Another fast method based on the measurement of shifts between strips uses vertical projection profiles determined for horizontal strips of the page [25], in contrast to the use of horizontal profiles on vertical strips by [24], as described above.

- Hough transform methods:

The Hough transform was useful for straight-line detection. It can be use for a similar purpose to find skew from text line components [26,27,28]. The Hough transform maps each point in the original (x,y) plane to all points in the (r, θ) Hough plane of possible lines through (x, y) with slope θ and distance form origin r .

- Nearest-neighbor methods:

All the above methods have some limitation in the maximum amount of skew that they can handle. Another approach - using nearest-neighbor clustering - does not have this limitation [Hahsizume 1986]. An extension of this nearest-neighbor approach is to obtain not just one neighbour for each component but k neighbor, where k typically is 4 or 5 [33].

(3) Page layout analysis

After skew detection, the image is usually rotated to zero skew angle, and then layout analysis is performed. *Structural layout analysis* is performed to obtain a physical segmentation into groups of document components. Structural layout analysis can be performed in top-down or bottom-up fashion.

- Top-down analysis:

The run-length smoothing algorithm is a popular method of projection profiles for this performing this smoothing [Wong 1982]. This method merges characters into words, words into text lines, and text lines into paragraphs. This use of horizontal and vertical projection profiles requires that the image be first skew corrected and that spacing is known and uniform within the image. A more structured top-down method that also uses projection profiles splits the document into successively smaller rectangular blocks by alternately making horizontal and vertical “cuts” along white space, starting with a full page and continuing with each subblock [29,30]. The results of segmentation are represented on an X-Y tree, where the top-level node is for the page, each lower node is for a block, and each level alternately represents the results of horizontal (X-cut) and vertical (Y-cut) segmentation. Another top-down layout technique analyzes white space to isolate blocks and then uses projection profiles to find lines [32]. For most page formats, this a very effective approach. However, for pages where text does not have linear bounds and where figures are intermixed both in and around text, these methods may be inappropriate.

- Bottom-up analysis:

One approach combines a number of the techniques described above [12]. First, the skew is found from the Hough transform, then, between-line spacing is found as the peak of the one-dimensional Fourier transform of the projection profile θ for fixed at the computed skew angle. Run-length smoothing is performed, and then within-line spacing is determined by finding the peak on a histogram of these within-line lengths of white spaces and of black lengths. Next, bottom-up merging of the text components is done by a sequence of run-length smoothing operation. The docstrum method [33] employs bottom-up k -nearest-neighbors clustering to group form characters into text lines and structural blocks of layout analysis by the docstrum method. Similar to the top-down method that uses functional labeling information in the course of performing structural layout analysis [31], a combination of the two processes can be used to advantage in bottom-up analysis. In [24], segmentation is performed, using field separators and then blank delimiters, as for many other methods.

- Functional labeling:

As described above, some of the layout analysis methods [31,24] perform functional labeling in the course of structural blocking. Other methods perform these steps sequentially, first obtaining structural blocks and then applying functional labels.

Graphics Analysis and Recognition

Graphics recognition and interpretation are important topics in document image analysis since graphics elements pervade textual material, with diagrams illustrating concepts in the text, company logos heading business letters, and lines separating fields in tables and sections of text. The graphics components that we deal with are the binary-valued entities that occur along with text and pictures in documents. We will omit this section because our topics involve in this content and it is not main content.

1.2.3 Document Classification

For the document classification, by survey, we haven't found methodologies about it. Many researchers only prompt document-processing system, but it is just relation to the segmentation, block classification, and it has no something with the document classification. In this thesis, I will present a method and its advanced method to achieve the two document matching. The two document matching is transferred the

two attribute random graph matching.

1.3 Fuzzy Methodology

The main methodologies for the DIP System will be include three parts that is segmentation, classification and pattern. In the three parts, we may be mainly used Fuzzy Methodology or other methodology to analysis the encountered question. In the here, we depict some general methodologies or concepts of the fuzzy using in the pattern recognition.

Fuzzy Set The function $\mu: X \rightarrow [0,1]$ is given the label A , and is A called a fuzzy (sub) set of X . μ is called the membership function of A .

Since a fuzzy set is always defined as a subset of a general set X , the “sub” is frequently abbreviated, and it is just called a fuzzy set. From the definition we see that the function over the interval $[0, 1]$ has a one-to-one correspondence with the fuzzy set.

α -Cut for a fuzzy set A ,

$$A_\alpha^\Delta = \{x \mid \mu_A(x) > \alpha\}; \alpha \in [0, 1)$$

$$A_\alpha = \{x \mid \mu_A(x) \geq \alpha\}; \alpha \in (0, 1]$$

are called the weak α -cut and strong α -cut respectively.

The term α -cut is a general term that includes both strong and weak types. The weak α -cut is also called the α level-set. The difference between strong and weak is the presence or absence of the equal sign. If the membership function is continuous, the distinction between strong and weak is not necessary due to the logical development inherent in the α -cut.

Resolution Principle

$$\mu_A(x) = \sup_{\alpha \in (0,1)} [\alpha \wedge X_{A_\alpha}(x)] = \sup_{\alpha \in (0,1)} [\alpha \wedge X_{A_\alpha}^\Delta(x)]$$

Using the resolution principle, if we define the fuzzy set αA_α here as

$$\alpha A_\alpha \leftrightarrow \mu_{\alpha A_\alpha}(x) = \alpha \wedge X_{A_\alpha}(x)$$

the resolution principle is expressed in the form

$$A = \bigcup_{\alpha \in (0,1]} \alpha A_{\alpha}$$

In other words, a fuzzy set A is decomposed into αA_{α} , $\alpha \in (0, 1]$ and is expressed as the union of these. This is what the resolution principle means. If $\alpha_1 < \alpha_2$, $\alpha_1 A \supset \alpha_2 A$. Given fuzzy sets such as $\alpha_1 A$ and $\alpha_2 A$, we can retrieve the original membership function of fuzzy set A by connecting the corners of their membership functions. This gives rise to the function

$$\mu_A(x) \leftrightarrow A_{\alpha}, \alpha \in (0, 1].$$

Fuzzy Relations Fuzzy relation R from set X to set Y (or between X and Y) is a fuzzy set in the direct product $X \times Y = \{ (x, y) \mid x \in X, y \in Y \}$, and is characterized by a membership function μ_R :

$$\mu_R: X \times Y \rightarrow [0, 1]$$

Especially when $X = Y$, R is known as a fuzzy relation to X .

Fuzzy Matrices and Fuzzy Graphs Given finite set $X = \{ x_1, x_2, \dots, x_m \}$, $Y = \{ y_1, y_2, \dots, y_n \}$, a fuzzy relation in $X \times Y$ can be expressed by an $m \times n$ matrix as follows:

$$R = \begin{bmatrix} \mu_R(x_1, y_1) & \mu_R(x_1, y_2) & \cdots & \mu_R(x_1, y_n) \\ \mu_R(x_2, y_1) & \mu_R(x_2, y_2) & \cdots & \mu_R(x_2, y_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mu_R(x_m, y_1) & \mu_R(x_m, y_2) & \cdots & \mu_R(x_m, y_n) \end{bmatrix}$$

This kind of matrix, which expresses a fuzzy relation, is called a *fuzzy matrix*. Since μ_R has values within the interval $[0, 1]$, the elements of the fuzzy matrix also have values within $[0, 1]$. In order to express fuzzy relation R in a graph, for $\mu_R(x_i, y_j)$, we make x_i, y_j vertices and add the grade $\mu_R(x_i, y_j)$ to the arc from x_i to y_j . This graph is called a *fuzzy graph*.

Direct Fuzzy Pattern Recognition Given the set U for all objects to be recognized. Each object u in the set U has p features index. i.e. u_1, u_2, \dots, u_p . Each feature index depicts the special feature of the object u . So $u = (u_1, u_2, \dots, u_p)$. Called feature vector.

Let the set U to be recognized divided into n classifications, and each classification is fuzzy set in the set U , signed as A_1, A_2, \dots, A_n . Each object u has a

group of membership degree: $\mu_{A_1}(u), \mu_{A_2}(u), \dots, \mu_{A_n}(u)$. They express the degree that the object u be subordinate to the A_1, A_2, \dots, A_n respectively.

• **Maximum Membership Degree Rule** Let the set A is the fuzzy set at the given interval U , the u_1, u_2, \dots, u_n are objects in the set U . If $\mu_{\underline{A}}(u_0) = \max(\mu_{\underline{A}}(u_1), \mu_{\underline{A}}(u_2), \dots, \mu_{\underline{A}}(u_n))$, then the membership of u_i precedes over the fuzzy subset \underline{A} .

• **Maximum Membership Rule** Let the set A_1, A_2, \dots, A_n is the fuzzy subset at the given interval U , the $u_0 \in U$ is recognized objects. If $\mu_{\underline{A}_i}(u_0) = \max(\mu_{\underline{A}_1}(u_0), \mu_{\underline{A}_2}(u_0), \dots, \mu_{\underline{A}_n}(u_0))$, then the membership of u_0 precedes over the fuzzy subset \underline{A}_i .

• **Cut Level Rule** Let the set A_1, A_2, \dots, A_n is the fuzzy subset at the given interval U , given a cut level $\lambda \in [0, 1]$, and the $u_0 \in U$ is recognized objects.

(1) If $\max(\mu_{\underline{A}_1}(u_0), \mu_{\underline{A}_2}(u_0), \dots, \mu_{\underline{A}_n}(u_0)) < \lambda$, then refuse to recognize, and find the cause.

(2) If $\max(\mu_{\underline{A}_1}(u_0), \mu_{\underline{A}_2}(u_0), \dots, \mu_{\underline{A}_n}(u_0)) \geq \lambda$, and $\mu_{\underline{A}_1}(u_0), \mu_{\underline{A}_2}(u_0), \dots, \mu_{\underline{A}_n}(u_0)$ great or equal to λ , then the result is feasible, and let u_0 belong to $\underline{A}_{i1} \cap \underline{A}_{i2} \cap \dots \cap \underline{A}_{ik}$.

In the practice, the two expressions above can modify as:

$$\sum_{i=1}^n \alpha_i \mu_{\underline{A}_i}(u_0) < \lambda$$

$$\sum_{i=1}^n \alpha_i \mu_{\underline{A}_i}(u_0) \geq \lambda$$

where $\sum_{i=1}^n \alpha_i = 1$. α_i is the weight of \underline{A}_i

Indirect Fuzzy Pattern Recognition Given the set U for all objects to be recognized. Each object B is a fuzzy subset in the set U , and each element in the U

has p features index (u_1, u_2, \dots, u_p) . Given fuzzy subset $\underline{A}_1, \underline{A}_2, \dots, \underline{A}_n$ in the set U . We need to determine the conjoint degree $\delta(\underline{B}, \underline{A}_i)$ when we judge the object \underline{B} is belonged to which fuzzy subset \underline{A}_i ($i = 1, 2, \dots, n$).

Select Near Rule: Let $\underline{A}_1, \underline{A}_2, \dots, \underline{A}_n$ is the fuzzy subset in the set U , and \underline{B} is also the fuzzy subset in the set U . If $\delta(\underline{B}, \underline{A}_i) = \max(\delta(\underline{B}, \underline{A}_1), \dots, \delta(\underline{B}, \underline{A}_n))$, then \underline{B} is belonged to the fuzzy subset \underline{A}_i .

Chapter 2 Technique for Pattern Recognition

The process of recognizing a pattern is the classification of a sample pattern into one of more predefined categories. There are mainly two different approaches for pattern recognition: (1) statistical pattern recognition and (2) Structural pattern recognition. The proposed DIP system adopted the ideas for structural pattern recognition. These approaches as well as the pattern representation in structural pattern recognition will be discussed in this section.

2.1 Statistical Pattern Recognition

It is a very sound theoretical method for classification of patterns. It is well suited in applications where a limited number of features are used. The purpose of statistical pattern recognition is to determine to which category or class a given sample belongs. Through observation and measurement processes, we obtain a set of numbers that make up the measurement vector. The vector is a random vector and its density function depends on its class. The design of a classifier consists of two parts. One is to collect data samples from various classes and to find the boundaries that separate the classes. This process is called classifier design, training, or learning. The other is to test the designed classifier by feeding the samples whose class identities are known.

2.2 Structural Pattern Recognition

This based on symbolic data structures like strings, trees, graphs, or arrays for pattern representation instead of using vectors of numbers in the statistical approach. These data structures allow the description of relations between elementary pattern components and provide means for hierarchical models showing how complex patterns are built from simpler parts.

We can recognize an unknown pattern by comparing its symbolic representation with a number of predefined object models. A symbolic match computes a measure of similarity between the unknown input and a number of prototype models.

2.3 Pattern Representation

In the structural approach, we use symbolic data structures for the representation of the patterns under study. There are usually two sets of patterns. First, model or prototype patterns. We used this sample set of patterns for system design. Secondly, we have unknown patterns to be recognized in the actual application phase of a

pattern recognition system. Typically, we used same kind of data structures for both the unknown patterns and the samples.

Words of symbols, or strings, are the most fundamental data structures for pattern representation. The individual symbols in a string usually represent atomic pattern components. Strings are a one-dimensional formalism but many patterns are inherently two or more dimensional. Therefore, it comes up with more general data structures for pattern representation. The most powerful class of symbolic structures for more-dimensional representation is graphs.

2.4 Graph Matching and Graph Isomorphism

Graph A graph consists of a set of nodes and a set of edges. Given a pattern in terms of a graph, the nodes usually represent simpler subpatterns and the edges indicate relations between those subpatterns. The relations may be spatial, temporal, or any other type. A graph representation is very useful of deriving a symbolic scene interpretation.

An important subclass of graphs is trees. A tree has three different classes of nodes, namely root, interior, and leave. There is exactly one root, which has only outgoing and no incoming edges. Each interior node has exactly one incoming and at east one outgoing edge. Trees are interesting for pattern recognition applications as they are representatively less expensive than graphs.

An array is a special type of graph where the nodes and the edges are arranged in a regular form. This type of data structure is particularly useful for low level pattern representation.

In structural pattern recognition, a pattern can be represented by a set of primitives and the relations among them. In this system, attributed graphs are being used for pattern representation. For attributed graphs, a vertex with attributed values is used to represent a pattern primitive and an attributed are is used to represent the relation between them. The basic concepts and definitions given below are adapted from with modifications to comply with the commonly accepted notations.

Attributed Graphs The nodes in attributed graph denote pattern primitives, and the branches between two nodes represent the relations between primitives.

Primitives: Let $Z = \{z_i \mid i = 1, 2, \dots, I\}$ be a nonempty and finite set of possible attributes for describing pattern primitives (vertex attributes, e.g. TextBlock1, GraphicBlock2, Line3 ... etc.) and, for each i , $S_i = \{s_{ij} \mid j = 1, 2, \dots, J\}$ be the set of

possible attribute values associated with z_i (e.g. no of black pixels in the block, size of the block ... etc.) Let $L_v = \{(z_i, s_{ij}) \mid i = 1, 2, \dots, I; j = 1, 2, \dots, J_j\}$ be the set of legal attribute value pairs. Let Π denote the set of all possible pattern primitives.

Relations: Similarly, let $F = \{f_i \mid i = 1, 2, \dots, I'\}$ be a nonempty finite set of possible relational attributes (edge attributes, e.g. On top of, Left of ... etc.). Let $L_a = \{(f_i, t_{ij}) \mid i = 1, 2, \dots, I'; j = 1, 2, \dots, T'_{ij}\}$ be the set of legal attribute value pairs. Let θ denote the set of all relations.

An *attributed graph* G over $L = (L_v, L_a)$, with an underlying graph structure $H = (N, E)$, is defined to be a pair (V, A) where $V = (N, \mu)$ is called an *attributed vertex set* and $A = (E, \delta)$ is called *vertex interpreter* and *arc interpreter*, respectively.

Graph Isomorphism Two attributed graphs $G_1 = (V_1, A_1)$ and $G_2 = (V_2, A_2)$ are said to be structurally isomorphic if there exists an isomorphism $T : H_1 \rightarrow H_2$ where $H_1 = (N_1, E_1)$ and $H_2 = (N_2, E_2)$ represent the structural aspects of G_1 and G_2 respectively. G_1 and G_2 are said to be completely isomorphic if there exists an attribute value preserving structural isomorphism T between G_1 and G_2 .

2.5 Attributed Random Graph

For the document classification, we can define the attributed random graph according the above definition as follows:

An *attributed random graph* (ARG) over L is a tuple $\Omega = (N, A, \xi, \zeta)$, where N is a non-empty finite set of nodes, $A \subset N \times N$ is a set of distinct ordered pairs of distinct elements in N called arcs; $\xi: N \rightarrow V_N$ is an node interpretation function; and $\zeta: A \rightarrow V_A$ is an arc interpretation function.

Nodes of an ARG: The syntactic of a node has a value chosen from the set of nodes v_i ($1 \leq i \leq n$). The semantic is ordinal used to denote each vector.

Relations of nodes in an ARG: The syntactic of a relation r_{ij} between node i and node j is represented as a vector. The semantic is not used.

Chapter 3 Fuzzy Segmentation for Document Image Analysis

This chapter describes a segmentation method called the Fuzzy Segmentation Method (FSM) for the analysis of document images. Based on the use of concepts in fuzzy set theory, we define fuzzy blank and fuzzy black lines and based on these definitions, we, in turn, define fuzzy blank and fuzzy black blocks. An efficient algorithm is proposed here for their identification. The ideas of λ -cut sets are utilized when determining the vertical and horizontal thresholds in the segmentation process. With the definition of fuzzy sets, such a process can be made automated in an adaptive process. When comparing to existing approaches, it is better able to correctly identify blocks in a document image.

3.1 Introduction to the segmentation of document classification

Even with the advance of computers, paper documents are still one of the most common medium for information transmission in today's society. Document image analysis, as a result, remains an important area of research in computing. Be it optical character recognition or document image classification, the images have to be first segmented and then classified. If segmentation cannot be performed satisfactorily, pattern recognition cannot be performed accurately.

For document image segmentation, two algorithms can be adopted: the Run Length Smoothing Algorithm (RLSA) [1], and the Recursive X-Y Cuts algorithm (RXYC) [2]. The RLSA is based on the 'distance' between two black pixels in which if it's less than a certain threshold, all the pixels will be merged to become a continuous stream of dark pixels. The procedures are applied row by row and then column by column. The two results are then combined by applying a logical OR to each pixel location [1, 3]. Briefly, if an image is represented as a binary sequence in which black pixels are represented by 0's and white by 1's, the RLSA can be used to transform a binary input sequence x into a binary output sequence y according to the following rules:

1. 1's in x are changed to 0's in y if the number of adjacent 1's is less than or equal to a predefined threshold c .
2. 0's in x are unchanged in y .

For example, with $c = 4$ the sequence x is mapped into y as follows:

x: 1111101101111111110101111011111111100

y: 111110000111111111000000011111111100

The RLSA is first applied row-by-row and then column-by-column, yielding two distinct bitmaps. The two results are then combined by applying a logical OR to each pixel location. Since spacing of document components tends to differ horizontally and vertically, the threshold C_h and C_v in the horizontal and vertical directions respectively need not be the same. Additional horizontal smoothing using the RLSA produces the final segmentation result.

Unlike the RLSA, the RXYC cuts image recursively into blocks. At each step of the recursive process, the projection profile is computed along both horizontal and vertical directions. A projection profile is then obtained by determining the number of black pixels that fall onto a projection axis (a projection along a line parallel to, say the x-axis, is simply a sum of all the pixel values along that line). A "local" peak detector is then applied to the horizontal and vertical "profiles" to detect local peaks (corresponding to thick black or white gaps) at which the cuts are placed [1].



Fig.3-1 The original image before segmentation

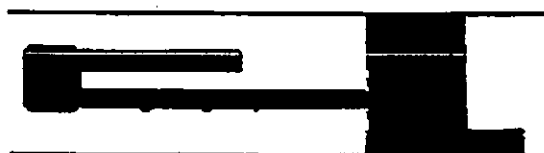


Fig.3-2 Segmentation with "too large" a threshold

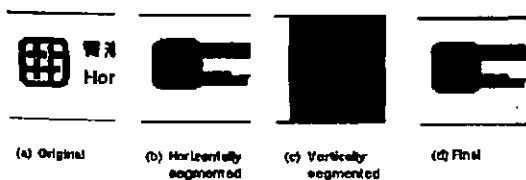
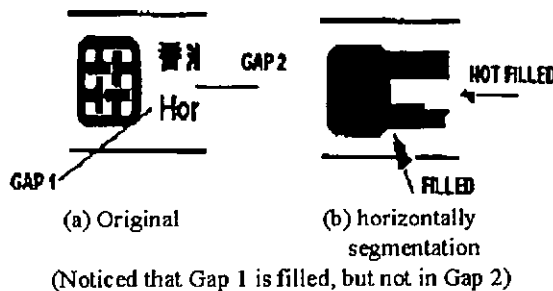


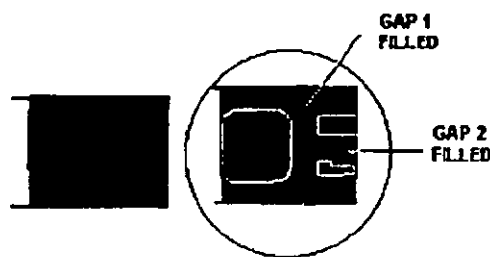
Fig.3-3 Doing horizontal and vertical segmentation independently



(a) Original

(b) horizontally segmented

(Noticed that Gap 1 is filled, but not in Gap 2)



(c)Vertically Segmentation

(Noticed that Gap 1 and Gap 2 are both filled).

Fig.3-4 After performing a logical OR

Projection profiles represent a global feature of a document. They play an important role in the document element extraction, character segmentation and skew normalization. All objects in a document are contained in rectangular blocks. Blanks

are placed between these rectangles. Thus, the document projection profile is a waveform whose deep valleys correspond to the blank areas of the documents. A deep valley with a width greater than an established threshold C , can be cut as the position corresponding to the edge of an object or block. Because a document generally consists of several blocks, the process of projection should be done recursively until all of the blocks have been located.

By intuition, in this phase, the choice of the smoothing threshold C is very important. A horizontal threshold that is too small simply merges within individual characters. Slightly larger threshold merge individual characters into a word but are not large enough to bridge the space between the two words. A threshold that is too large often causes sentences to join to non-text regions, or to connect to adjacent columns.

The suggested threshold C_h and C_v for the horizontal and vertical directions are 300 and 500 respectively. However, since we are dealing with office documents such as letters, memos, forms, etc., in which different blocks may lie closely to each other, a new set of thresholds are required. Consider, for example, the image of a letter head in Fig.3-1 in which a logo is found on the left, a company name in the middle, and an address on the right and they are all bounded by two horizontal lines.

However, if we are to use the proposed threshold, the logo will merge with the company name, and the company name will merged with part of the address after horizontal segmentation (see Fig.3-2). Since the blocks are located so close to each other, this same problem will occur. After vertical segmentation, the 2 horizontal lines will merge with the address block. And as a result, all desired information is lost, resulting only in a single "large" block (see Fig.3-2).

In order to preserve all the useful information, smaller thresholds of $C_h = 10$ and $C_v = 4$ can be used. However, there is still a problem if we perform a logical OR to the 2 independently segmented bitmaps. As shown in Fig.3-3, the logo merges with the company name. The original idea of having a logical OR (AND if black=1 and white=0) is to preserve the line spacing between every single text blocks. However, an undesirable side effect, as illustrated in Fig.3-4b and 3-4c, results. The space between the logo and the text block (gap 1) is filled both in the horizontal and vertical segmentation, making the 2 blocks merge with each other after performing a logical OR. This is unexpected because of the 2 horizontal lines that bounded the logo and

text blocks and these lines provide misleading results after vertical segmentation.

To overcome these problems, we present here a fuzzy segmentation method (FSM) for document image analysis. The FSM allows the threshold to be determined automatically and adaptively.

3.2 The Fuzzy Segmentation Method (FSM)

Let $F(i, j)$ be an $N \times M$ (where N, M are non-negative integers) matrix representing a document image, and let

$$F(i, j) = \begin{cases} 0 & \text{white pixel } 0 \leq i < N \text{ (Width of Horizontal Pixel)} \\ 1 & \text{Black pixel } 0 \leq j < M \text{ (Height of Vertical Pixel)} \end{cases}$$

Given $F(i, j)$, we can define a $S(j)$ to be $S(j) = \sum_{i=0}^{N-1} F(i, j)$. Fig.3-6 shows a plot of

$S(j)$ against j for the image in Fig.3-5. Since $S(j)$ can take on a value in the interval from 0 to M , its universe of discourse, U is defined to be $U = [0, M]$. The crisp value of $S(j)$ can then be mapped into a fuzzy set defined on U and we define the membership function of this fuzzy set to be:

$$\mu(j) = 1 - \frac{S(j)}{M} \quad (0 \leq j < M) \quad (1)$$

According to (1), if $S(j) \rightarrow M$, then $\mu(j) \rightarrow 0$. This happens when the j th row is or is almost a black line. If there are at least some black pixels in the j th row, the degree of membership, $\mu(j)$, indicates, roughly, the amount of black pixels in it (Figs.3-5 and 3-6).

Given the fuzzy set $S(j)$, we can define a lambda-cut set, S_λ , $\lambda \in [0, 1]$, of it so that

$$S_\lambda = \{S(j) \mid \mu(j) > \lambda\} \text{ and}$$

$$\bar{S}_\lambda = \{S(j) \mid \mu(j) \leq \lambda\} \text{ where } 0 \leq j < M$$

and members of S_λ can be considered a fuzzy blank line whereas that of \bar{S}_λ can be considered a fuzzy black one. Based on the definitions of S_λ , \bar{S}_λ , we can construct a set of all fuzzy blank lines and a set of all fuzzy black lines to contain those lines that has these characteristics respectively. These two sets can, therefore, be defined as follows:

$$J_\lambda = \{j \mid S(j) \in \bar{S}_\lambda\} \quad 0 \leq j < M$$

$$\bar{J}_\lambda = \{j \mid S(j) \in S_\lambda\} \quad 0 \leq j < M$$

The set J_λ is the set of fuzzy black lines and the set \bar{J}_λ is the set of fuzzy blank lines.

Let $j_k, 0 \leq j < M$ be the element of J_λ , then $J_\lambda = \{j_0, \dots, j_k, \dots, j_l\}, 0 \leq k, l < M$. Based on this definition, we can construct subsets of J_λ so that each such subset contains a continuous block of fuzzy black lines. To form a block, two lines have to be adjacent to each other. For this reason, a block of black lines can be defined by the following subset of J_λ as follows:

$$\{j_i | j_{i+1} - j_i = 1, 0 \leq i < M\} \subseteq J_\lambda \quad (2)$$

where $0 \leq j_l < M, 0 \leq l < M$. If we choose different λ , i.e. $0 \leq \lambda_0 < \lambda_1 < \dots < \lambda_n \leq 1, 0 \leq n < M$, the definition of fuzzy black lines can be altered and based on the different definitions, we have different fuzzy black blocks:

$$\{j_i | j_{i+1} - j_i = 1, 0 \leq j < M\}_k \subseteq J_{\lambda_k} \quad \text{and}$$

$$\bigcup_{k=0}^n \{j_i | j_{i+1} - j_i = 1, 0 \leq i < M\}_k \subseteq \bigcup_{k=0}^n J_{\lambda_k}$$

where $0 \leq k \leq n, 0 \leq n < M, 0 \leq j_l < M, 0 \leq l < M$.

By defining $A = \bigcup_{k=0}^n \{j_i | j_{i+1} - j_i = 1, 0 \leq i < M\}_k$, we can eliminate all intersecting subsets and construct the following function $C(j)$ as follows:

$$C(j) = \begin{cases} 1, j \in \{j_i | j_{i+1} - j_i = 1, 0 \leq i < M\}_l \quad \text{and} \\ \quad \quad \quad \bigcup_{k=0}^{l-1} \{j_i | j_{i+1} - j_i = 1, 0 \leq i < M\}_k \\ 0, \text{ otherwise} \end{cases}$$

Let N_i be used to denote the number of the elements of the i th subset of A that satisfies $C(j) = 1$, then

$$N_i = |\{j_i | j_{i+1} - j_i = 1, 0 \leq i < M\}|$$

In other words, $N_i, (0 \leq i < M)$, can be considered the width of a black block or a threshold C (C_h or C_v) and we can define the set T of contain all the black box in the image

$$T = \{N_0, N_1, \dots, N_i\} \quad (0 \leq i < M) \quad (3)$$

Depending on the particular applications, the elements in T can be chosen

accordingly. If a local threshold is preferred, T can be used directly.

3.3 Automatic Determination of Thresholds

To use the FSM, we have to determine the value of λ . For S_λ , $\lambda \in [0,1]$, if the value of λ is in the interval from 0 to 0.5, the possibility for this row to be black is greater but if the value of λ is in the 0.5 to 1 interval, then the possibility of this row to be blank is greater. Therefore, λ can be determined according to this definition of fuzzy blank and black lines. From (1), the following equations can be used to determine λ :

$$\alpha = \max_{j=0}^k \left[1 - \frac{S(j)}{M} \right] \quad (0 \leq j \leq k < M)$$

$$\lambda_i = \frac{i}{\beta} \alpha \quad (i=0, \dots, \beta; 0 \leq i \leq \beta < M)$$

In general, if the value of λ is from 0.5 to 1, then using 5 grades for λ (i.e. $\beta = 4$) for both C_h and C_v should be good enough. The value of λ can be given according to the following rules:

1. the valid range of the image, i.e., the valid value of the width or height in pixels. If the width or height is more bigger, the β is more bigger.
2. the performance of machine. The value of β can be given bigger if the machine runs fast.
3. for the office document, in general we usually use A4 papers, therefore, using 5 grades is enough.

In a word, to check the effect, we had given the different values to β . If the β is too smaller, e.g. β is been given value 1 or 2, it is insignificance because the interval is too bigger. If the β is too bigger, e.g. β is been given value that is close to the width for the vertical direction or the height for the horizontal direction, it is affected the speed of process because the interval is too smaller. By intuitively, it is insignificance if β is been given value that is bigger than the width for the vertical direction or the height for the horizontal direction.

For document image segmentation, FSM should be applied in two directions, the horizontal and vertical directions. The horizontal and vertical thresholds, C_h and C_v , should be different at different locations in the image. Therefore, the document structure should be considered first according to (1) and (3). If $\mu(j) = 1$ and the

thresholds of C_h (horizontal threshold for blank blocks) or C_v (vertical threshold for blank blocks) is the largest amongst all, then the document can be separated from here.

Given an image, therefore, FSM is first applied in one direction (vertical or horizontal) to the whole document to obtain a set of thresholds T_v . For each segmented unit corresponding to the elements of T_v , we then apply FSM to find T_h . The "algorithm 2" and "algorithm 3" below describes how we perform segmentation in each direction separately.

[Algorithm-1] // *Calculation of the vertical threshold*

- 1) Calculate the sum of pixels for each row.
- 2) Construct the membership function, and the maximum sum of row for calculation of the lambda for the vertical direction.
- 3) Calculate the lambda for the vertical direction.
- 4) Construct the checking function, $C(j)$.
- 5) Calculate the values of the vertical threshold for the set T_v .

To perform vertical segmentation based on the above, we use the following algorithm:

[Algorithm-2] // *Vertical Segmentation*

Initial states: $nbytes = (\text{Width of Horizontal Pixel} + 7) / 8$;

$nbytes = (nbytes + 3) / 4$; $nbytes *= 4$; $x = 0$;

1) // *loop row by row, bit after bit*

$x < nbytes$;

2) // *check bit by bit, with bit mask started with 0x80 to check first bit*

while Bit wise Loop < 8 , then

3) // *loop row by row*

$y = 0$;

4) $y < \text{Height of Vertical Pixel}$, then copy all Bit wise Loopth bit of x Byte in row y to scanline[y]; reset scanline[y] to 0 in case if distance exceeds local threshold in the set T_v ;

5) $y++$; then go to 4);

6) // *loop row by row again*

$y = 0$;

7) $y < \text{Height of Vertical Pixel}$, then perform logical AND with Mask and Current

```

Byte;
// the Mask is created according to start and end 'row'
copy all Bit wise Loopth to scanline;
8) y++; then go to 7);
9) // mask shift one position
bit mask >>= 1;
10) Bit wise Loop++; then go to 2);
11) x++; then go to 1).

```

The horizontal segmentation performs another bit wise operation. the algorithm below gives the details as follows:

[Algorithm-3] // *Horizontal Segmentation*

Initial states: nbytes = (Width of Horizontal Pixel + 7) / 8;

nbytes = (nbytes + 3) / 4; y = 0;

1) // loop byte by byte, row after row

y < Height of Vertical Pixel

2) x = 0;

3) x < nbytes;

4) // check bit by bit, with bit mask started with 0x80 to check first bit

while Bit wise Loop < 8, then Output Bit = (Current Byte & bit mask) ? 1:0;

5) If Output Bit == 0 then

If Distance > Local threshold in the set T_h , then go to bit where the 1st 'black' occurred and perform logical AND with Mask;

// Mask created according to the position of first & last occurrence of black pixel.

else Distance++;

end if

end if

6) // mask shift one position

bit mask >>= 1;

7) Bit wise Loop++; then go to 4);

8) x++; then go to 3);

9) y++; then go to 1).

In addition, in order to improve the effectiveness of FSM, we can use some noise

elimination techniques before applying FSM. One can use a simple 3×3 window to reduce the noise [4] or one can also use the 1×3 window to expand the pixel or "black box" method to weight the sum of the pixel (Fig.3-7 and Fig.3-8) so that if the sum of pixels for each row or column is large, then the value of lambda can be easily determined. This algorithm is given below:

[Algorithm-4] // *Black Box method*

- 1) For each pixel in the document,
 - if the pixel is black pixel, then
 - 2) Checking each pixel is whether or not connect with it in around it
 - 3) If connect with it, then mark it with flag and go to 2)
 - 4) If all pixel in all for checking pixel don't connect, then calculate the coordinate of the left top and right bottom. Go to 1)
- end
- 6) until the end of document

[Algorithm-5] // *the process*

- 1) Copy a scanned image file
- 2) To the copied file,
 - (1) Using the 3×3 window to reduce the noise
 - (2) Expand 2 pixels in the copied file for each pixel in source scanned image file to left or right when segment the rows, or to the up or down when segment the column
- 3) Using FMS for rows or columns
- 4) Paragraph block
- 5) The result react to source scanned image file, and counters the each parameter of each paragraph block.
- 6) To use it in future

It is depicted very simple because this method don't include much technique. It only uses the source scanned image file as a reference file, the copied file as operating file. However, the result is very good because it can weight the sum of the pixel and can conveniently operate for programming in C++.

3.4 Features of the FSM

The segmentation technique proposed above has these unique features. The use of

FSM allows us to eliminate noise by getting rid of isolated pixels. It also allows us to automatically determine the vertical and horizontal thresholds. For example, assume there are some isolated pixels between two lines, then according to (1), if the sum of those pixels is negligible when compared to the sum of the two lines respectively, then those pixels will not be considered and they do not affect the results of FSM. The degree of membership of those pixels is closer to 1 than the degree of membership of the two lines respectively.

The main aim of the FSM is that it can automatically determine the thresholds in any position in the document. Depending on the particular position and the particular application, we can calculate the vertical and horizontal thresholds in order to meet specific requirement. In the case when we need to differentiate a line and a paragraph, then the vertical threshold can be determined by giving an extra weight for each threshold so that realise such an objective to differentiate a line and a paragraph according to the proportion of the threshold.

For the case of a text line with different font types, or font size, or with different languages such as both English and Chinese on the same line, etc., the use of FSM is also very satisfactory (see Fig.3-5). If we use a fixed-threshold approach such as RLSA or RXYC for the segmentation task, this threshold could either be too large or too small. But this problem can be overcome easily with FSM.

Using FSM, we can obtain the distance between any row and column in a document image. We can utilise those features to distinguish further each segmented unit such as a horizontal line, a vertical line, or even English or Chinese character or handwriting or a picture. It also allow us to better determine the characteristics of each segmented unit such as the length, the width, the height, the mid-point, the density, etc. This information will be helpful when we classify the blocks into text, picture, graphics, lines, etc. And also when we try to classify them according to document types.

3.5 Paragraph Block

For some office document, according to our estimate, the blocks dealt with FSM can merge within 20 paragraph blocks. For the part of head and tail of the image file, it is best to merge little blocks, for the part of middle image file, to merge more blocks. The rule of the mergence is a distance between rows or columns.


3.6 Experimental Results

The images used for our experimental were obtained by scan with 400 DPI. The figures from 3-5 to 3-13 show the results of the experiments. Fig.3-5 shows the source document for testing the FSM. Fig.3-6 shows the histogram of the distribution of black pixels in a row by row fashion from top to bottom (i.e. from 0 to 783). Fig.3-7 shows the “black block” found by the methodology proposed above. Fig.3-8 shows the distribution of the black pixels in the segmented image shown in Fig.3-7. From Fig.3-6 and 3-8, their differences are easily noticeable. To make them clearer, we show in Fig.3-9 the degree of membership for the fuzzy set “blank line” for the original (dotted line) and the processed image (solid line) using FSM. When compared with a fixed threshold non-fuzzy approach such as RXYC and RLSA, the differences can be found in Fig.3-10 to 3-17. From these figures, we can see the result of the Fig.3-10 is relatively more satisfactory when compared to Fig.3-11 and Fig.3-12. However, Fig.3-13 that shows the result with FSM is the most satisfactory.

3.7 Summary

In this chapter, we presented a fuzzy set based method to automatically determine the threshold for segmentation of document images. Compared to existing techniques such as the RLSA or RXYC, the proposed FSM approach is better able to determine the boundaries of the different blocks in a document. The definition of fuzzy blank and black box allow the boundaries to be more accurately located. It also eliminates noise more efficiently. Experimental results have confirmed that the FSM does have such advantageous features.

(0,0) Form No. H12316



HONG KONG INSTITUTION OF SCIENCE
Application for Ordinary Membership

FORM NO. H12316
Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should visit to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; application made on duplicated forms will not be accepted.

NAME IN ENGLISH: Paul H. Fung (F=36)
Surname Other names Chinese

HKID No.: 626919

Note:
(1) Names should be as they appear on Hong Kong Identity Card
(2) If you do not hold a Hong Kong Identity Card, then voter passport number and also attach evidence showing a substantial connection with Hong Kong.

NAME IN CHINESE: 馮浩波 (F=36)
Surname Other names Chinese

HKID No.: 626919

Note:
(1) Names should be as they appear on Hong Kong Identity Card
(2) If you do not hold a Hong Kong Identity Card, then voter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: Cable Park, 2nd Greenway,
Mong Kok, Kowloon, Hong Kong

Telephone: 2693-2652

OFFICE Address: Cable Park, 2nd Greenway,
Mong Kok, Kowloon, Hong Kong

Telephone: 2693-2652

Fig.3-5 The original document

Fig.3-7 After applying Algorithm 4

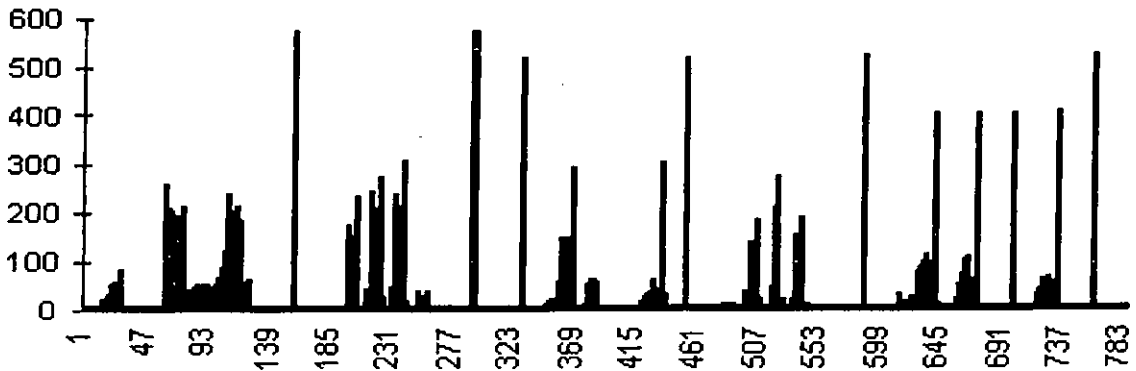


Fig.3-6 The pixel distribution of the original document

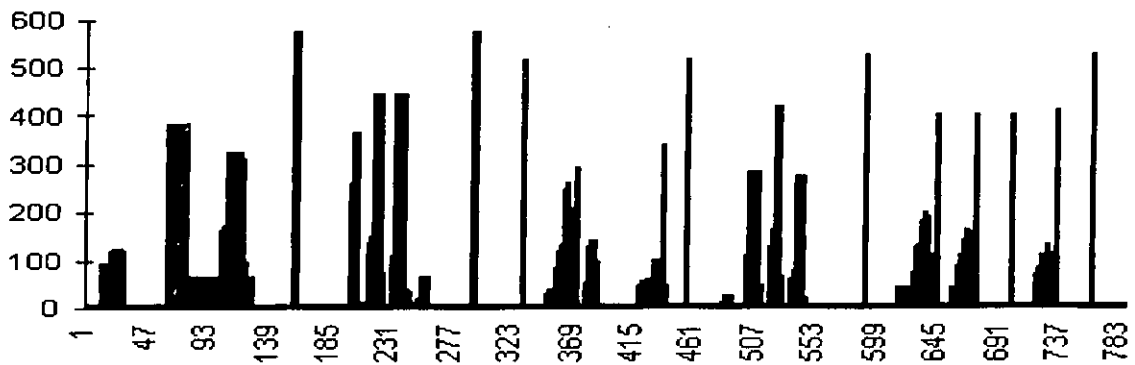


Fig.3-8 The pixel distribution of the document in Fig. 3-7

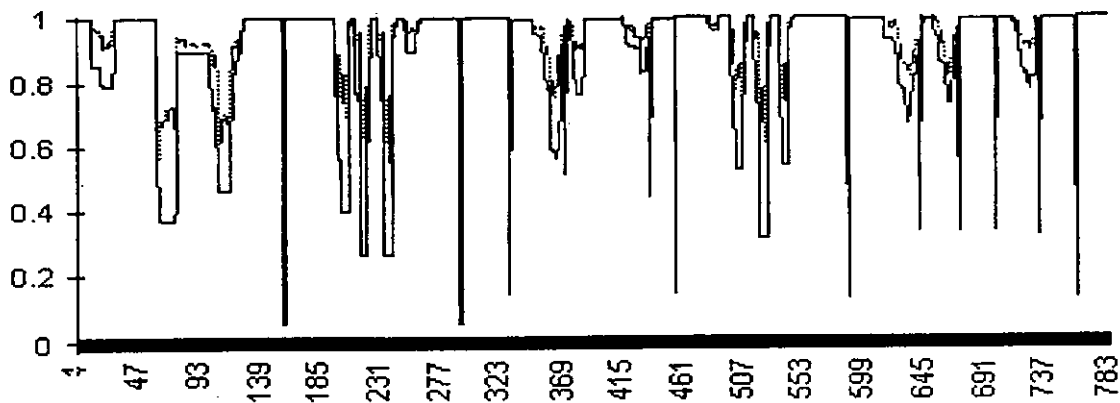


Fig.3-9 The difference in degree of membership between Fig. 3-5 (dotted line) and Fig. 3-7 (solid line)

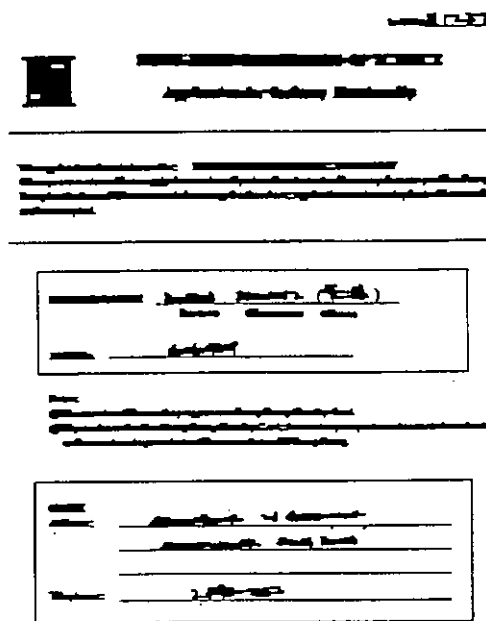


Fig.3-10 The result of using fixed threshold with $C_h = 10$ and $C_v = 4$

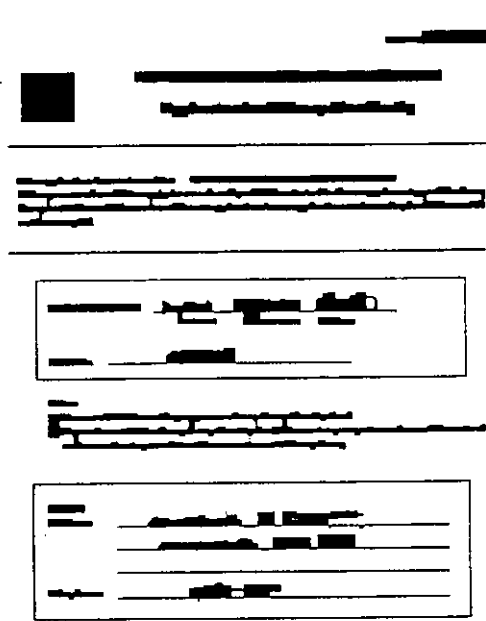


Fig.3-11 The result of using fixed threshold with $C_h = 10$ and $C_v = 8$

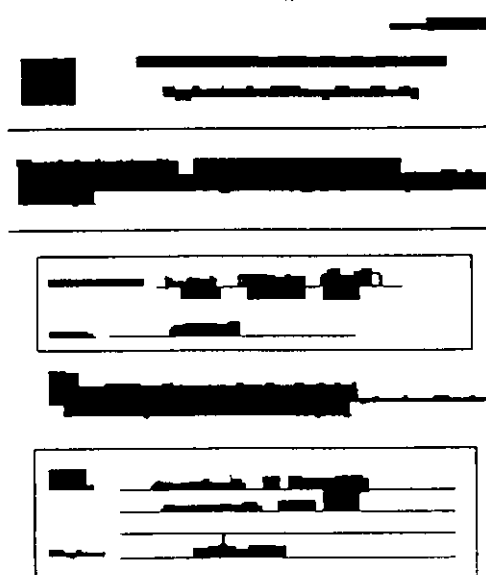


Fig.3-12 The result of using fixed threshold with $C_h = 10$ and $C_v = 14$

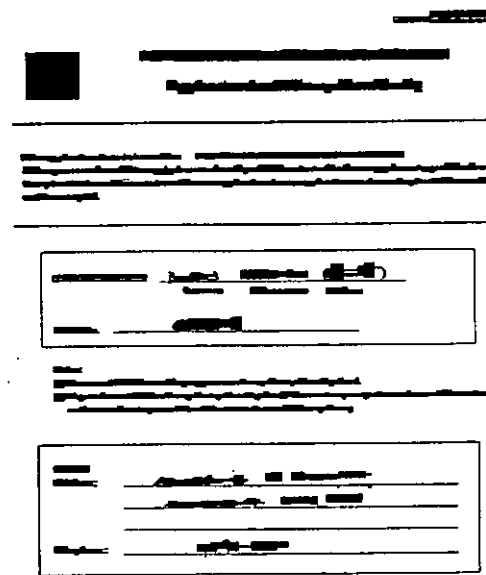


Fig.3-13 The result using FSM

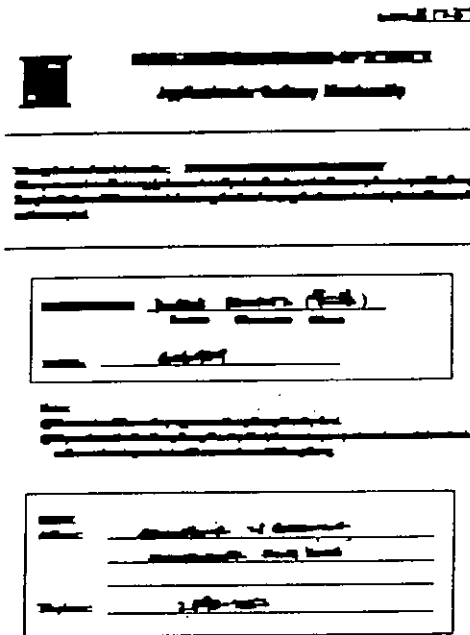


Fig.3-14 The result of using fixed threshold with $C_h = 12$ and $C_v = 4$

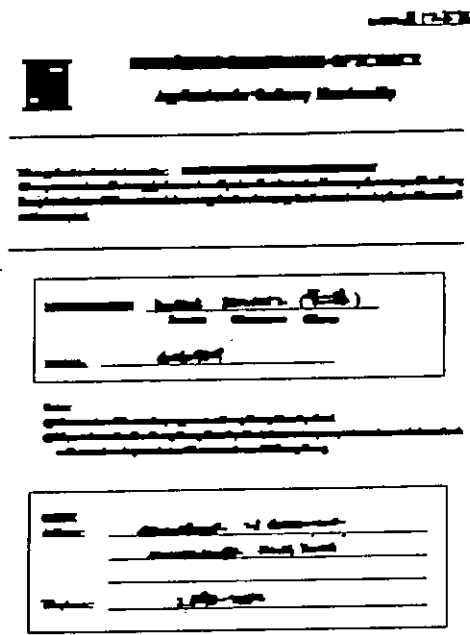


Fig.3-15 The result of using fixed threshold with $C_h = 14$ and $C_v = 4$

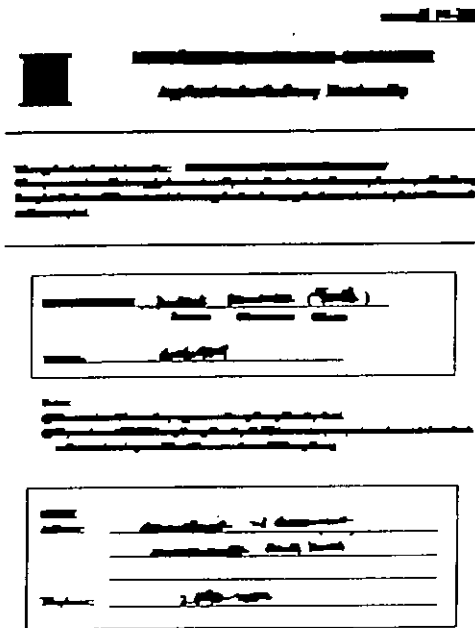


Fig.3-16 The result of using fixed threshold with $C_h = 16$ and $C_v = 4$

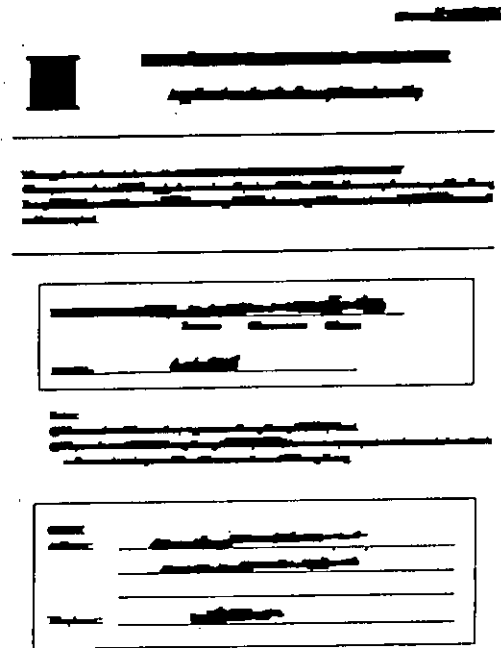


Fig.3-17 The result of using fixed threshold with $C_h = 20$ and $C_v = 4$

Chapter 4 Block Classification of Document

In this chapter, the four methods for block classification were surveyed. The shortcoming of them is shown. Then APACS [62] method is applied to the block classification. It only uses for representing the reason why the block classification is needless discussing in the later chapter. The algorithm, features, and how to select the samples is specified in detail. The questions about how to select the sample and how to classify the images are prompted. There is a criterion, and all on the basis of this criterion to classify the source image, otherwise it will generate different result.

4.1 Introduction

After segmentation, each block is classified into several basic classes: halftone image, text, graphics, vertical and horizontal lines. In order to accomplish such tasks, at first, it is necessary to extract appropriate features from each block. To be located and classified the blocks according to the content, a unique labeling technique that two-dimensional plane consisting of mean value of the block height versus run length of the block mean black pixel is established to classify document blocks into text, non-text, horizontal line, and vertical line [1]; a newspaper classification method brings in the black-white pair run-length matrix (BW Matrix) and black-white-black combination run-length matrix (BWB Matrix) to deduce three features: short run emphasis, long run emphasis, and extra long run emphasis for clustering [3]; a rule-based classification uses the features such as height, aspect ratio, density, perimeter, $\text{perimeter}^2/\text{area}$, and $\text{perimeter}/\text{width}$ ratio and some segmentation rule parameters [12]; and the clustering rules the block classification classifies each block into one of text, horizontal or vertical lines, graphics, and pictures [21].

Labeling technique

Using the labeling technique the following measurements are taken. These are total number of black pixels in the segmented block, minimum $x - y$ coordinates of a block and its $x - y$ lengths, total number of black pixels in the original block, and number of horizontal white-black transitions in the original image block, etc. Given these measurements, several features are computed. These are the height of each block (H), its eccentricity (E), the ratio of the number of black pixels to the area of the surrounding rectangle (S), and the mean horizontal length of the black runs in the original data from the block (R). A block is considered to be text if it is a textured

stripe of mean height H_m and mean black-run length R_m . The distribution of values in the $R - H$ plane derived from sample documents are observed to determine the discriminating function. Low R and H values represent regions containing text. To determine the threshold values of R and H that define the text region in the $R - H$ plane, an adaptive method is used. This method estimates H_m , R_m and the standard deviation of R and H . These values are then used to classify the various blocks by using the following pattern classification scheme that assumes linear separability:

- Text, if $R < C_{21}R_m$ and $H < C_{22}H_m$;
- Horizontal solid black lines, if $R > C_{21}R_m$ and $H < C_{22}H_m$;
- Graphic and Halftone images, if $E > 1/C_{23}$ and $H > C_{22}H_m$; and
- Vertical solid black lines, if $E < 1/C_{23}$ and $H > C_{22}H_m$.

The constants C_{ij} are determined by heuristic rules from examining the $R - H$ plane plot of typical document and the values of R_m and H_m .

Newspaper classification method

The newspaper classification method it involves in two matrices, i.e. BW matrix and BWB matrix. For the BW Matrix, the texture of a block of text is characterized by the fundamental elements that are line segments with different widths for different font sizes, and the line segments assemble with certain density. To represent these properties, the black-white pair run length matrix is defined as a set of consecutive black pixels followed by a set of consecutive white pixels. The length of the run is the number of pixels in the run. For simplification, various combinations of black-white pair runs with different proportions of length will be quantified into nine categories: 1, 2, ..., 9. The category number represents the percentage (within an interval) of white part in a black-white pair run. The matrix element $p(i, j)$ specifies the number of times that the image contains a black-white pair run of length j , in the horizontal direction, consisting of white pixel runs having length as many as $10 \cdot i$ percent of j .

For the BWB Matrix, first, the black-white-black combination run is defined as a pixel sequence in which two black pixel runs are separated by a white pixel run. The length of the run is defined as the number of white pixels in the white pixel run. The length of a black pixel run is fixed and assigned into one of three categories. Both black pixel runs should have approximately the same length and lie in the same category. The matrix element $p(i, j)$ is the number of times that the image contains a

black-white-black combination run, in the horizontal direction, with white pixel run length j and black pixel runs with length lying in category i . So the BWB matrix should have three rows.

From BW matrix can be derive two features, i.e. Short Run Emphasis and Long Run Emphasis. Short Run Emphasis can check the letters belonging to small letter blocks or large letter blocks. Long Run Emphasis can check the letters whether are large letter blocks.

Rule-based classification

Rule-based classification is mainly using segmentation rules and segmentation rules parameters to determine each block classification features. The parameters include two types, i.e. static parameters and adjusted parameters. For the static parameters (e.g., density, aspect ratio, perimeter to width ratio, and perimeter squared to area ratio thresholds) are dimensionless and therefore are invariant to type and style. However, perimeter filter parameters need to be adjusted when processing at different document resolutions. This may be necessary since pixel averaging resolution reduction operations affect character perimeter values.

Clustering rules the block classification

Let the origin of the document image be located at the upper-left corner. Each block is measured in terms of the following:

- a) Minimum x - and y -coordinates (i.e. the upper-left corner) of a block and its width and height ($x_{\min}, y_{\min}, \Delta x, \Delta y$).
- b) Total number of black pixels in a block of the original image (N).
- c) Horizontal transitions of white to black pixels in a block of the original image (TH).
- d) Vertical transitions of white to black pixels in a block of the original image (TV).
- e) When a block of the original image is projected onto x -axis, the number of columns in which black pixels exist (δx).

Since in most cases the projection profile of a block onto y -axis contains black pixels in each row, it makes redundant to measure the number of rows in which black pixel exists. The features used in block classification can be calculated:

1. The height of each block, $H = \Delta y$.
2. The ratio of width to height (or aspect ratio), $R = \frac{\Delta x}{\Delta y}$.

3. The density of black pixels in a block, $D = \frac{N}{\Delta x \Delta y}$.
4. The horizontal transitions of white to black pixels per unit width, $TH_x = \frac{TH}{\delta x}$.
5. The vertical transitions of white to black pixels per unit width, $TV_x = \frac{TV}{\delta x}$.
6. The horizontal transitions of white to black pixels per unit height, $TH_y = \frac{TH}{\delta y}$.
7. The vertical transitions of white to black pixels per unit height, $TV_y = \frac{TV}{\delta y}$.

Let TH_x^{\max} and TH_x^{\min} denote the maximum and minimum TH_x s values of all characters, respectively, and similar notations TV_x^{\max} and TV_x^{\min} are used intuitively,

$$TH_x^{\min} \leq TH_x \leq TH_x^{\max}, \text{ and}$$

$$TV_x^{\min} \leq TV_x \leq TV_x^{\max}$$

Let H_m be the average height of mostly popular blocks. The rule-based block segmentation algorithm is described as follows:

1. Rule 1: if $c_1 H_m < H < c_2 H_m$, this block belongs to text.
2. Rule 2: if $H < c_1 H_m$ and $c_{h1} < TH_x < c_{h2}$, this block belongs to text.
3. Rule 3: if $H < c_1 H_m$, and $R > c_3$, and $0.9 < TV_x < 1.1$, this block is horizontal line.
4. Rule 4: if $\Delta x < c_1 H_m$, $R < 1/c_3$, and $0.9 < TH_y < 1.1$, this block is a vertical line.
5. Rule 5: if $H > c_2 H_m$, $c_5 < \frac{\delta x}{\Delta x} < c_6$, and $c_{h1} < TH_x < c_{h2}$, this block belongs to text.
6. Rule 6: if $D < c_4$, this block belongs to graphics.
7. Rule 7: otherwise, this block is a picture.

Where $c_1, c_2, c_3, c_4, c_5, c_6, c_{h1}, c_{h2}$ is parameters.

4.2 Block Classification Using ACAPS

For the block classification, all method [1][3][12][21] use the experienced parameter as criteria standard that the block belongs to. It is important to select the parameter according to different facility. Escaping to subjective mistake, and

according to the features of the block classification that (i) each block classification types as text, graphics, horizontal line, and vertical line is expressed in symbolic form, and (ii) the variables have to be made based on symbolic representations of each block, used the APACS that is a method for the efficient acquisition of classification rules from training instances which may contain inconsistent, incorrect, or missing information [62][70].

After segmentation using FSM, let $P = \{T, G, H, V\}$, where T, G, H and V is denoted text, graphics, horizontal line and vertical line respectively, the total elements are typed as 4 class block. Let $K = \{dx, dy, N, TH, TV, tx, R, D\}$, where dx, dy, N, TH, TV, tx, R , and D is denoted block width, block height, total number of block pixels, horizontal transitions of white to black pixels in a block of the original image, vertical transitions of white to black pixels in a block of the original image, the number of columns in which black pixels exist when a black of the original image is projected onto x-axis, the ratio of width to height, and the density of black pixels in a block respectively.

4.3 The Algorithm of ACAPS

The ACAPS consists of three phases: (i) detect the patterns inherent attribute values of the objects via the selected sample, (ii) construct the prediction rules based on the detected patterns, and (iii) use of these rules to predict the characteristics of future objects.

To depict the three phases, following [62], suppose that there is an ordered sequence training instances that contains N objects, each of which belongs to one of P classes, $c_p, p = 1, \dots, P$. Suppose also that each object in the sequence is scribed as n distinct attributes, $a_1, \dots, a_j, \dots, a_n$, so that, in any instantiation of the object description, an attribute a_j takes on a specific value, $val_j \in domain(a_j) = \{v_{jk} \mid k = 1, \dots, J_j\}$, which may be numerical or symbolic, or both. In general, the number of the training sample of the certain object is equal or bigger than that the number of the attributes n multiply the number of classes P , and multiply the experiential parameter 5, i.e. $n \times P \times 5$.

As an illustration, construct a two-dimensional contingency table with P rows and K columns (shown as table 1), where P denotes the total number of the classes, K denotes the total number of different values that a_j can be take on, and let o_{pk} be the total number of objects in c_p characterized by v_{jk} and e_{pk} be the total number of objects

expected to have the characteristic v_{jk} . The o_{p+} is the total number of objects in the training set that are in c_p , and o_{+k} is the total number of training objects that have the characteristic v_{jk} . The $M = \sum_{p,k} o_{pk}$, due to the possibility of having missing values in the data, it is less than or equal to the total number of samples N . i.e. $M \leq N$.

Table 1. A two-dimensional contingency table with P rows and K columns

Class	a_j				Totals	
	v_{j1}	...	v_{jk}	...		v_{jK}
c_1	o_{11}	...	o_{1k}	...	o_{1K}	o_{1+}
\vdots	(e_{11})	...	(e_{1k})	...	(e_{1K})	\vdots
c_p	o_{p1}	...	o_{pk}	...	o_{pK}	o_{p+}
\vdots	(e_{p1})	...	(e_{pk})	...	(e_{pK})	\vdots
c_P	o_{P1}	...	o_{Pk}	...	o_{PK}	o_{P+}
\vdots	(e_{P1})	...	(e_{Pk})	...	(e_{PK})	\vdots
Totals	o_{+1}	...	o_{+k}	...	o_{+K}	M

Then the first phase is used the *adjusted difference* to measure the pattern inherent attributes, and it can be constructed as follows:

$$(1) \quad e_{pk} = \sum_{i=1}^K o_{pi} \sum_{i=1}^P o_{ik} / M$$

$$(2) \quad z_{pk} = (o_{pk} - e_{pk}) / \sqrt{e_{pk}}$$

$$(3) \quad v_{pk} = (1 - \frac{o_{p+}}{M})(1 - \frac{o_{+k}}{M})$$

$$(4) \quad d_{pk} = z_{pk} / \sqrt{v_{pk}}$$

The expression (4) is the adjusted difference, and using it as checking criteria. If $d_{pk} > +1.96$, it indicates that the presence of v_{jk} is a relevant feature for c_p . If $d_{pk} \leq -1.96$, it indicates the absence of v_{jk} is a relevant feature for c_p . If $-1.96 < d_{pk} \leq 1.96$, the values of a_j that show no correlation with any class yield no information on how an object should be classified. Such values are *irrelevant* for the learning process. Their presence may cause overfitting and the generation of misleading classification rules, and hence they are discarded from further analysis.

The prediction rules based on the detected patterns can be construct as following forms:

Rule: If *<condition>* then *<conclusion>* with weight of evidence W .

Where the condition part specifies the attribute values that an object should possess if it is to belong to the class predicted by the conclusion part. W is the weight of evidence associated with the rule.

Suppose that v_{jk} is a relevant feature for c_p . An object characterized by v_{jk} is more likely to belong to c_p than to other classes. This information can be represented in the form of a rule as follows:

If a_j of an object is v_{jk} then that an object belongs to c_p is with weight of evidence $W(\text{object in } c_p / \text{object not in } c_p \mid \text{object characterized by } v_{jk})$.

Where W measures the amount of positive or negative evidence that is provided by v_{jk} supporting or refuting an object that it characterizes to be classified into c_p . And it can be expressed, equivalently, as

$$(5) \quad W(\text{Class} = c_p / \text{Class} \neq c_p \mid v_{jk}) = \log \frac{\Pr(v_{jk} \mid \text{Class} = c_p)}{\Pr(v_{jk} \mid \text{Class} \neq c_p)}$$

And the last phase can be described as follows:

Suppose that the *obj* to classified is described by the n attributes, and only m ($m \leq n$) of them, $\text{val}_{[1]}, \dots, \text{val}_{[j]}, \dots, \text{val}_{[m]}$ with $\text{val}_{[j]} \in \{\text{val}_j \mid j = 1, \dots, n\}$, are found to match one or more classification rules, then based on the weight of evidence measure, *obj* is assigned to c_p if

$$(6) \quad \begin{aligned} &W(C_{obj} = c_p / C_{obj} \neq c_p \mid \text{val}_{[1]}, \dots, \text{val}_{[m]}) \\ &> W(C_{obj} = c_h / C_{obj} \neq c_h \mid \text{val}_{[1]}, \dots, \text{val}_{[m]}) \\ &h = 1, 2, \dots, P' \text{ and } h \neq p \end{aligned}$$

Where P' ($\leq P$) denotes the number of classes that partially matched by the attribute values of *obj*. It should be noted that it is possible for two different plausible values to have the same greatest weight of evidence. In this case, there may be more than one plausible class assignment for *obj*. On the other hand, if there is no evidence for or against any specific class assignment, classification may either be refrained in order to avoid the furnishing of an inaccurate one or that *obj* can be assigned to the class to which the majority of training objects belong. If it happens that there is no relevant value for determining of *obj* is completely nondeterministic or there is a lack of training instances for the learning process.

4.4 The features of using the ACAPS

The notable characteristics of the ACAPS include: (1) its ability to identify the

values of an attribute that provide important information for the characterization of a class of instances; (2) its ability to quantitatively measure, combine, and compare the evidence concerning the class assignment of an instance whose descriptions do not satisfy that of any class completely; (3) its ability to accommodate the uncertain and nonhomogeneous nature of human concepts through the use of the weight of evidence which may be interpreted as a measure of an object's typicality of a class of instances; (4) its ability to accommodate an important aspect of intelligence behavior - the human ability to allow a certain degree of variation in their decision criteria when they are faced with uncertainty; (5) its ability to avoid the construction of rules that are too specific, that have weak predictive power, and that are unable to distinguish signal from noise; thereby, it overcomes the problem of overfitting; (6) its ability to make the rule space less complex without having to sacrifice classification accuracy, since it is capable of discarding irrelevant values early in the learning process; (7) its ability to acquire accurate decision rules without the need for much domain knowledge, which, if available, can also be readily included in the learning process to further improve the efficiency of the classification tasks; (8) its ability to efficiently handle data of high dimensionality even when the training sample size is small and even when the assumption concerning any specific mathematical model for the data cannot be made; (9) its use to aid the knowledge acquisition process for the construction of expert systems which perform tasks that are classificatory in nature; (10) the proposed classification method can be easily extended to deal with some forms of structure-valued data.

4.5 The Implementation of using the ACAPS

Using the ACAPS, at first the samples are selected according to the experiential expression $n \times P \times 5$. For example, the three classes of the source image samples shown in the Fig.4.1. In the shown figure, the feature dx for each block is only discussed according to the 5 ranks. The table 2 depicts the image distributed among three classes. The table 3 shows the contingency how dx with 5 ranks are distributed among 3 classes.

Therefore, the $P=3$, $K=5$, i.e., there are 3 classes: graph, text and line, and there are 5 ranks: 1~35, 36~50, 51~85, 86~100, and 101~200. The graph, text, and line class has 16, 11, and 6 samples respectively. The rank 1~35, 36~50, 51~85, 86~100, and 101~200 has 14, 6, 4, 2, and 7 samples respectively.

The $o_{11} = 9$, it means that the graph class in rank 1~35 has 9 samples, i.e., #1, #2, #3, #4, #5, #6, #7, #8, and #9. The $o_{12} = 3$, it means that the graph class in rank 36~50 has 3 samples, i.e., #12, #13, and #14, and so on.

The $e_{11} = \frac{\sum_{i=1}^5 o_{1i} \sum_{i=1}^3 o_{i1}}{33} = \frac{(o_{11} + o_{12} + o_{13} + o_{14} + o_{15})(o_{11} + o_{21} + o_{31})}{33} = \frac{(9+3+2+0+2)(9+4+1)}{33} = 6.788$, the $e_{12} = \frac{\sum_{i=1}^5 o_{1i} \sum_{i=1}^3 o_{i2}}{33} = \frac{(o_{11} + o_{12} + o_{13} + o_{14} + o_{15})(o_{12} + o_{22} + o_{32})}{33} = \frac{(9+3+2+0+2)(3+2+1)}{33} = 2.909$, and so on.

Next the rules are generated according to ACAPS by the samples.

We can calculate the value d_{pk} in farther, but it is needless because we can deduce the result as follows:

1. Only selected enough samples, the result is approving. Then we can construct the valid rules according to ACAPS.

2. It is very important how to select the samples and how to collect the samples. For different case, the samples must be different.

3. It is difficult to classify the images belong to which classification. For example, the *COMP* is classified as whether graphic or text. At first, there is a criterion, and all on the basis of this criterion to classify the source image, otherwise it will generate different result.

4. The number of class can be considered as 4, i.e., graph, text, horizontal line, and vertical line, but this is also a fuzzy notion.

5. It is too difficult to confirm the number of the rank for the document image classification. It is impossible to use ACAPS to block classification of document if the number of rank don't confirm.

6. To analysis the problems, we will continue to use ACAPA in the next chapter (chapter 5). The number of the rank is confirmed by my subjectivity, it is only for test, but in the chapter 6 this method will not use again.

Table 2. Showing d_x with 5 ranks are distributed among 3 classes

D_x	1~35	36~50	51~85	86~100	101~200
Graph	#1, #2, #3, #4, #5, #6, #7, #8, #9	#12, #13, #14	#10, #11		#15, #16
Text	#17, #18, #19, #22	#20, #21		#23	#24, #25, #27, #28
Line	#31	#29	#26, #33	#32	#30

Table 3. A contingency showing how d_x with 5 ranks are distributed among 3 classes

Class	1~35	36~50	51~85	86~100	101~200	Totals
Graph	(9, 6.788)	(3, 2.909)	(2, 1.939)	(0, 0.970)	(2, 3.394)	16
Text	(4, 6.667)	(2, 2.000)	(0, 1.333)	(1, 0.667)	(4, 2.333)	11
Line	(1, 2.121)	(1, 0.909)	(2, 0.606)	(1, 0.303)	(1, 1.061)	5
Totals	14	6	4	2	7	33

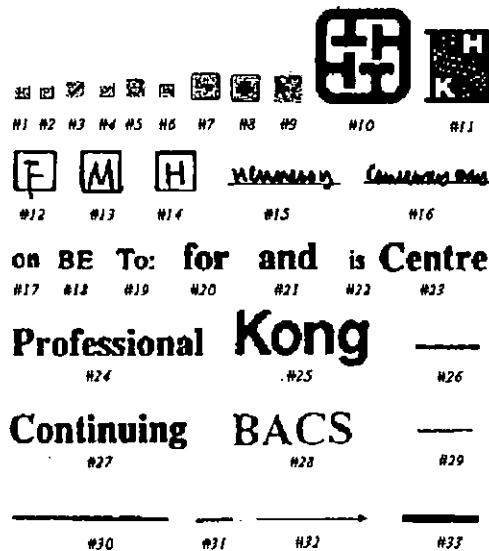


Fig.4.1 Three classes of source image

Chapter 5 Document Classification Using Branch-and-Bound Technique Based on ARG Matching

This chapter depicts mainly the document classification using Branch-and-Bound Technique based on ARG (attributed random graph). ARG construction, decision tree and using branch-and-bound to match two ARGs are presented in detail. The document classification is successfully transferred to how to match the two ARGs.

5.1 Introduction

After segmentation and block classification, the final recognition stage is document classification. ARG is adopted for this stage. This recognition stage includes following contents:

- For each document, the type of classified blocks and relations of classified blocks construct an ARG. Then decision trees are constructed using ARG. Finally, the branch-and-bound technique is applied to matching two ARG's.
- ARG classification training process is by process, in which the ARG's classification of the document classification is generated through build the ARG from documents and synthesis them within the same classification.
- In ARG classification training process, in which the ARG's classification of the document classification is generated through build the ARG from documents and synthesis them within the same classification.
- ARG matching in which the similarities between the incoming document image and predefined document classes are matched.
- Document Classification in which the incoming document image is classified into the class belonged by the largest value of similarity.

The Branch-and Bound Technique is adopted to match two ARG's of a document. A new evaluation function of the Branch-and-Bound algorithm [71] is developed.

5.2 Construct Attribute Random Graph

A graph $T = (V_N, V_A)$ consists of two sets, where V_N is the set of node labels and V_A is the set of arc labels, respectively. Any element belonging to V_N or V_A has the form (u, v) , where u is a syntactic symbol denoting the structure and $v = (v_1, v_2, \dots, v_n)$ is a semantic vector denoting n numbered attributes.

An *attributed random graph* (ARG) over T is a tuple $\Omega = (N, A, \xi, \zeta)$, where N is a

non-empty finite set of nodes, $A \subset N \times N$ is a set of distinct ordered pairs of distinct elements in N called arcs; $\xi: N \rightarrow V_N$ is a node interpretation function; and $\zeta: A \rightarrow V_A$ is an arc interpretation function.

Nodes of an ARG: For classified blocks, the syntactic of a node has a value chosen from $v_i \in \{T, G, V, H\}$ ($1 \leq i \leq n$). The semantic is ordinal used to denote each vector.

Relations of nodes in an ARG: For classified blocks, the syntactic of a relation r_{ij} between block i and block j is represented as a vector $r_{ij} = (a_{ij}^1, a_{ij}^2, a_{ij}^3, a_{ij}^4)$, where a_{ij}^1, a_{ij}^2 are denoted as the x -axis, y -axis of the center point of the line from block i to block j , respectively; a_{ij}^3 is the distance from node v_i to v_j when v_i locates "left to" v_j , if v_i is right to v_j , then a_{ij}^3 is 0; and a_{ij}^4 is the distance from node v_i to v_j when v_i locates "up to" v_j , if v_i is down to v_j , then a_{ij}^4 is 0. In Fig.5.1, $a_{ij}^1 = x_i + x_j$, $a_{ij}^2 = y_i + y_j$, $a_{ij}^3 = x_j - x_i$, $a_{ij}^4 = 0$. The semantic is not used.

Using the above definitions, the constructed ARG is a completed relation graph. This is very important for the next two ARG matching process.

5.3 Construct Decision Tree

Assume that two ARG's are $G = (N_{pattern}, A_{pattern}, \xi, \zeta)$ and $H = (N_{base}, A_{base}, \xi, \zeta)$, respectively, and G is a pattern graph with order m , H is the base graph with order n ($m \leq n$). In the special case where $m = n$, the problem is to find the optimal isomorphism between G and H . To accomplish this goal, the decision tree is constructed first. It has height m , and $n - p$ sons for each node at level $p = 0, 1, \dots, m - 1$. Therefore, at any level $p > 0$, there is a path that consists of ordinal nodes in the decision tree from the root to node N , where

$$N = \{(i, q_i) \mid i = 1, 2, \dots, p\} \quad (1)$$

and from node N to a leaf node

$$T = \{(i, q_i) \mid i = 1, 2, \dots, m\} \quad (2)$$

Using the branch-and-bound technique to search through the decision tree, we can identify the optimal monomorphism between the pattern ARG and the base ARG. Then the document classification is transferred to the problem of how to match two ARG's.

5.4 ARG matching Use Branch-and-Bound Technique

From expression (1) and (2), the set of the pattern ARG and base ARG is

correspondingly divided into two parts, i.e.

$$N_{pattern} = N_1 \cup N_2 = \{1, 2, \dots, p\} \cup \{q_1, q_2, \dots, q_p\}$$

$$N_{base} = M_1 \cup M_2 = \{p+1, p+2, \dots, m\} \cup \{q \mid q \in \{1, 2, \dots, n\} \setminus N_2\}$$

The branch-and-bound technique used to solve the problem requires an evaluation function that assigns a cost to the branch incident to each node N of the tree. Let each node $N = \{(i, q_i)\}$, where $i \in N_1$, $q_i \in N_2$, and $v_i = v_{q_i}$ ($v_i \in N_{pattern}$, $v_{q_i} \in N_{base}$), assume that $N = \{(1, q_1), (2, q_2), \dots, (p, q_p)\}$ indicates the unique path from the root to node N , then the cost $k(p, q_p)$ assigned to the branch incident to N is defined as

$$k(p, q_p) = \sum_{j=1}^{p-1} c'((j, p), (q_j, q_p))$$

where $c'((a, b), (c, d)) = c((a, b), (c, d)) + c((b, a), (d, c))$,

and $c(x, y)$ is the mapping cost between the graph elements x and y in G and H , respectively.

Referring to [71], the evaluation function f^* is defined as the value of $f^*(N)$ at a node $N = \{(i, q_i) \mid i = 1, 2, \dots, p\}$, which is the cost $g^*(N)$ of an optimal path from the root to node N plus the cost $h^*(N)$ of an optimal path from node N to a leaf $T = \{(i, q_i) \mid i = 1, 2, \dots, m\}$, i.e.

$$f^*(N) = g^*(N) + h^*(N)$$

$$g^*(N) = \sum_{i=1}^p k(i, q_i)$$

$$h^*(N) = \min_t \left\{ \sum_{i=p+1}^m k(i, q_{it}) \right\}$$

where t denotes a feasible path from N to T .

If the consistent lower bounded estimate value $h(N) \leq h^*(N)$, then use $h(N)$ replace the $h^*(N)$ for each node N , i.e.

$$f^*(N) = g^*(N) + h(N)$$

Let $k'(i, q)$ be the cost of adding a pair of vertices (i, q) to N , where $i \in M_1$ and $q \in M_2$.

Define

$$k'(i, q) = \sum_{j=1}^p c'((i, j), (q, q_j))$$

for each unmatched vertex $i \in M_1$, we find a corresponding vertex $q \in M_2$ such that the cost $k'(i, q)$ is minimized. Note that such a mapping $H: M_1 \rightarrow M_2$ may be many-

to-one. Let $a(N)$ be the total cost of H . We have

$$a(N) = \sum_{i=p+1}^m \min_{q \in M_1} k'(i, q)$$

Similarly, $b(N)$ can be defined as the total cost of an optimal mapping $H': M_1 \times M_2 \rightarrow M_1 \times M_2$. i.e., for each arc (i, j) with endpoints $i, j \in M_1$, a minimum cost mapping $H': (i, j) \rightarrow (q, r)$ is found, such that (q, r) is an arc with endpoints $q, r \in M_2$. H' can also be many-to-one. Hence,

$$b(N) = \sum_{\substack{ij \in M_1 \\ i < j}} \min_{\substack{q \neq r \\ q, r \in M_2}} c'((i, j), (q, r))$$

With $a(N)$ and $b(N)$ defined, their summation is used as the lower bounded estimate of $h^*(N)$, i.e.,

$$h(N) = a(N) + b(N)$$

Therefore, $f(N) = g^*(N) + a(N) + b(N)$.

5.5 Application Algorithm for Matching ARG

Initiation

For level p from 1 to m do

{

For loop times from 1 to $n-p+1$ do

{

Set the set $N_1 = \{1, 2, \dots, p\}$.

Set the set $N_2 = \{\text{surplus nodes}\}$.

Set the set $M_1 = \{p+1, \dots, m\}$.

Set the set $M_2 = \{N_{base} \setminus N_2\}$.

if v_i isn't equal v_{q_i} , then $f(N)$ is assigned infinity,

otherwise compute $f(N) = g^*(N) + a(N) + b(N)$.

Record the $f(N)$

}

To find the minimum $f(N)$, then assign the matching nodes.

To find the surplus nodes.

}

Output the optimal solution and path.

The optimal solution and path of an example is shown in Fig.5.2. The parameters

in the two graphs are from [71], and the content of nodes is only modified to show differences. For simplicity, we set the $c(x, y) = |s - t|$, where s and t are the attribute values of x and y , respectively. The decision tree is shown in Fig.5.3.

5.6 Whole Procession of DIC

Connection with previous chapters it can be consists of the whole procession of the document image classification. This part depicts the whole procession and resultful analysis.

According to the survey, in general, the whole procession of the document image classification is consisted of three stages, i.e. document segmentation, block classification, and document classification.

The functions of document segmentation are mainly to segment the image into blocks by analysis the construction of document and so on. The functions of block classification are to depict the features of each block. The document classification is finally aim that the documents were departed into different classification or group according to the construction and the content.

Therefore, the whole procession of the document image classification can be construction as Fig.5.4. In Fig.5.4, for the segmentation stage, we use the FSM to segment the image into blocks. For the block classification, use APACS to abstract each block's features. This stage involves in two parts. The one part is training procession that consists of their contents, i.e., select enough sample belong to four classes T, H, V, G ; training block class; and construct rules. The another part is testing procession that the construct rules were used for abstracting the features for each block. For the document classification, we use ARG and Branch-and-Bound Technique to classify the document. This stage also involves in two parts. The one part is training procession that consist of five contents, i.e., selected document samples, Construct ARG, Decision Tree, Document Class Training and ARG and Parameters for all Document Class. The another part is testing procession that has also five contents, i.e., inputting the document, Construct ARG, Decision Tree, ARG Matching and find optimal solution, Different Document Class.

The proposed method for solving the document image classification by layout analysis has been implemented in a system called DIC.

5.7 Experimental Results

At first, depict the data structure using in the DIC as follows (reference appendix d

and e):

```
#define MAXV 12
#define MAXA 6

// Attribute to the vertex:
class VertexAttribute
{
public:
    long int Attribute[MAXV]; // information relation to vertex
};

// Attribute to the arc:
class ArcAttribute
{
public:
    long int Attribute[MAXA]; // information relation to two vertexes
};

// Random Graph
class RandomGraph
{
public:
    int      VertexNumber; // the number of vertex
    int      GrpOfVA;      // the groups of vertex attributes
    int      GrpOfAA;      // the groups of arc attributes
    VertexAttribute Vertex[MAXVERTEX+1];
    ArcAttribute  Arc[MAXVERTEX+1][MAXVERTEX+1];
};
```

The training and testing function are relation to the following data files, the structures in details as follows:

(1) The number of the graphs, it means total number graphs for based pattern data:

NumOfG

(2) The number of the vertexes, the number of the type vertex attributes, the number of the type arc attributes, and this image file belongs to which class, all these parameters are saved in order:

NumofV GrpofVA GrpofAA BelongToClass

(3) The 8 attributes for the vertexes v_1, v_2, \dots, v_{1n} : $K = \{dx, dy, N, TH, TV, tx, R, D\}$ (For each meaning reference previous chapter 4). The first subscript denotes the attributes. The second subscript denotes the vertexes. For example, the A_{21} denotes the second attribute for the first vertex.

$$\begin{matrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \end{matrix}$$

... ..
 $A_{81} A_{82} \dots A_{8n}$

(4) The 6 attributes for the arcs such as: Left Up X position, Left Up Y position, Right Down X position, Right Down Y position, (Left Up X position + Right Down X position)/2, (Left Up Y position + Right Down Y position)/2. The first subscripts denote the attributes. The second two subscripts denote that this attribute is from which vertex to which vertex. For example, the A_{121} denotes the first attribute that from the second vertex to first the vertex.

$R_{111} R_{112} \dots R_{11n}$
 $R_{121} R_{122} \dots R_{12n}$

 $R_{1n1} R_{1n2} \dots R_{1nn}$ - the attributes for the first feature R1

 $R_{611} R_{612} \dots R_{61n}$
 $R_{621} R_{622} \dots R_{62n}$

 $R_{6n1} R_{6n2} \dots R_{6nn}$ - the attributes for the 6th feature R6
 ...

In the appendix e, the result is shown in details.

The system DIC has been trained with five group/class documents and each group/class has six documents which selected from different representative documents, and tested with different documents belong to or don't belong to the five group/class. The results are satisfied that DIC can depart the tested document into the group/class that it belongs to, or tells apart it from the five group/class. But there are some problem, details as following: (1) How to selected the samples for training. In the Fig.4.1, as an illustration, we selected the same sample for the graphics, line, and text. In those selected samples which is belong to the graphics, and which is belong to the text, the criteria need ascertain presciently. (2) Compared Fig.5.5 with Fig.5.8, we can find the difference at the log and the title, and the remaining is the same. For those documents, we view them whether or not as one class. (3) For the Fig.5.9 and Fig.5.10, due to the complexity, if we write irregularly to joint unites together, then the result is different in the segmentation, and don't correct to class them. The Fig.5.5 shown as the document labeled with the v_i and the element of the set P . The Fig.5.6 depicted the data of r_{ij} for the ten nodes in the 6 attributes for the arcs.

5.8 Conclusion

In this chapter, we depicted the document classification using Branch-and-Bound technique based on ARG matching and the whole procession of the document image

classification. Use the sample testing our system DIC. According to the results, we have following conclusion.

The FSM can be determined the threshold heuristically in the document image segmentation. The APACS is able to discover the probabilistic patterns in a sequence of objects and to construct prediction rules based on these patterns for future using.

The document classification using the Branch-and-Bound Technique based on the ARG, it transferred successfully the document classification to the two ARG matching.

However there is a shortcoming in DIC. For example, in Fig.5.5, if the node v_2 is a logo denoting other institution supposed as "GUANG ZHOU INSTITUTION OF SCIENCE", and the v_3 is replaced with coincident content, then this documents have same ARG constructions and belong to same classification. In nature, these documents should belong to the different classifications. To overcome the demerit, an advanced method is presented in the next chapter.

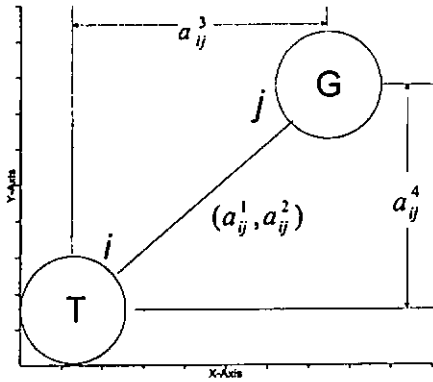


Fig. 5.1 Relations between two nodes

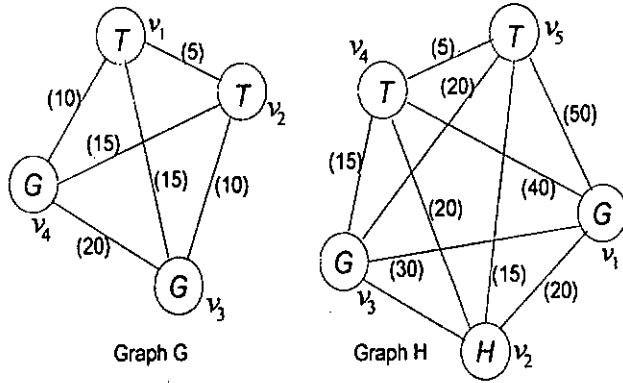


Fig. 5.2 Pattern graph G and base graph H

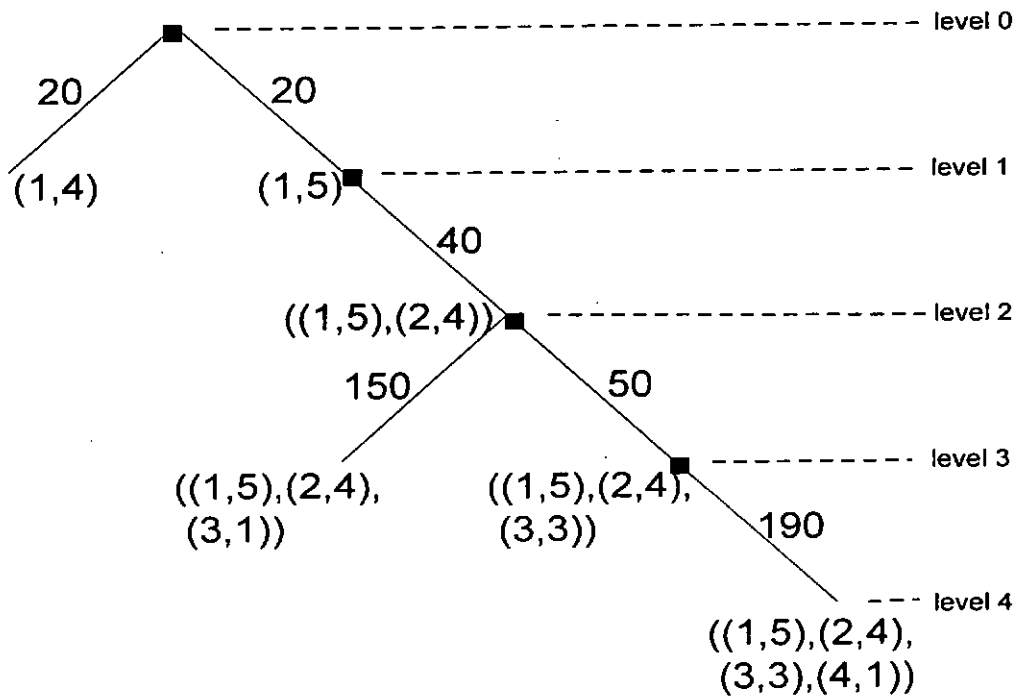


Fig. 5.3 Decision tree for G and H

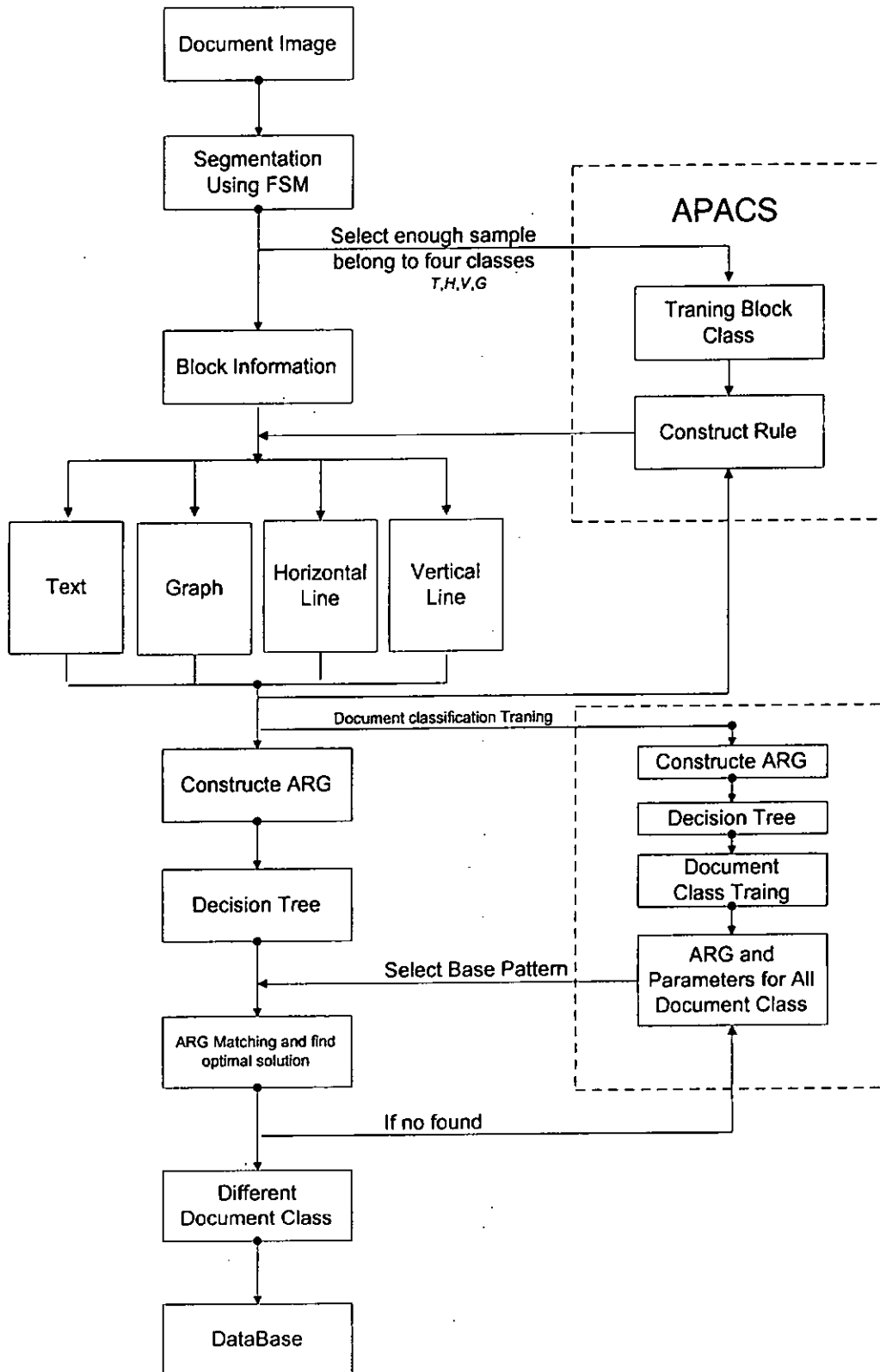


Fig.5.4 An overall organization of the proposed document processing system

Form No. H12376 (1/76)

HONG KONG INSTITUTION OF SCIENCE
Application for Ordinary Membership

The application form is issued to: **HONG KONG POLYTECHNIC UNIVERSITY**
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form, application made on duplicated forms will not be accepted.

NAME IN ENGLISH: TUNG KAO (張浩)
Surname Other names Chinese

HKID No. 6 01 110 (7)

Notes:
(1) Names should be as they appear on Hong Kong Identity Card.
(2) If you do not hold a Hong Kong Identity Card, then please passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: Tung Kuo Jui - 1000 KEE

Telephone: _____

Fig.5.5 The document labelled with v_j and the element of P

0	145	158	134	100	138	138	138	141	138
145	0	129	105	71	109	108	109	112	109
158	129	0	118	84	123	121	122	123	123
134	105	118	0	59	98	97	98	101	98
100	71	84	59	0	64	63	64	67	64
138	109	123	98	64	0	101	102	103	102
138	108	121	97	63	101	0	101	104	101
138	109	123	98	64	102	101	0	105	102
141	112	123	101	67	103	104	105	0	103
138	109	123	98	64	102	101	102	103	0
0	239	232	232	235	213	203	188	168	143
239	0	224	214	217	206	195	180	161	135
232	224	0	218	220	199	189	173	154	128
232	224	218	0	220	199	189	173	154	128
235	227	230	220	0	202	191	176	157	131
213	206	199	199	202	0	170	154	135	110
203	195	189	189	191	170	0	144	125	99
188	180	173	173	176	154	144	0	109	84
168	161	154	154	157	135	125	109	0	65
143	135	128	128	131	110	99	84	65	0
0	15	29	29	23	66	87	118	156	207
0	0	13	13	8	37	71	102	140	192
0	0	0	0	0	37	58	89	127	178
0	0	0	0	0	42	63	94	132	184
0	0	0	0	0	0	20	51	90	141
0	0	0	0	0	0	0	31	69	120
0	0	0	0	0	0	0	0	38	89
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
29	13	0	0	3	0	0	0	0	0
29	13	0	0	3	0	0	0	0	0
23	8	0	0	0	0	0	0	0	0
66	50	37	37	42	0	0	0	0	0
87	71	58	58	63	20	0	0	0	0
118	102	89	89	94	51	31	0	0	0
146	140	127	127	132	90	69	38	0	0
207	192	178	178	184	141	120	89	51	0
0	58	33	33	149	72	74	73	66	73
0	0	0	0	22	91	14	15	14	14
0	23	0	0	116	39	41	40	34	40
0	0	0	0	68	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	8	77	0	1	0	0
0	0	0	0	8	75	0	0	0	0
0	0	0	0	8	76	0	0	0	0
0	0	0	0	14	82	5	7	6	6
0	0	0	0	8	76	0	1	0	0
0	0	0	0	0	0	0	0	0	0
58	0	23	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0
81	23	48	0	0	8	7	8	14	8
149	91	116	68	0	77	73	76	82	76
72	14	39	0	0	0	0	0	3	0
74	15	41	0	0	1	0	1	7	1
72	14	40	0	0	0	0	0	4	0
66	8	34	0	0	0	0	0	0	0
72	14	40	0	0	0	0	0	6	0

Fig.5.6 The data about Fig.5.5 shows as r_{ij}

Form No. H12376

HONG KONG POLYTECHNIC UNIVERSITY
Application for Ordinary Membership

The Hong Kong Polytechnic University
Applications for Attending Staff Development Programs (Local or Overseas) Leading to Academic Award

Please read the Notes for Guidance on the last page before completing the form.

1. Name: CHAN TAI KEVIN Staff No. 1345
Department: Gen. Dr. Zou Post: XYZ Extension: 123
Date of first appointment: 11/196
Terms of service: full / supernumerary local / overseas
Leave balance: (vacation, sick leave) 30
Previous University sponsored staff development leading to academic award in the past 5 years or University funded conference attendance in the last 12 months:
Date: 1/10/95 Amount (HK\$): 10000 at all development/conference attendance
Date: _____ Amount (HK\$): _____ at all development/conference attendance
Date: _____ Amount (HK\$): _____ at all development/conference attendance
(Please use additional sheet if the above space is insufficient)

2. Title of Programme: Language Training Programme
Development
Institution / Location: Gen. Dr.
Mode of study: Full-time / Part-time / Distance Learning
Other (please specify)
Level of Award: Bachelor degree / Master degree / Doctoral
Other (please specify)
Duration of programme: From 96 to 97

Fig.5.7 The source document to be classified

Form No. H12376

HONG KONG POLYTECHNIC UNIVERSITY
Application for Ordinary Membership

This application form is issued to: **HONG KONG POLYTECHNIC UNIVERSITY**
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form, application made on duplicated forms will not be accepted.

NAME IN ENGLISH: YANG Seng-ki (楊生基)
Surname Other names Chinese

HKID No. K07659

Notes:
(1) Names should be as they appear on Hong Kong Identity Card.
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: Opp. 28 Garden View
Happy Valley

Telephone: 2572522

Fig.5.8 The source document to be classified

Chapter 6 Document Classification Based on Advanced ARG Matching

In the previous chapter, we analyze the system DIC using the testing documents. For some documents it can be classified accurately, but for other documents, the result is not perfect. To prompt the capability of DIC, we present the modified algorithm based on Branch-and-Bound Technique discussing in previous chapter for the two ARG matching. The block classification is omitted. The vertex and arc's attributes are only using by the ARG matching algorithm as parameters. That is to say we must compute the attributes for each block, but we needn't to judge the block belonging to which class such as graph, text, horizontal line, or vertical line.

6.1 Shortcoming of the whole procession

The result of the analysis for the testing documents can be reduced as follows:

- The FSM, APACS, and Branch-and-Bound Technique based on ARG, these methods or technique are all advisable methods for the process.
- Even if those documents have same ARG constructions, we can't surely think them belong to the same classification, for example, discussing in the conclusion of chapter 5.
- Using the block classification for document classification, it doesn't use enough the useful features of image.

To overcome the shortcoming above, we needn't abstract each block into the Text, Graph, Horizontal Line or Vertical Line. The block classification is omitted. Matching algorithm directly uses the all features of each block. Using this idea, we can realize enough the information of blocks to tell their subtle difference.

Therefore, the whole procession for the document classification can be constructed shown as Fig.6.1. In Fig.6.1, the abstraction of block features is omitted. The Branch-and Bound Technique is adopted to match two ARG's of a document. A new evaluation function of the Branch-and-Bound algorithm [71] is developed.

6.2 Construct Attribute Random Graph

A *graph* $T = (V_N, V_A)$ consists of two sets, where V_N is the set of node labels and V_A is the set of arc labels, respectively. Any element belonging to V_N or V_A has the form (u, v) , where u is a syntactic symbol denoting the structure and $v = (v_1, v_2, \dots, v_n)$ is a

semantic vector denoting n numbered attributes.

An *attributed random graph* (ARG) over T is a tuple $\Omega = (N, A, \xi, \zeta)$, where N is a non-empty finite set of nodes, $A \subset N \times N$ is a set of distinct ordered pairs of distinct elements in N called arcs; $\xi: N \rightarrow V_N$ is an node interpretation function; and $\zeta: A \rightarrow V_A$ is an arc interpretation function.

Nodes of an ARG: The syntactic of an node is represented as a features vector v_i , and $v_i \in \{v_i^j | j = 1, 2, \dots, 8\}$, where $v_i^1, v_i^2, \dots, v_i^8$ are equal to dx, dy, \dots, D . The semantic is ordinal used to denote each vector.

Relations of nodes in an ARG: For classified blocks, the syntactic of a relation r_{ij} between block i and block j is represented as a vector $r_{ij} = (a_{ij}^1, a_{ij}^2, a_{ij}^3, a_{ij}^4)$, where a_{ij}^1, a_{ij}^2 are denoted as the x -axis, y -axis of the center point of the line from block i to block j , respectively; a_{ij}^3 is the distance from node v_i to v_j when v_i locates "left to" v_j , if v_i is right to v_j , then a_{ij}^3 is 0; and a_{ij}^4 is the distance from node v_i to v_j when v_i locates "up to" v_j , if v_i is down to v_j , then a_{ij}^4 is 0. In Fig.5-1, $a_{ij}^1 = x_i + x_j$, $a_{ij}^2 = y_i + y_j$, $a_{ij}^3 = x_j - x_i$, $a_{ij}^4 = 0$. The semantic is not used.

Using the above definitions, the constructed ARG is a completed relation graph. This is very important for the next two ARG matching process.

6.3 Construct Decision Tree

Assume that two ARG's are $G = (N_{pattern}, A_{pattern}, \xi, \zeta)$ and $H = (N_{base}, A_{base}, \xi, \zeta)$, respectively, and G is a pattern graph with order m , H is the base graph with order n ($m \leq n$). In the special case where $m = n$, the problem is to find the optimal isomorphism between G and H . To achieve this goal, the decision tree is constructed first. It has height m , and $n - p$ sons for each node at level $p = 0, 1, \dots, m - 1$. Therefore, at any level $p > 0$, there is a path that consists of ordinal nodes in the decision tree from the root to node N , where

$$N = \{(i, q_i) | i = 1, 2, \dots, p\} \quad (1)$$

and from node N to a leaf node

$$T = \{(i, q_i) | i = 1, 2, \dots, m\} \quad (2)$$

Using the branch-and-bound technique to search through the decision tree, we can identify the optimal monomorphism between the pattern ARG and the base ARG. Then the document classification is transferred to the problem of how to match two

ARG's.

6.4 ARG matching Use Branch-and-Bound Technique

From expression (1) and (2), the set of the pattern ARG and base ARG is correspondingly divided into two parts, i.e.

$$N_{pattern} = N_1 \cup N_2 = \{1, 2, \dots, p\} \cup \{q_1, q_2, \dots, q_p\}$$

$$N_{base} = M_1 \cup M_2 = \{p+1, p+2, \dots, m\} \cup \{q \mid q \in \{1, 2, \dots, n\} \setminus N_2\}$$

The branch-and-bound technique used to solve the problem requires an evaluation function that assigns a cost to the branch incident to each node N of the tree. Assume that $N = \{(1, q_1), (2, q_2), \dots, (p, q_p)\}$ indicates the unique path from the root to node N , then the cost $k(p, q_p)$ assigned to the branch incident to N is defined as

$$k(p, q_p) = c(p, q_p) * w(p, q_p) + \sum_{j=1}^{p-1} \{c'((j, p), (q_j, q_p)) * w'((j, p), (q_j, q_p))\}$$

where $c'((a, b), (c, d)) = c((a, b), (c, d)) + c((b, a), (d, c))$, $w'((a, b), (c, d)) = w((a, b), (c, d)) + w((b, a), (d, c))$, $0 < w \leq 1$. The $c(x, y)$, and $w(x, y)$ is the mapping cost between the graph elements x and y in G and H , and the mapping weight between the graph elements x and y in G and H , respectively.

Referring to [71], the evaluation function f^* is defined as the value of $f^*(N)$ at a node $N = \{(i, q_i) \mid i = 1, 2, \dots, p\}$, which is the cost $g^*(N)$ of an optimal path from the root to node N plus the cost $h^*(N)$ of an optimal path from node N to a leaf $T = \{(i, q_i) \mid i = 1, 2, \dots, m\}$, i.e.

$$f^*(N) = g^*(N) + h^*(N)$$

$$g^*(N) = \sum_{i=1}^p k(i, q_i)$$

$$h^*(N) = \min_t \left\{ \sum_{i=p+1}^m k(i, q_{it}) \right\}$$

where t denotes a feasible path from N to T .

If the consistent lower bounded estimate value $h(N) \leq h^*(N)$, then use $h(N)$ replace the $h^*(N)$ for each node N , i.e.

$$f^*(N) = g^*(N) + h(N)$$

Let $k'(i, q)$ be the cost of adding a pair of vertices (i, q) to N , where $i \in M_1$ and $q \in M_2$.

Define

$$k'(i, q) = c(i, q) * w(i, q) + \sum_{j=1}^p \{c'((i, j), (q, q_j)) * w'((i, j), (q, q_j))\}$$

for each unmatched vertex $i \in M_1$, we find a corresponding vertex $q \in M_2$ such that the cost $k'(i, q)$ is minimized. Note that such a mapping $H: M_1 \rightarrow M_2$ may be many-to-one. Let $a(N)$ be the total cost of H . We have

$$a(N) = \sum_{i=p+1}^m \min_{q \in M_2} k'(i, q)$$

Similarly, $b(N)$ can be defined as the total cost of an optimal mapping $H': M_1 \times M_2 \rightarrow M_1 \times M_2$. i.e., for each arc (i, j) with endpoints $i, j \in M_1$, a minimum cost mapping $H': (i, j) \rightarrow (q, r)$ is found, such that (q, r) is an arc with endpoints $q, r \in M_2$. H' can also be many-to-one. Hence,

$$b(N) = \sum_{\substack{ij \in M_1 \\ i < j}} \min_{\substack{q \neq r \\ q, r \in M_2}} \{c'((i, j), (q, r)) * w'((i, j), (q, r))\}$$

With $a(N)$ and $b(N)$ defined, their summation is used as the lower bounded estimate of $h^*(N)$, i.e.,

$$h(N) = a(N) + b(N)$$

Therefore, $f(N) = g^*(N) + a(N) + b(N)$.

6.5 Application Algorithm for Matching ARG

From the definition above, let $c(x, y) = \|v_x - v_y\|$, $c'((a, b), (c, d)) = \|r_{ab} - r_{cd}\| + \|r_{ba} - r_{dc}\|$, and use Minkovski to express the distance between the samples, i.e.,

$$d_{ij}(q) = \left[\sum_{k=1}^p |x_{ik} - x_{jk}|^q \right]^{1/q}, q > 0$$

The following algorithm can be used for computing the optimal cost and optimal path.

Initiation

For level p from 1 to m do

{

For loop times from 1 to n-p+1 do

{

Set the set $N_1 = \{1, 2, \dots, p\}$.

Set the set $N_2 = \{\text{surplus nodes}\}$.

Set the set $M_1 = \{p+1, \dots, m\}$.

Set the set $M_2 = \{N_{base} \setminus N_2\}$.

Compute $f(N) = g^*(N) + a(N) + b(N)$.

Record the $f(N)$

}

To find the minimum $f(N)$, then assign the matching nodes.

To find the surplus nodes.

}

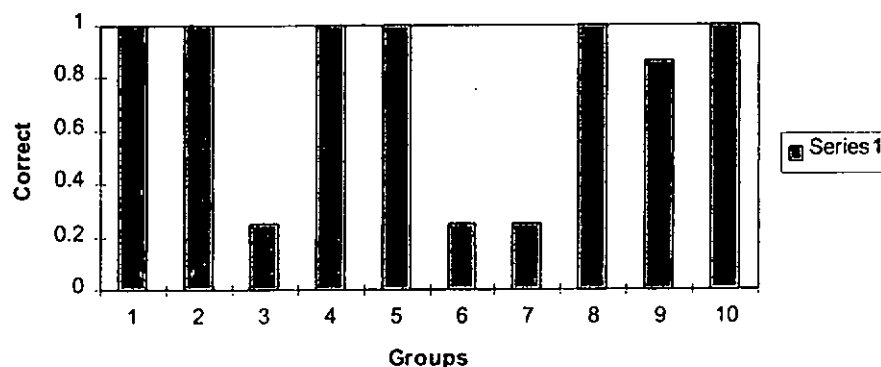
Output the optimal solution and path.

6.6 Experimental Results

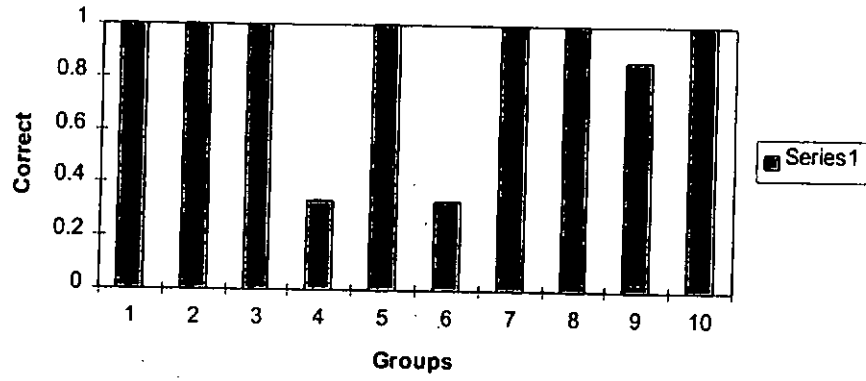
The proposed method for solving the document image classification by layout analysis has been implemented in a system also called DIC. The system has been trained with ten group/class documents and each group/class has six documents (the group 9, and 10 have 10 samples) which selected from different representative documents, and tested with different documents belong to or don't belong to the five group/class. The results are satisfied that DIC can depart the tested document into the group/class that it belongs to.

The testing divided into five parts. In the first part, the samples 1 and 3 of each group are selected as training, the 2, 4, 5, 6 as testing, and the results noted as Rts13. In the second part, the samples 2, 4 and 5 is selected as training, the 1, 3, and 6 as testing, and the results noted as Rts245, and so on. The all cases of the correction percent are shown in the figure. The statistics table and the analysis for incorrect matching are also enclosed.

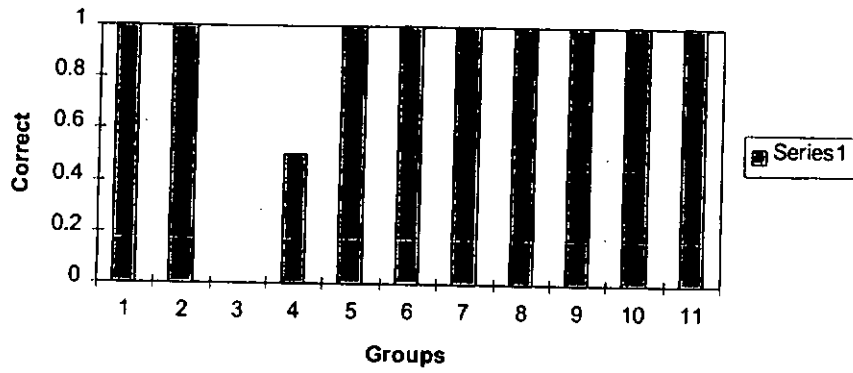
The Correct Percent of Rts13



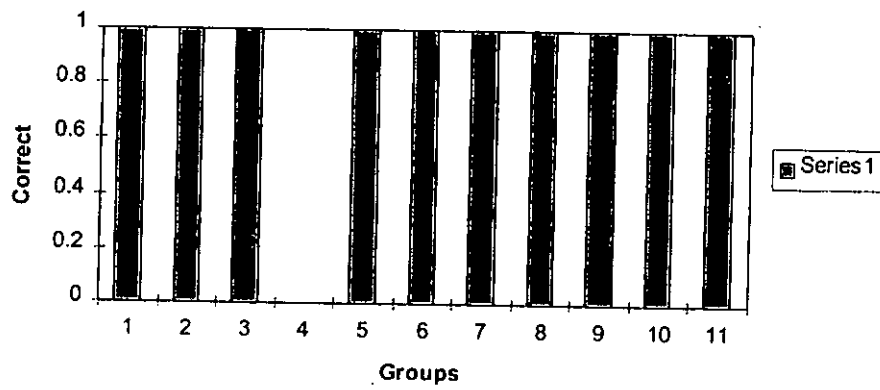
The Correct Percent of Rts245



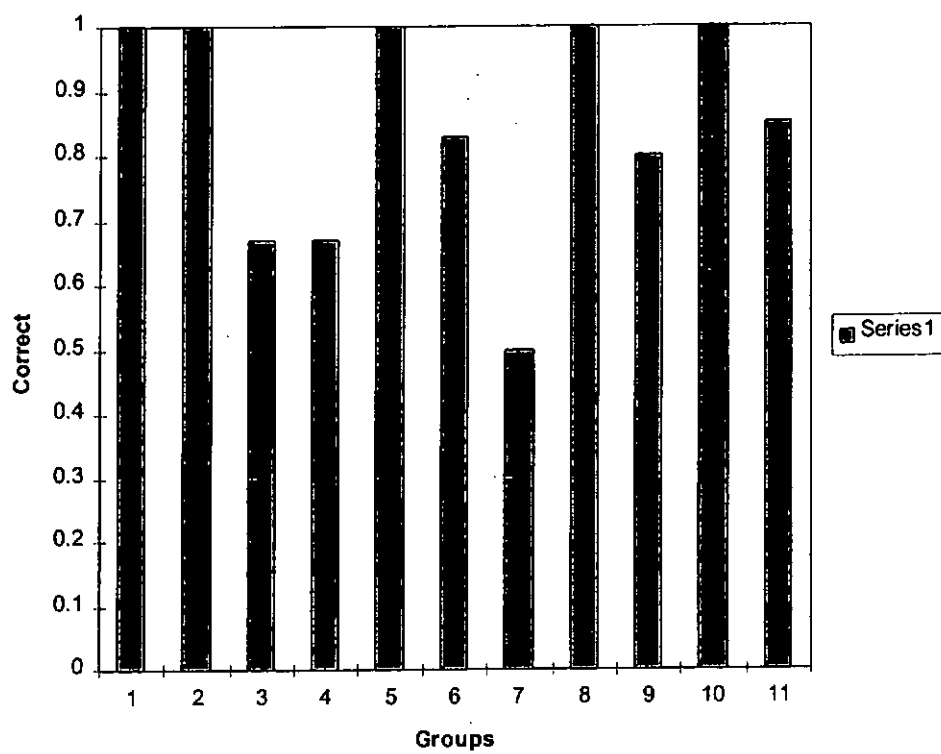
The Correct Percent of Rts1356



The Correct Percent of Rts12456



The Correct Percent of All Samples



The statistics of the document classification using DIC

Smp. Class	1, 3		2,4, 5		1,3, 5,6		1,2,4, 5,6	
	Cort.	Error	Cort.	Error	Cort.	Error	Cort.	Error
1	100%		100%		100%		100%	
2	100%		100%		100%		100%	
3	25%	75%	100%		0%	100%	100%	
4	100%		33%	67%	50%	50%	0%	100%
5	100%		100%		100%		100%	
6	25%	75%	33%	67%	100%		100%	
7	25%	75%	100%		100%		100%	
8	100%		100%		100%		100%	
9	86%	12%	86%	12%	100%		100%	
10	100%		100%		100%		100%	
11								
12								

Analysis for the Incorrect Matching

Groups	Training Groups	Testing Groups	Error Groups	Cause
Rts13	Group 1: h1_1.bmp,, h1_6.bmp Group 3: h3_1.bmp,, h3_6.bmp	h3_2.bmp	10	The pixels in h3_2.bmp is more, and the numbers of blocks is close.
		h7_2.bmp	10	The structure of h7_2.bmp and group 10 (h10_1.bmp, ..., h10_10.bmp) is very like. The distribution is also same.
		h7_4.bmp	10	The structure of h7_4.bmp and group 10 (h10_1.bmp, ..., h10_10.bmp) is very like. The distribution is also same.
		h7_6.bmp	10	The structure of h7_6.bmp and group 10 (h10_1.bmp, ..., h10_10.bmp) is very like. The distribution is also same.
		h11_5.bmp	7	The number of segmented blocks is close, and the distribution is near.
Rts245	Group 2: h2_1.bmp,, h2_6.bmp Group 4: h4_1.bmp,, h4_6.bmp Group 5: h5_1.bmp,, h5_6.bmp	h4_3.bmp	3	The pattern sample h3_5.bmp's distribution is like h4_3.bmp
		h7_1.bmp	3	The h7_1.bmp and h3_2.bmp have almost same structure.
		h11_10.bmp	7	This is like Rts13's h7_2.bmp comparing with group 10
Rts1356	Group 1: h1_1.bmp,, h1_6.bmp Group 3: h3_1.bmp,, h3_6.bmp Group 5: h5_1.bmp,, h5_6.bmp h6_1.bmp	h3_2.bmp	7	The structure of h3_2.bmp is very like as h7_5.bmp and h7_6.bmp

		h4_2.bmp	3	The pixels' distribution is more close.
Rts12456	<p><i>Group 1:</i> h1_1.bmp, h1_6.bmp</p> <p><i>Group 2:</i> h2_1.bmp, h2_6.bmp</p> <p><i>Group 4:</i> h4_1.bmp, h4_6.bmp</p> <p><i>Group 5:</i> h5_1.bmp, h5_6.bmp</p> <p><i>Group 6:</i> h6_1.bmp, h6_6.bmp</p>	h4_3.bmp	3	This is like Rts245's h4_3.bmp comparing with group 3
RtsAll	<i>All groups</i>	h3_2.bmp	7	This is like Rts1356's h3_2.bmp comparing with group 7
		h3_4.bmp	4	This is like Rts245's h4_3.bmp comparing with group 3
		h4_2.bmp	3	This is like Rts245's h4_3.bmp comparing with group 3
		h4_3.bmp	3	This is like Rts245's h4_3.bmp comparing with group 3
		h7_1.bmp	3	This is like Rts1356's h3_2.bmp comparing with group 7.
		h9_2.bmp	10	The number of pixel and the structure is very close.
		h11_5.bmp	7	This is like Rts13's h11_5.bmp comparing with group 7.
		h11_10.bmp	7	This is like Rts245's h11_10.bmp comparing with group 7.

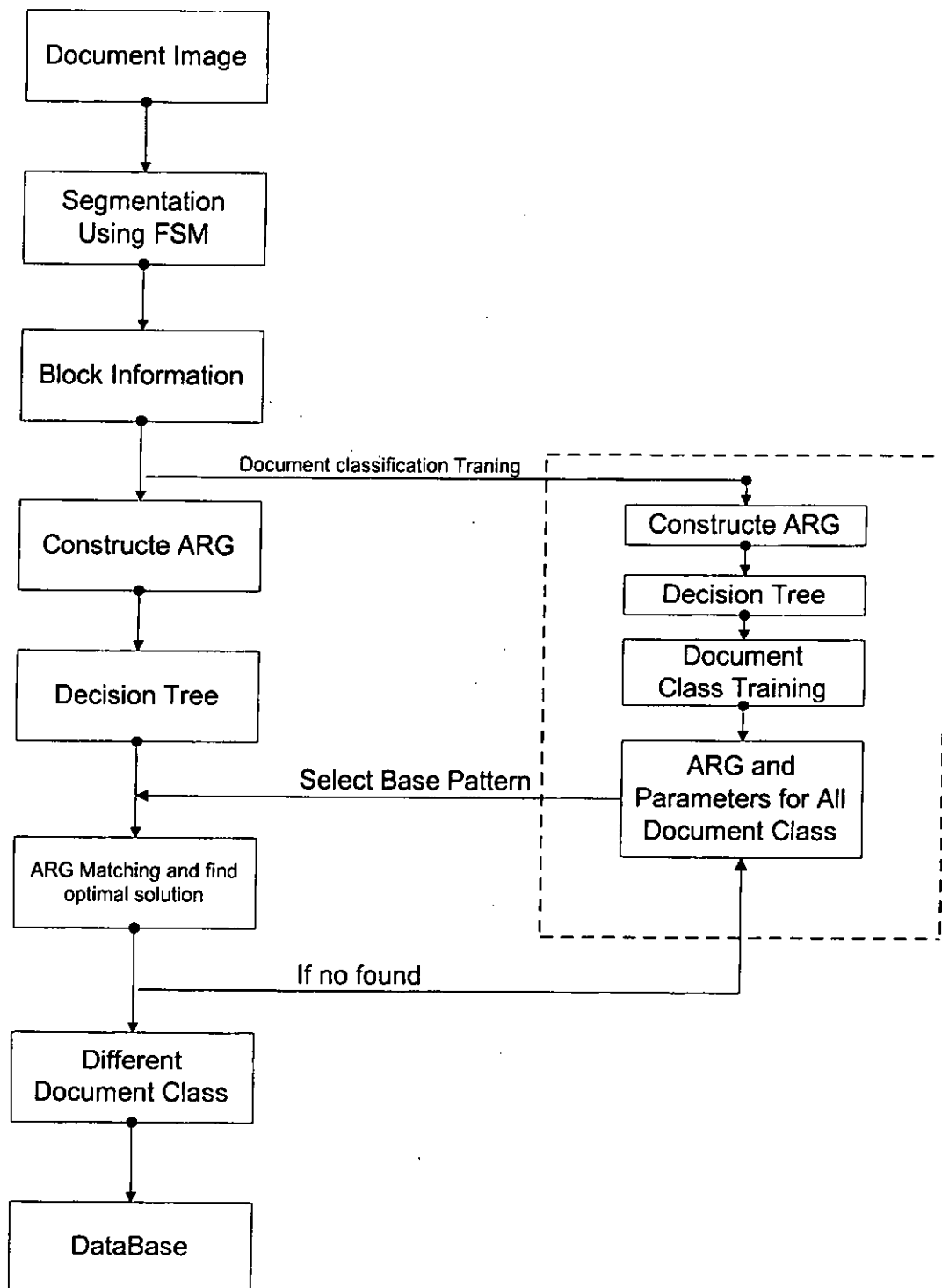


Fig.6.1 An overall organization of the proposed document processing system

Chapter 7 Conclusion

The main objective of this research work is to investigate the possibility of document classification. Document segmentation, block classification, and document classification, mainly three parts, was investigated enough and presents my concept and ideal. In the practice, we seek for new theory and methods to solve encounter new questions.

7.1 Summary

An *introduction* of my research is presented in chapter 1. In this chapter, why we need to develop our system for document classification, why so many companies research for it, what aim and performance we are requested, all this was depicted. The overview of the result is details in three parts. i.e. segmentation, block classification and document classification. The fuzzy methodology using in the pattern recognition is depicted also.

In *Chapter 2*, the technique for pattern recognition is surveyed particularly. The exiting question and the resolving method such as statistical pattern recognition, structural pattern recognition, pattern representation and attributed random graph are presented.

From chapter 3 to 5, three parts were deliberated in detail. For the segmentation, a method called the Fuzzy Segmentation Method for the analysis of document images was developed. Based on the use of the fuzzy set theory, we define fuzzy blank and fuzzy black blocks. The ideas of λ -cut sets are utilized when determining the vertical and horizontal thresholds in the segmentation process. With the definition of fuzzy sets, such a process can be made automated in an adaptive process. For the block classification, the four methods were analyzed. The shortcoming of them is shown. Then APACS method is applied to the block classification. The algorithm, features, and how to select the samples is specified. For the document classification, the document classification using Branch-and-Bound Technique based on ARG was depicted mainly. ARG construction, decision tree and using Branch-and-Bound to match two ARGs are presented in detail. The document classification is successfully transferred to how to match the two ARGs. In *chapter 5*, the whole procession of the document image classification was constructed. The experimental results were analyzed.

In *chapter 6*, to overcome the shortcoming of the system DIC and to prompt the capability, we present the modified algorithm for the two ARG matching. The experimental results are shown precisely.

7.2 Future Work

In a word, the capability, function and performance of the system DIC are satisfied. But those can be advanced further. For each stages of the document procession can still be sought for a new idea or methodology. There are lots of works waiting for doing.

(1) For the segmentation, the Run Length Smoothing Algorithm (RLSA) [1], and the Recursive X-Y Cuts algorithm (RXYC) [2] can be adopted for the fixed threshold in some special document procession. In this case, these two algorithms are more popular and practical. If the threshold is too difficulty to confirm, the FSM can be used for solution those problems. However, how to prompt the performance of FSM, and how to use it effectively, we can do more work in this area or can find much effectively method or algorithm.

(2) For the block classification, the methods presented in [1][3][12][21] all use the experienced parameter as criteria standard to affirm class to which the block belongs. It is important to select the parameter according to different situation. If we need accurately declare the feature of the each block, we must need lots of the sample and we must use the accordant criteria standard. How to select a criteria standard is very important. The criteria standard is usually constructed by the expert. Even if we can efficiently make the criteria standard, can accurately affirm the feature of each blocks and have enough samples for training, there are still some problem for the document classification mentioned in the chapter 5. Therefore, in our DIC system, we omit the part of affirming the feature of the blocks. Document classification algorithm employs the features of the blocks as parameters. This can reduce the error. In a word, this is only try for it. But the result is more satisfied.

(3) For the document classification algorithm, many papers such as [7][8][9] on the attributed random graph, and some other papers for example [71][73][74][77] on the graph matching had surveyed. The Branch-and-Bound Technique is adopted to match two ARG's of the documents after comparison. A new evaluation function of the Branch-and-Bound algorithm [71] is developed. But if the node of ARG is very big, according to the testing, if the node is big than 68, then the performance of running is

lower. In respect that there are $(2n)!$ combination of the nodes of base ARG and the nodes of pattern ARG orderings, if the DIC system has n nodes of base ARG with n nodes of pattern ARG. And if it has k non-zeros evaluation functions in the system, it will need $O(k)$ evaluation functions, giving it $O(k!)$ combinations there. All in all, there are $O((2n)!/k!)$ possible combinations. For the 56×56 system with roughly 12% non-zeros, this amounts to a number of combinations in the order of 10^{185} . The new idea and new methods for two attributed random graph matching need to seek further.

(4) Based on the document classification, the information in the document can be excerpted and saved into the database according to prescient document form. Much research can be done in this area.

Bibliography

- [1] Wong, K.Y., R. G. Casey, and F. M. Wahl, "Document analysis system," *IBM J. Res. Develop.* 26, No. 6, 1982, pp. 647-656.
- [2] Nagy, G., S. C. Seth, and S. D. Stoddard, "Document analysis with an expert system," *Proceedings, Pattern Recognition in Practice II*, Amsterdam, June 19-21, 1985.
- [3] Wang, D., Srihari, S.N., "Classification of Newspaper Image Block Using Texture Analysis," *Computer Vision, Graphics, Image Processing*, vol. 47, pp. 327-352, 1989.
- [4] Shih, C-C., and R. Kasturi, "Generation of a Line-Description File for Graphics Recognition," *Proc. SPIE Conf. Applications of Artificial Intelligence*, Vol. 937, SPIE, Bellingham, Wash., 1988, pp. 568-575.
- [5] T. Watanabe, Q. Luo, and N. Sugie, "Layout Recognition of Multi-Kinds of Table-Form Documents," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 4, April 1995, pp. 432-445.
- [6] Chiang C.-C., Cheng T., and Yu S.-S., "An Iterative Rule-Based Character Segmentation Method for Chinese Document," *International Conference on Chinese Computing '96*, June 4-7 1996, Singapore, pp. 301-307.
- [7] Tsai, W.-H., and K.-S. Fu, "Error-Correcting Isomorphisms of Attributed Relational Graphs for Pattern Analysis," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-9, No. 12, December 1979, pp. 757-768.
- [8] Sanfeliu, A., and K.-S. Fu, "A Distance Measure Between Attributed Relational Graphs for Pattern Recognition," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-13, No. 3, May/June 1983, pp. 353-362.
- [9] Wong, A.K.C., and M. You, "Entropy and Distance of Random Graphs with Application to Structural Pattern Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 5, Sept. 1985, pp. 599-609.
- [10] Liu, J., W.K. Cham, and M.M.Y. Chang, "Online Chinese character recognition using attributed relational graph matching," *IEE Proceeding: Vision, Image and Signal Processing*, V143 n2 Apr. 1996, pp. 125-131.
- [11] Caillol, H., A. Hillion, and W. Pieczynski, "Fuzzy Random Fields and Unsupervised Image Segmentation," *IEEE Trans. on Geoscience and Remote*

- Sensing*, Vol. 31, No. 4, July 1993, pp. 801-810.
- [12] Fisher, J.L., S.C. Hinds and D. P. D' Amato, "A rule-based system for document image segmentation." *Proc. IEEE 10th Int. Conf. Pattern Recognition*, Atlantic City, NJ, pp.567-572, June 1990.
- [13] Fletcher, L. A., and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Trans. Pattern Anal, Machine Intell.*, vol. 10, no. 6 pp. 910-918, Nov. 1988.
- [14] Wahl, F. M., K. Y. Wong and R. G. Casey, "Block segmentation and text extraction in mixed text/image documents," *Computer Graphics, Image Processing*, vol. 20, pp.375-390, 1982.
- [15] Bley, H., "Segmentation and preprocessing of electrical schematics using picture graphs," *Computer Vision, Graphics, Image Processing*, vol. 28, pp. 271-288, 1984.
- [16] Srihari, S. N., and G. W. Zack, "Document Image Analysis", *Proc. Inter'l Conf. On Pat. Recog.* 1986, pp.434-439.
- [17] Srihari, S. N., "Document Image Understanding," *Proceedings of ACM-IEEE Computer Society 1986 Fall Joint Computer Conference*, Dallas, TX, Nov., 1986.
- [18] Masude, I., N. Hagita, and T. Akiyama, "Approach to Smart Document Reader System," *Iproc. Conf. CVPR*, San Francisco, 1985, pp. 550-557.
- [19] Baird, H.S., "The Skew Angle of Printed Documents," *Proc. SPSE 40th Conf. And Symp. On Hybrid Imaging Sys.*, Rochester, NY, May, 1987, pp. 21-24.
- [20] Casey, R.G., and K.Y. Wong, "Survey of Document Analysis Systems and Techniques," *Proceedings of the ITL Conference on Pattern Recognition and Image Processing*, Toronto, Canada, Aug., 1987, pp. 289-911.
- [21] Shih, F. Y., S.-S. Chen, D. C. F. Hung, and A. N. Peter, "A Document Segmentation, Classification and Recognition System," *ICSI '92. Proceedings of the Second International Conference on Systems Integration*, June, 1992, pp. 258-167
- [22] Postl, W., "Detection of Linear Oblique Structures and Skew Scan in Digitized Documents," *Proc. 8th Int'l Conf. Pattern Recognition (ICPR)*, IEEE CS Press, Los Alamitos, Calif., 1986, pp. 687-689.
- [23] Baird, H.S., "The Skew Angle of Printed Documents," *Proc. Conf. Photographic Scientists and Engineers*, SPIE, Bellingham, Wa., 1987, pp. 14-21.
- [24] Akiyama, T., and N. Hagita, "Automated Entry System for Printed Documents,"

- Pattern Recognition*, Vol. 23, No. 11, 1990, pp. 1141-1154.
- [25] Pavlidis, T., and J. Zhou, " Page Segmentaton by White Streams," *Proc. 1st Int'l Conf. Document Analysis and Recognition (ICDAR)*, Int'l Assoc. Pattern Recognition, 1991, pp. 945-953.
- [26] Hinds, S.C., J.L. Fisher, and D.P. D'Amato, " A Document Skew Detection Method Using Run-Length Encoding and the Hough Transform," *Proc. 10th Int'l Conf. Pattern Recognition (ICPR)*, IEEE CS Pres, Los Alamitos, Calif., 1990, pp. 464-468.
- [27] Nakano, Y. et al., "An Algorithm for the Skew Normalization of Document Images," *Proc. 10th Int'l Conf. Pattern Recognition (ICPR)*, IEEE CS Press, Los Alamitos, Calif., 1990, pp. 8-13.
- [28] Srihari, S.N., and V. Govindaraju, " Analysis of Textual Images Using the Hough Transform," *Machine Vision and Applications*, Vol. 2, 1989, pp. 141-153.
- [29] Nagy, G., and S. Seth, " Hierarchical Representation of Optically Scanned Documents," *Proc. 7th Int'l Conf. Pattern Recognition (ICPR)*, IEEE CS Press, Los Alamitos, Calif., 1984, pp. 347-349.
- [30] Nagy, G., S. Seth, and M. Viswanathan, " A Prototype Document Image Analysis System for Technical Journals," *Computer*, Vol. 25, No. 7, July 1992, pp. 10-22.
- [31] Nagy 1992b] Nagy, G., " Toward a Structured Document Image Utility," in *Structured Document Image Analysis*, H.S. Baird, H. Bunke, and K. Yamamoto, eds., Springer Verlag, Berlin, 1992, pp. 54-69.
- [32] Baird, H.S., and K. Thompson, " Reading Chess," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 12, No. 6, June 1990, pp. 552-559.
- [33] O'Gorman, L., " The Document Spectrum for Structural Page Layout Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 15, No. 11, 1993, pp. 1031-1044.
- [34] Chan, K.C.C., Vieth, J.O, and Wong, A.K.C., " Training of an Artificial Neural Network for Pattern Classification: An Efficient Approach Based on Residual Analysis," *Proceedings of the SPIE Conference on Applications of Artificial Neural Network*, Orlanda, Fl, 1991.
- [35] Fraser, S., " Paperless Office," *Bus. Equip. Dig.*, Feb., 1990, pp. 40-1.
- [36] Ker, N., " No More Waste Paper," *Image Process.*, Vol. 2, No. 1, Mar.-Apr., 1990, pp. 46-50.
- [37] Olding, R., " Using Document Image Processing to Eliminate Community

- Charge Paperwork," *OIS International 1989. Proceedings of the Sixth Annual Conference on Optical Information Systems*, London, UK, May, 1989, pp. 184-8.
- [38] Minnich, R.L., "Automating Paper Processing with Optical Disk in the Insurance Industry," *Reaping the Benefits of Technology. Proceedings: LOMA's Orlando, Fl.*, Mar., 1989, pp. 19-22.
- [39] Willimas, B., "DIP at the Home Office," *Inf. Media Technol.*, Vol. 23 No.1, Jan, 1990, pp.21-2.
- [40] Friedman, P., "Large Document Image Systems: Experience and Opportunities," in *OIS International 1989. Proceedings of the Sixth Annual Conference on Optical Information System*, London, UK, May, 1989, pp.15-7.
- [41] Remmer, D., "Deciding DIP Standards," *Image Process.* Vol. 1, No.1, Summer 1989, pp. 50-2.
- [42] Trammell, B., "Too Little Too Late? Not at USAA," *Inform*, Jul, -Aug., 1989, pp. 24-52.
- [43] "The Image of A Paperless Office," *Mind your Own Business*, Vol.12, No.9, Oct., 1989, pp. 15-6.
- [44] "Electronic Document Management: Part I," *IIS Analyzer*, Vol.27, No.5, May, 1989, pp.1-14.
- [45] Horten, M., "The Image Makers," *Bank. Technol.*, Apr. 1989, pp.24-8.
- [46] Richardson, S., "Finding the Needle Without Disturbing the Haystack," *Document Image Processing. Proceeding of the First European Conference*, London, UK, Mar. 1988, pp.41-5.
- [47] "Canada's Investors Group and Document-Image Processing," *IMC Journal*, Mar.-Apr., 1989, pp.24-6.
- [48] Macadam, R., "Document Processing: Operational Control vs Library Storage," *Document Image Processing. Proceedings of the First European Conference*, London, UK, Mar. 1988, pp.41-5.
- [49] Mawdsley, D., "Clearing a Filing Room Log-Jam," *Document Image Processing. Proceeding of the First European Conference*, London, UK, Mar. 1988, pp.41-5.
- [50] "File It in FileNet," *Insurance Software Review*, Apr.-May 1989, pp.8-16.
- [51] Wallace, L.G., "Image and Data Processing Working Together," *Optical Information Systems '88, Conference Proceedings*, Washington, DC, Sept., 1988, pp. 7-9.
- [52] Williams, T., "Document Processing Moves to the Desktop," *Computer Design*,

- July, 1989, pp.63-70.
- [53] Townsend, K., " Making Document Redundant," *Mind Your Own Bus.*, Vol.12, No.10, Nov., 1989, pp.74-5.
- [54] Brown, A.E., and Roderick, S.J., " A Document Image Management System Tames the Insurance Industry's Paper Tiger," *Optical Information systems*, Mar.-Apr., 1990, pp.79-83.
- [55] Springford, A., " Choosing A Document Image Processing System," *ROCC Inf. Manage.* Autumn 1990, pp.9-11.
- [56] Williams, B., " CMB: Document Image Processing in A Technical Information Services Environment," *Info. Media & Technol.*, Vol.22, No.6, Nov., 1989, pp.271-2.
- [57] Townsend, K., " Lmage Makers," *Comput. Syst. Eru.*, Vol.10, No.3, Mar. 1990, pp21-7.
- [58] Parker, C., " Creating the Right Image," *HP World*, Vol.3, No.3, Mar 1990, pp.38-42.
- [59] Sullivan, R.K., " Integrated Image Systems: A Year in Review," *Optical Information Systems '88, Conference Proceedings*, Washington, DC, Sept., 1988, pp.7-9.
- [60] " DIP Update: Mid Range to Large System," *Wharton Rep.*, No.133, Sept, 1989, pp. 1-10.
- [61] Esposito, F., Malerba, D., and Semeraro, G., " Empirical Learning Methods for Digitized Document Recognition: An Integrated Approach to Inductive Generation," *Proc. Conf. AI Appl.*, Santa Babara, CA, Mar. 1990, pp.37-45.
- [62] Chan, K.C.C., and Wong, A.K.C., " APACS: A System for Automated Pattern Analysis and Classification," *Comput. Intell.* Vol.6, 1990, pp.119-131.
- [63] Chan, K.C.C., and Wong, A.K.C., " A Statistical Technique for Extracting Classificatory Knowledge", G. Piatetsky-Shapiro and W. Frawley, (eds.), *Knowledge Discovery in Databases*, AAAI Press, 1990.
- [64] Chan, K.C.C., and Wong, A.K.C., " Performance Analysis of A Probabilistic Inductive Learning System," *Proceedings of the International Conference on Machine Learning*, Austin, Texas, Jun., 1990.
- [65] Chan, K.C.C., and Wong, A.K.C., " Automatic Construction of Expert System from Data: A Statistical Approach," *Proceedings of the IJCAI'89 Workshop on Knowledge Discovery in Databases*, Detroit, Michigan, Aug., 1989.

- [66] Chan, K.C.C., and Wong, A.K.C., " PIS: A Probabilistic Inference System," *Proceedings of the IAPR Eleventh International Conference on Pattern Recognition*, Rome, Italy, Nov., 1988.
- [67] Chan, K.C.C., " Inductive Learning in the Presence of Uncertainty," Ph.D., Dissertation, Department of Systems Design Engineering, University of Waterloo, Waterloo, Ontario, 1989.
- [68] Wong, A.K.C., and Chan, K.C.C., " Automating the Knowledge Acquisition Process in The Construction of Medical Expert Systems," *Arti. Intell. In Medi.*, Nov. 1990.
- [69] Dengel, A., and Barth, G., " High Level Document Analysis Guided by Geometric Aspects," *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.2, No.4, 1988, pp.641-655.
- [70] Keith C. C. Chan, Andrew K. C. Wong, and David K. Y. Chiu, "Learning Sequential Patterns for Probabilistic Inductive Prediction," *IEEE Trans. on Sys., Man, and Cybernetics*, Vol.24, No.10, pp.1532-1547, 1994.
- [71] A.K.C. Wong, M. You, and S.C. Chan, "An Algorithm for Graph Optimal Monomorphism," *IEEE Trans. on Sys., Man, and Cybernetics*, Vol.20, No.3, pp.628-636, May/June 1990.
- [72] Keith C.C. Chan, X. D. Huang, and P. Bao, "Fuzzy Segmentation for Document Image Analysis," *Proc. 1997 IEEE Int. Conf. On Systems Man and Cybernetics*, 1997.
- [73] David E. Ghahraman, Andrew K, C. Wong, and Tung Au, "Graph Optimal Monomorphism Algorithms," *IEEE Trans. on Sys., Man, and Cybernetics*, Vol. SMC-10, No.4, pp.181-188, April 1980.
- [74] David E. Ghahraman, Andrew K, C. Wong, and Tung Au, "Graph Monomorphism Algorithms," *IEEE Trans. on Sys., Man, and Cybernetics*, Vol. SMC-10, No.4, pp.189-196, April 1980.
- [75] K.P. Chan, and Y. S. Cheung, "Fuzzy-Attribute Graph with Application to Chinese Recognition," *IEEE Trans. on Sys. Man, and Cybernetics*, Vol.22, No.1, pp.153-160, Jan./Feb. 1992.
- [76] L.G. Shapiro, and R.M.Haralick,"Structural Descriptions and Inexact Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.3, pp.504-519, Sept. 1981.
- [77] W.H. Tsai and K.S. Fu, "Subgraph error-correcting isomorphism for syntactic

pattern recognition," *IEEE Trans. Sys. Man Cybernetics*, Vol. SMC-13, No.1, pp.48-62, Jan./Feb. 1983.

Appendix A

Shell Commands for Testing and Training in UNIX

```
// TrnTst
#####
# 1. Training the first group of samples
dic h1_1.bmp 1
dic h1_3.bmp 1
cp BGraph.txt BG13
#
dic h2_1.bmp 2
dic h2_3.bmp 2
cp BGraph.txt BG13
#
dic h3_1.bmp 3
dic h3_3.bmp 3
cp BGraph.txt BG13
#
dic h4_1.bmp 4
dic h4_3.bmp 4
cp BGraph.txt BG13
#
dic h5_1.bmp 5
dic h5_3.bmp 5
cp BGraph.txt BG13
#
dic h6_1.bmp 6
dic h6_3.bmp 6
cp BGraph.txt BG13
#
dic h7_1.bmp 7
dic h7_3.bmp 7
cp BGraph.txt BG13
#
dic h8_1.bmp 8
dic h8_3.bmp 8
cp BGraph.txt BG13
#
dic h9_1.bmp 9
dic h9_3.bmp 9
cp BGraph.txt BG13
#
dic h10_1.bmp 10
dic h10_3.bmp 10
cp BGraph.txt BG13
#
dic h11_1.bmp 11
dic h11_3.bmp 11
cp BGraph.txt BG13
#
dic h12_1.bmp 12
dic h12_3.bmp 12
cp BGraph.txt BG13
# 1. Testing the first group of samples
dic h1_2.bmp
dic h1_4.bmp
dic h1_5.bmp
dic h1_6.bmp
cp Results.txt .Rts13.txt
#
dic h2_2.bmp
dic h2_4.bmp
dic h2_5.bmp
dic h2_6.bmp
cp Results.txt .Rts13.txt
#
dic h3_2.bmp
dic h3_4.bmp
dic h3_5.bmp
dic h3_6.bmp
cp Results.txt .Rts13.txt
#
dic h4_2.bmp
dic h4_4.bmp
dic h4_5.bmp
dic h4_6.bmp
cp Results.txt .Rts13.txt
#
dic h5_2.bmp
dic h5_4.bmp
dic h5_5.bmp
dic h5_6.bmp
cp Results.txt .Rts13.txt
#
dic h6_2.bmp
dic h6_4.bmp
dic h6_5.bmp
dic h6_6.bmp
cp Results.txt .Rts13.txt
#
dic h7_2.bmp
dic h7_4.bmp
dic h7_5.bmp
dic h7_6.bmp
cp Results.txt .Rts13.txt
#
dic h8_2.bmp
dic h8_4.bmp
dic h8_5.bmp
dic h8_6.bmp
cp Results.txt .Rts13.txt
#
dic h9_2.bmp
dic h9_4.bmp
dic h9_5.bmp
dic h9_6.bmp
cp Results.txt .Rts13.txt
#
dic h10_2.bmp
dic h10_4.bmp
dic h10_5.bmp
dic h10_6.bmp
cp Results.txt .Rts13.txt
#
dic h11_2.bmp
dic h11_4.bmp
dic h11_5.bmp
dic h11_6.bmp
dic h11_7.bmp
dic h11_8.bmp
dic h11_9.bmp
dic h11_10.bmp
cp Results.txt .Rts13.txt
#
dic h12_2.bmp
dic h12_4.bmp
dic h12_5.bmp
```



```

dic h12_6.bmp
dic h12_7.bmp
dic h12_8.bmp
dic h12_9.bmp
dic h12_10.bmp
cp Results.txt .Rts13.txt
#
rm core
rm BGraph.txt
rm Results.txt
#
#####
# 2. Training the first group of samples
dic h1_2.bmp 1
dic h1_4.bmp 1
dic h1_5.bmp 1
cp BGraph.txt BG245
#
dic h2_2.bmp 2
dic h2_4.bmp 2
dic h2_5.bmp 2
cp BGraph.txt BG245
#
dic h3_2.bmp 3
dic h3_4.bmp 3
dic h3_5.bmp 3
cp BGraph.txt BG245
#
dic h4_2.bmp 4
dic h4_4.bmp 4
dic h4_6.bmp 4
cp BGraph.txt BG245
#
dic h5_2.bmp 5
dic h5_4.bmp 5
dic h5_5.bmp 5
cp BGraph.txt BG245
#
dic h6_2.bmp 6
dic h6_4.bmp 6
dic h6_5.bmp 6
cp BGraph.txt BG245
#
dic h7_2.bmp 7
dic h7_4.bmp 7
dic h7_5.bmp 7
cp BGraph.txt BG245
#
dic h8_2.bmp 8
dic h8_4.bmp 8
dic h8_5.bmp 8
cp BGraph.txt BG245
#
dic h9_2.bmp 9
dic h9_4.bmp 9
dic h9_5.bmp 9
cp BGraph.txt BG245
#
dic h10_2.bmp 10
dic h10_4.bmp 10
dic h10_5.bmp 10
cp BGraph.txt BG245
#
dic h11_2.bmp 11
dic h11_4.bmp 11
dic h11_5.bmp 11
cp BGraph.txt BG245
#
dic h12_2.bmp 12
dic h12_4.bmp 12
dic h12_5.bmp 12
cp BGraph.txt BG245
# 2. Testing the first group of samples
dic h1_1.bmp
dic h1_3.bmp
dic h1_6.bmp
cp Results.txt .Rts245.txt
#
dic h2_1.bmp
dic h2_3.bmp
dic h2_6.bmp
cp Results.txt .Rts245.txt
#
dic h3_1.bmp
dic h3_3.bmp
dic h3_6.bmp
cp Results.txt .Rts245.txt
#
dic h4_1.bmp
dic h4_3.bmp
dic h4_6.bmp
cp Results.txt .Rts245.txt
#
dic h5_1.bmp
dic h5_3.bmp
dic h5_6.bmp
cp Results.txt .Rts245.txt
#
dic h6_1.bmp
dic h6_3.bmp
dic h6_6.bmp
cp Results.txt .Rts245.txt
#
dic h7_1.bmp
dic h7_3.bmp
dic h7_6.bmp
cp Results.txt .Rts245.txt
#
dic h8_1.bmp
dic h8_3.bmp
dic h8_6.bmp
cp Results.txt .Rts245.txt
#
dic h9_1.bmp
dic h9_3.bmp
dic h9_6.bmp
cp Results.txt .Rts245.txt
#
dic h10_1.bmp
dic h10_3.bmp
dic h10_6.bmp
cp Results.txt .Rts245.txt
#
dic h11_1.bmp
dic h11_3.bmp
dic h11_6.bmp
dic h11_7.bmp
cp Results.txt .Rts245.txt
dic h11_8.bmp
dic h11_9.bmp
dic h11_10.bmp
cp Results.txt .Rts245.txt
#
dic h12_1.bmp
dic h12_3.bmp
dic h12_6.bmp
dic h12_7.bmp
cp Results.txt .Rts245.txt
dic h12_8.bmp
dic h12_9.bmp
dic h12_10.bmp
cp Results.txt .Rts245.txt
#
rm core
rm BGraph.txt
rm Results.txt

```

```

%
#####
# 3. Training the first group of samples
dic h1_1.bmp 1
dic h1_3.bmp 1
cp BGraph.txt BG1356
dic h1_5.bmp 1
dic h1_6.bmp 1
cp BGraph.txt BG1356
#
dic h2_1.bmp 2
dic h2_3.bmp 2
cp BGraph.txt BG1356
dic h2_5.bmp 2
dic h2_6.bmp 2
cp BGraph.txt BG1356
#
dic h3_1.bmp 3
dic h3_3.bmp 3
cp BGraph.txt BG1356
dic h3_5.bmp 3
dic h3_6.bmp 3
cp BGraph.txt BG1356
#
dic h4_1.bmp 4
dic h4_3.bmp 4
cp BGraph.txt BG1356
dic h4_5.bmp 4
dic h4_6.bmp 4
cp BGraph.txt BG1356
#
dic h5_1.bmp 5
dic h5_3.bmp 5
cp BGraph.txt BG1356
dic h5_5.bmp 5
dic h5_6.bmp 5
cp BGraph.txt BG1356
#
dic h6_1.bmp 6
dic h6_3.bmp 6
cp BGraph.txt BG1356
dic h6_5.bmp 6
dic h6_6.bmp 6
cp BGraph.txt BG1356
#
dic h7_1.bmp 7
dic h7_3.bmp 7
cp BGraph.txt BG1356
dic h7_5.bmp 7
dic h7_6.bmp 7
cp BGraph.txt BG1356
#
dic h8_1.bmp 8
dic h8_3.bmp 8
cp BGraph.txt BG1356
dic h8_5.bmp 8
dic h8_6.bmp 8
cp BGraph.txt BG1356
#
dic h9_1.bmp 9
dic h9_3.bmp 9
cp BGraph.txt BG1356
dic h9_5.bmp 9
dic h9_6.bmp 9
cp BGraph.txt BG1356
#
dic h10_1.bmp 10
dic h10_3.bmp 10
cp BGraph.txt BG1356
dic h10_5.bmp 10
dic h10_6.bmp 10
cp BGraph.txt BG1356
#
dic h11_1.bmp 11
dic h11_3.bmp 11
dic h11_5.bmp 11
dic h11_6.bmp 11
cp BGraph.txt BG1356
dic h11_7.bmp 11
dic h11_8.bmp 11
dic h11_9.bmp 11
dic h11_10.bmp 11
cp BGraph.txt BG1356
#
dic h12_1.bmp 12
dic h12_3.bmp 12
dic h12_5.bmp 12
dic h12_6.bmp 12
cp BGraph.txt BG1356
dic h12_7.bmp 12
dic h12_8.bmp 12
dic h12_9.bmp 12
dic h12_10.bmp 12
cp BGraph.txt BG1356
# 3. Testting the first group of samples
dic h1_2.bmp
dic h1_4.bmp
cp Results.txt .Rts1356.txt
#
dic h2_2.bmp
dic h2_4.bmp
cp Results.txt .Rts1356.txt
#
dic h3_2.bmp
dic h3_4.bmp
cp Results.txt .Rts1356.txt
#
dic h4_2.bmp
dic h4_4.bmp
cp Results.txt .Rts1356.txt
#
dic h5_2.bmp
dic h5_4.bmp
cp Results.txt .Rts1356.txt
#
dic h6_2.bmp
dic h6_4.bmp
cp Results.txt .Rts1356.txt
#
dic h7_2.bmp
dic h7_4.bmp
cp Results.txt .Rts1356.txt
#
dic h8_2.bmp
dic h8_4.bmp
cp Results.txt .Rts1356.txt
#
dic h9_2.bmp
dic h9_4.bmp
cp Results.txt .Rts1356.txt
#
dic h10_2.bmp
dic h10_4.bmp
cp Results.txt .Rts1356.txt
#
dic h11_2.bmp
dic h11_4.bmp
cp Results.txt .Rts1356.txt
#
dic h12_2.bmp
dic h12_4.bmp
cp Results.txt .Rts1356.txt
#
rm core
rm BGraph.txt
rm Results.txt

```

```

#####
#4. Training the first group of samples
dic h1_1.bmp 1
dic h1_2.bmp 1
dic h1_4.bmp 1
cp BGraph.txt BG12456
dic h1_5.bmp 1
dic h1_6.bmp 1
cp BGraph.txt BG12456
#
dic h2_1.bmp 2
dic h2_2.bmp 2
cp BGraph.txt BG12456
dic h2_4.bmp 2
dic h2_5.bmp 2
dic h2_6.bmp 2
cp BGraph.txt BG12456
#
dic h3_1.bmp 3
dic h3_2.bmp 3
cp BGraph.txt BG12456
dic h3_4.bmp 3
dic h3_5.bmp 3
dic h3_6.bmp 3
cp BGraph.txt BG12456
#
dic h4_1.bmp 4
dic h4_2.bmp 4
cp BGraph.txt BG12456
dic h4_4.bmp 4
dic h4_5.bmp 4
dic h4_6.bmp 4
cp BGraph.txt BG12456
#
dic h5_1.bmp 5
dic h5_2.bmp 5
cp BGraph.txt BG12456
dic h5_4.bmp 5
dic h5_5.bmp 5
dic h5_6.bmp 5
cp BGraph.txt BG12456
#
dic h6_1.bmp 6
dic h6_2.bmp 6
cp BGraph.txt BG12456
dic h6_4.bmp 6
dic h6_5.bmp 6
dic h6_6.bmp 6
cp BGraph.txt BG12456
#
dic h7_1.bmp 7
dic h7_2.bmp 7
cp BGraph.txt BG12456
dic h7_4.bmp 7
dic h7_5.bmp 7
dic h7_6.bmp 7
cp BGraph.txt BG12456
#
dic h8_1.bmp 8
dic h8_2.bmp 8
cp BGraph.txt BG12456
dic h8_4.bmp 8
dic h8_5.bmp 8
dic h8_6.bmp 8
cp BGraph.txt BG12456
#
dic h9_1.bmp 9
dic h9_2.bmp 9
cp BGraph.txt BG12456
dic h9_4.bmp 9
dic h9_5.bmp 9
dic h9_6.bmp 9

cp BGraph.txt BG12456
#
dic h10_1.bmp 10
dic h10_2.bmp 10
cp BGraph.txt BG12456
dic h10_4.bmp 10
dic h10_5.bmp 10
dic h10_6.bmp 10
cp BGraph.txt BG12456
#
dic h11_1.bmp 11
dic h11_2.bmp 11
dic h11_4.bmp 11
cp BGraph.txt BG12456
dic h11_5.bmp 11
dic h11_6.bmp 11
dic h11_7.bmp 11
dic h11_8.bmp 11
dic h11_9.bmp 11
dic h11_10.bmp 11
cp BGraph.txt BG12456
#
dic h12_1.bmp 12
dic h12_2.bmp 12
dic h12_4.bmp 12
dic h12_5.bmp 12
cp BGraph.txt BG12456
dic h12_6.bmp 12
dic h12_7.bmp 12
dic h12_8.bmp 12
dic h12_9.bmp 12
dic h12_10.bmp 12
#cp BGraph.txt BG12456
#
# 4. Testting the first group of samples
dic h1_3.bmp
cp Results.txt .Rts12456.txt
#
dic h2_3.bmp
cp Results.txt .Rts12456.txt
#
dic h3_3.bmp
cp Results.txt .Rts12456.txt
#
dic h4_3.bmp
cp Results.txt .Rts12456.txt
#
dic h5_3.bmp
cp Results.txt .Rts12456.txt
#
dic h6_3.bmp
cp Results.txt .Rts12456.txt
#
dic h7_3.bmp
cp Results.txt .Rts12456.txt
#
dic h8_3.bmp
cp Results.txt .Rts12456.txt
#
dic h9_3.bmp
cp Results.txt .Rts12456.txt
#
dic h10_3.bmp
cp Results.txt .Rts12456.txt
#
dic h11_3.bmp
cp Results.txt .Rts12456.txt
#
dic h12_3.bmp
cp Results.txt .Rts12456.txt
#
rm core
rm boxed.bmp

```

```

#####
# 1. Training the first group of samples
dic h1_1.bmp 1
dic h1_2.bmp 1
dic h1_3.bmp 1
dic h1_4.bmp 1
dic h1_5.bmp 1
dic h1_6.bmp 1
cp BGraph.txt BGAll.txt
#
dic h2_1.bmp 2
dic h2_2.bmp 2
dic h2_3.bmp 2
dic h2_4.bmp 2
dic h2_5.bmp 2
dic h2_6.bmp 2
cp BGraph.txt BGAll.txt
#
dic h3_1.bmp 3
dic h3_2.bmp 3
dic h3_3.bmp 3
dic h3_4.bmp 3
dic h3_5.bmp 3
dic h3_6.bmp 3
cp BGraph.txt BGAll.txt
#
dic h4_1.bmp 4
dic h4_2.bmp 4
dic h4_3.bmp 4
dic h4_4.bmp 4
dic h4_5.bmp 4
dic h4_6.bmp 4
cp BGraph.txt BGAll.txt
#
dic h5_1.bmp 5
dic h5_2.bmp 5
dic h5_3.bmp 5
dic h5_4.bmp 5
dic h5_5.bmp 5
dic h5_6.bmp 5
cp BGraph.txt BGAll.txt
#
dic h6_1.bmp 6
dic h6_2.bmp 6
dic h6_3.bmp 6
dic h6_4.bmp 6
dic h6_5.bmp 6
dic h6_6.bmp 6
cp BGraph.txt BGAll.txt
#
dic h7_1.bmp 7
dic h7_2.bmp 7
dic h7_3.bmp 7
dic h7_4.bmp 7
dic h7_5.bmp 7
dic h7_6.bmp 7
cp BGraph.txt BGAll.txt
#
dic h8_1.bmp 8
dic h8_2.bmp 8
dic h8_3.bmp 8
dic h8_4.bmp 8
dic h8_5.bmp 8
dic h8_6.bmp 8
cp BGraph.txt BGAll.txt
#
dic h9_1.bmp 9
dic h9_2.bmp 9
dic h9_3.bmp 9
dic h9_4.bmp 9
dic h9_5.bmp 9

dic h9_6.bmp 9
cp BGraph.txt BGAll.txt
#
dic h10_1.bmp 10
dic h10_2.bmp 10
dic h10_3.bmp 10
dic h10_4.bmp 10
dic h10_5.bmp 10
dic h10_6.bmp 10
cp BGraph.txt BGAll.txt
#
dic h11_1.bmp 11
dic h11_2.bmp 11
dic h11_3.bmp 11
dic h11_4.bmp 11
dic h11_5.bmp 11
dic h11_6.bmp 11
cp BGraph.txt BGAll.txt
#
dic h12_1.bmp 12
dic h12_2.bmp 12
dic h12_3.bmp 12
dic h12_4.bmp 12
dic h12_5.bmp 12
dic h12_6.bmp 12
cp BGraph.txt BGAll.txt
# 1. Testing the first group of samples
dic h1_1.bmp
dic h1_2.bmp
dic h1_3.bmp
dic h1_4.bmp
dic h1_5.bmp
dic h1_6.bmp
cp Results.txt RtsAll.txt
#
dic h2_1.bmp
dic h2_2.bmp
dic h2_3.bmp
dic h2_4.bmp
dic h2_5.bmp
dic h2_6.bmp
cp Results.txt RtsAll.txt
#
dic h3_1.bmp
dic h3_2.bmp
dic h3_3.bmp
dic h3_4.bmp
dic h3_5.bmp
dic h3_6.bmp
cp Results.txt RtsAll.txt
#
dic h4_1.bmp
dic h4_2.bmp
dic h4_3.bmp
dic h4_4.bmp
dic h4_5.bmp
dic h4_6.bmp
cp Results.txt RtsAll.txt
#
dic h5_1.bmp
dic h5_2.bmp
dic h5_3.bmp
dic h5_4.bmp
dic h5_5.bmp
dic h5_6.bmp
cp Results.txt RtsAll.txt
#
dic h6_1.bmp
dic h6_2.bmp
dic h6_3.bmp
dic h6_4.bmp
dic h6_5.bmp
dic h6_6.bmp

```

Cp Results.txt RtsAll.txt

dic h7_1.bmp
dic h7_2.bmp
dic h7_3.bmp
dic h7_4.bmp
dic h7_5.bmp
dic h7_6.bmp
cp Results.txt RtsAll.txt
#

dic h8_1.bmp
dic h8_2.bmp
dic h8_3.bmp
dic h8_4.bmp
dic h8_5.bmp
dic h8_6.bmp
cp Results.txt RtsAll.txt
#

dic h9_1.bmp
dic h9_2.bmp
dic h9_3.bmp
dic h9_4.bmp
dic h9_5.bmp
dic h9_6.bmp
cp Results.txt RtsAll.txt
#

dic h10_1.bmp
dic h10_2.bmp
dic h10_3.bmp
dic h10_4.bmp
dic h10_5.bmp
dic h10_6.bmp
cp Results.txt RtsAll.txt
#

dic h11_1.bmp
dic h11_2.bmp
dic h11_3.bmp
dic h11_4.bmp
dic h11_5.bmp
dic h11_6.bmp
dic h11_7.bmp
dic h11_8.bmp
dic h11_9.bmp
dic h11_10.bmp
cp Results.txt RtsAll.txt
#

dic h12_1.bmp
dic h12_2.bmp
dic h12_3.bmp
dic h12_4.bmp
dic h12_5.bmp
dic h12_6.bmp
dic h12_7.bmp
dic h12_8.bmp
dic h12_9.bmp
dic h12_10.bmp
cp Results.txt RtsAll.txt
#

rm core
#rm BGraph.txt
#rm BGResults.txt

#####

Appendix B

All Combinations' Testing and Training Results

// Rts13		h9_2.bmp to 10th class, CostValue	1.37
-----		-----	
h1_2.bmp to 1th class, CostValue	0.56	h9_4.bmp to 10th class, CostValue	1.46
-----		-----	
h1_4.bmp to 1th class, CostValue	0.56	h9_5.bmp to 10th class, CostValue	1.52
-----		-----	
h1_5.bmp to 1th class, CostValue	0.60	h9_6.bmp to 9th class, CostValue	1.35
-----		-----	
h1_6.bmp to 1th class, CostValue	0.56	h10_2.bmp to 10th class, CostValue	1.34
-----		-----	
h2_2.bmp to 2th class, CostValue	0.06	h10_4.bmp to 10th class, CostValue	0.99
-----		-----	
h2_4.bmp to 2th class, CostValue	0.06	h10_5.bmp to 10th class, CostValue	0.73
-----		-----	
h2_5.bmp to 2th class, CostValue	0.04	h10_6.bmp to 10th class, CostValue	0.88
-----		-----	
h2_6.bmp to 2th class, CostValue	0.05	h11_2.bmp to 11th class, CostValue	0.33
-----		-----	
h3_2.bmp to 10th class, CostValue	1.43	h11_4.bmp to 11th class, CostValue	0.00
-----		-----	
h3_4.bmp to 4th class, CostValue	0.88	h11_5.bmp to 7th class, CostValue	1.76
-----		-----	
h3_5.bmp to 4th class, CostValue	1.15	h11_6.bmp to 11th class, CostValue	0.00
-----		-----	
h3_6.bmp to 3th class, CostValue	0.31	h11_7.bmp to 11th class, CostValue	0.00
-----		-----	
h4_2.bmp to 4th class, CostValue	0.93	h11_8.bmp to 11th class, CostValue	0.00
-----		-----	
h4_4.bmp to 4th class, CostValue	0.93	h11_9.bmp to 11th class, CostValue	0.00
-----		-----	
h4_5.bmp to 4th class, CostValue	0.94	h12_2.bmp to 12th class, CostValue	0.02
-----		-----	
h4_6.bmp to 4th class, CostValue	0.94	h12_4.bmp to 12th class, CostValue	0.02
-----		-----	
h5_2.bmp to 5th class, CostValue	0.58	h12_5.bmp to 12th class, CostValue	0.02
-----		-----	
h5_4.bmp to 5th class, CostValue	0.33	h12_6.bmp to 12th class, CostValue	0.02
-----		-----	
h5_5.bmp to 5th class, CostValue	0.27	h12_7.bmp to 12th class, CostValue	0.02
-----		-----	
h5_6.bmp to 5th class, CostValue	0.89	h12_8.bmp to 12th class, CostValue	0.02
-----		-----	
h7_2.bmp to 10th class, CostValue	1.44	h12_9.bmp to 12th class, CostValue	0.02
-----		-----	
h7_4.bmp to 10th class, CostValue	1.45	h12_10.bmp to 12th class, CostValue	0.02
-----		-----	
h7_5.bmp to 7th class, CostValue	1.09	// Rts245	
-----		-----	
h7_6.bmp to 10th class, CostValue	1.82	h1_1.bmp to 1th class, CostValue	0.56
-----		-----	

h1_3.bmp to 1th class, CostValue	0.58	-----	
h1_6.bmp to 1th class, CostValue	0.58	-----	
h2_1.bmp to 2th class, CostValue	0.08	-----	
h2_3.bmp to 2th class, CostValue	0.04	-----	
h2_6.bmp to 2th class, CostValue	0.04	-----	
h3_1.bmp to 3th class, CostValue	1.26	-----	
h3_3.bmp to 3th class, CostValue	1.41	-----	
h3_6.bmp to 3th class, CostValue	1.32	-----	
h4_1.bmp to 4th class, CostValue	0.47	-----	
h4_3.bmp to 3th class, CostValue	1.32	-----	
h4_6.bmp to 4th class, CostValue	0.47	-----	
h5_1.bmp to 5th class, CostValue	0.44	-----	
h5_3.bmp to 5th class, CostValue	0.66	-----	
h5_6.bmp to 5th class, CostValue	0.81	-----	
h7_1.bmp to 3th class, CostValue	1.48	-----	
h7_3.bmp to 7th class, CostValue	0.56	-----	
h7_6.bmp to 7th class, CostValue	1.36	-----	
h9_1.bmp to 9th class, CostValue	3.89	-----	
h9_3.bmp to 9th class, CostValue	3.86	-----	
h9_6.bmp to 9th class, CostValue	3.94	-----	
h10_1.bmp to 10th class, CostValue	0.59	-----	
h10_3.bmp to 10th class, CostValue	1.45	-----	
h10_6.bmp to 10th class, CostValue	0.71	-----	
h11_1.bmp to 11th class, CostValue	1.30	-----	
h11_3.bmp to 11th class, CostValue	1.30	-----	
h11_6.bmp to 11th class, CostValue	1.30	-----	
h11_7.bmp to 11th class, CostValue	1.30	-----	
h11_8.bmp to 11th class, CostValue	1.30	-----	
h11_9.bmp to 11th class, CostValue	1.30	-----	
h11_10.bmp to 7th class, CostValue	1.06	-----	

		h12_1.bmp to 12th class, CostValue	0.00

		h12_3.bmp to 12th class, CostValue	0.02

		h12_6.bmp to 12th class, CostValue	0.00

		h12_7.bmp to 12th class, CostValue	0.00

		h12_8.bmp to 12th class, CostValue	0.00

		h12_9.bmp to 12th class, CostValue	0.00

		h12_10.bmp to 12th class, CostValue	0.00

		// Rts1356	

		h1_2.bmp to 1th class, CostValue	0.43

		h1_4.bmp to 1th class, CostValue	0.43

		h2_2.bmp to 2th class, CostValue	0.05

		h2_4.bmp to 2th class, CostValue	0.05

		h3_2.bmp to 7th class, CostValue	1.17

		h3_4.bmp to 4th class, CostValue	0.64

		h4_2.bmp to 3th class, CostValue	1.54

		h4_4.bmp to 4th class, CostValue	0.24

		h5_2.bmp to 5th class, CostValue	0.24

		h5_4.bmp to 5th class, CostValue	0.49

		h7_2.bmp to 7th class, CostValue	0.92

		h7_4.bmp to 7th class, CostValue	0.82

		h9_2.bmp to 10th class, CostValue	1.31

		h9_4.bmp to 10th class, CostValue	1.40

		h10_2.bmp to 10th class, CostValue	1.16

		h10_4.bmp to 10th class, CostValue	0.66

		h11_2.bmp to 11th class, CostValue	1.23

		h11_4.bmp to 11th class, CostValue	1.31

		h12_2.bmp to 12th class, CostValue	0.02

		h12_4.bmp to 12th class, CostValue	0.02

		// Rts12456	

		h1_3.bmp to 1th class, CostValue	0.37

-----		h2_4.bmp to 2th class, CostValue	0.05	-----			
h2_3.bmp to 2th class, CostValue	0.05	-----		h2_5.bmp to 2th class, CostValue	0.04	-----	
-----		h3_3.bmp to 3th class, CostValue	0.69	-----		h2_6.bmp to 2th class, CostValue	0.04
-----		h4_3.bmp to 3th class, CostValue	1.42	-----		-----	
-----		h5_3.bmp to 5th class, CostValue	0.59	-----		h3_1.bmp to 3th class, CostValue	0.58
-----		h7_3.bmp to 7th class, CostValue	0.95	-----		-----	
-----		h9_3.bmp to 9th class, CostValue	1.86	-----		h3_2.bmp to 7th class, CostValue	1.10
-----		h10_3.bmp to 10th class, CostValue	1.37	-----		-----	
-----		h11_3.bmp to 11th class, CostValue	1.31	-----		h3_3.bmp to 3th class, CostValue	0.65
-----		h12_3.bmp to 12th class, CostValue	0.02	-----		-----	
-----		// RtsAll		-----		h3_4.bmp to 4th class, CostValue	0.65
-----		-----		-----		-----	
-----		h1_3.bmp to 1th class, CostValue	0.37	-----		h3_5.bmp to 3th class, CostValue	0.96
-----		-----		-----		-----	
-----		h2_3.bmp to 2th class, CostValue	0.05	-----		h3_6.bmp to 3th class, CostValue	0.52
-----		-----		-----		-----	
-----		h3_3.bmp to 3th class, CostValue	0.69	-----		h4_1.bmp to 4th class, CostValue	0.19
-----		-----		-----		-----	
-----		h4_3.bmp to 3th class, CostValue	1.42	-----		h4_2.bmp to 3th class, CostValue	1.43
-----		-----		-----		-----	
-----		h5_3.bmp to 5th class, CostValue	0.59	-----		h4_3.bmp to 3th class, CostValue	1.44
-----		-----		-----		-----	
-----		h7_3.bmp to 7th class, CostValue	0.95	-----		h4_4.bmp to 4th class, CostValue	0.18
-----		-----		-----		-----	
-----		h9_3.bmp to 9th class, CostValue	1.86	-----		h4_5.bmp to 4th class, CostValue	0.18
-----		-----		-----		-----	
-----		h10_3.bmp to 10th class, CostValue	1.37	-----		h4_6.bmp to 4th class, CostValue	0.19
-----		-----		-----		-----	
-----		h11_3.bmp to 11th class, CostValue	1.31	-----		h5_1.bmp to 5th class, CostValue	0.82
-----		-----		-----		-----	
-----		h12_3.bmp to 12th class, CostValue	0.02	-----		h5_2.bmp to 5th class, CostValue	0.23
-----		-----		-----		-----	
-----		h1_1.bmp to 1th class, CostValue	0.82	-----		h5_3.bmp to 5th class, CostValue	0.47
-----		-----		-----		-----	
-----		h1_2.bmp to 1th class, CostValue	0.32	-----		h5_4.bmp to 5th class, CostValue	0.52
-----		-----		-----		-----	
-----		h1_3.bmp to 1th class, CostValue	0.33	-----		h5_5.bmp to 5th class, CostValue	0.55
-----		-----		-----		-----	
-----		h1_4.bmp to 1th class, CostValue	0.32	-----		h5_6.bmp to 5th class, CostValue	0.43
-----		-----		-----		-----	
-----		h1_5.bmp to 1th class, CostValue	0.84	-----		h7_1.bmp to 3th class, CostValue	1.37
-----		-----		-----		-----	
-----		h1_6.bmp to 1th class, CostValue	0.32	-----		h7_2.bmp to 7th class, CostValue	0.80
-----		-----		-----		-----	
-----		h2_1.bmp to 2th class, CostValue	0.07	-----		h7_3.bmp to 7th class, CostValue	0.94
-----		-----		-----		-----	
-----		h2_2.bmp to 2th class, CostValue	0.05	-----		h7_4.bmp to 7th class, CostValue	0.71
-----		-----		-----		-----	
-----		h2_3.bmp to 2th class, CostValue	0.04	-----		h7_5.bmp to 7th class, CostValue	0.95
-----		-----		-----		-----	
-----				-----		h7_6.bmp to 7th class, CostValue	0.71
-----				-----		-----	
-----				-----		h9_1.bmp to 9th class, CostValue	1.85
-----				-----		-----	
-----				-----		h9_2.bmp to 10th class, CostValue	1.27
-----				-----		-----	
-----				-----		h9_3.bmp to 9th class, CostValue	1.80
-----				-----		-----	

----- h9_4.bmp to 10th class, CostValue	1.36
----- h9_5.bmp to 10th class, CostValue	1.40
----- h9_6.bmp to 9th class, CostValue	1.69
----- h10_1.bmp to 10th class, CostValue	0.57
----- h10_2.bmp to 10th class, CostValue	1.07
----- h10_3.bmp to 10th class, CostValue	1.27
----- h10_4.bmp to 10th class, CostValue	0.53
----- h10_5.bmp to 10th class, CostValue	0.46
----- h10_6.bmp to 10th class, CostValue	0.32
----- h11_1.bmp to 11th class, CostValue	0.67
----- h11_2.bmp to 11th class, CostValue	0.62
----- h11_3.bmp to 11th class, CostValue	0.67
----- h11_4.bmp to 11th class, CostValue	0.67
----- h11_5.bmp to 7th class, CostValue	1.02
----- h11_6.bmp to 11th class, CostValue	0.67
----- h11_7.bmp to 11th class, CostValue	0.67
----- h11_8.bmp to 11th class, CostValue	0.67
----- h11_9.bmp to 11th class, CostValue	0.67
----- h11_10.bmp to 7th class, CostValue	1.02
----- h12_1.bmp to 12th class, CostValue	0.02
----- h12_2.bmp to 12th class, CostValue	0.02
----- h12_3.bmp to 12th class, CostValue	0.01
----- h12_4.bmp to 12th class, CostValue	0.02
----- h12_5.bmp to 12th class, CostValue	0.02
----- h12_6.bmp to 12th class, CostValue	0.02
----- h12_7.bmp to 12th class, CostValue	0.02
----- h12_8.bmp to 12th class, CostValue	0.02
----- h12_9.bmp to 12th class, CostValue	0.02
----- h12_10.bmp to 12th class, CostValue	0.02

Appendix C

Representative Samples

Hong Kong Polytechnic
Centre for Professional and Continuing Education
Application Form

Name: Mr. T. S. CHAN (Printed Name)
Address: 100, Nathan Road, Kowloon, Hong Kong
Phone: 2332 1234
Course: Computer Studies
Enrollment Date: 1/1/92
Institution: City University of Hong Kong

Signature: T. S. Chan
Date: 1/1/92

Received from: City University of Hong Kong
Being for the account: City

Amount: 1000.00

Source sample: 1_1

Hong Kong Polytechnic
Centre for Professional and Continuing Education
Application Form

Name: Mr. T. S. CHAN (Printed Name)
Address: 100, Nathan Road, Kowloon, Hong Kong
Phone: 2332 1234
Course: Computer Studies
Enrollment Date: 1/1/92
Institution: City University of Hong Kong

Signature: T. S. Chan
Date: 1/1/92

Received from: City University of Hong Kong
Being for the account: City

Amount: 1000.00

Source sample: 1_3

Hong Kong Polytechnic
Centre for Professional and Continuing Education
Application Form

Name: Mr. T. S. CHAN (Printed Name)
Address: 100, Nathan Road, Kowloon, Hong Kong
Phone: 2332 1234
Course: Computer Studies
Enrollment Date: 1/1/92
Institution: City University of Hong Kong

Signature: T. S. Chan
Date: 1/1/92

Received from: City University of Hong Kong
Being for the account: City

Amount: 1000.00

Source sample: 1_2

Hong Kong Polytechnic
Centre for Professional and Continuing Education
Application Form

Name: Mr. T. S. CHAN (Printed Name)
Address: 100, Nathan Road, Kowloon, Hong Kong
Phone: 2332 1234
Course: Computer Studies
Enrollment Date: 1/1/92
Institution: City University of Hong Kong

Signature: T. S. Chan
Date: 1/1/92

Received from: City University of Hong Kong
Being for the account: City

Amount: 1000.00

Source sample: 1_4

The Hong Kong Polytechnic University

Application for Attending
Staff Development Programme (Local or Overseas)
Leading to Academic Award

Please read the Notes for Candidates on the last page before completing the form

1 Name: Paula Gough Staff No: 2810
 Department: CCBP Post: Assistant of Administration (A1)
 Date of first appointment: 9.10.16
 Terms of service: contract / permanent / temporary
 Leave balance: (vacation, sick leave)
 Previous University sponsored staff development leading to academic award in the past 5 years or University funded certificate attendance in the last 12 months:
 Date: _____ Amount paid: _____ staff development/certificate attendance:
 Date: 12.11.18 Amount paid: 12000.00 staff development/certificate attendance:
 Date: 12.11.18 Amount paid: 12000.00 staff development/certificate attendance:
 (Please use additional sheet if the above space is insufficient)

2 Title of Programme: Business Classification
Further Recognition
 Institution / Location: Hong Kong Polytechnic University
 Mode of study: Full-time / Part-time / Distance Learning /
 Other (please specify)
 Level of Award: Bachelor degree / Master degree / Doctorate /
 Other (please specify)
 Duration of programme: From _____ to _____

Source sample: 2_3

The Hong Kong Polytechnic University

Application for Attending
Staff Development Programme (Local or Overseas)
Leading to Academic Award

Please read the Notes for Candidates on the last page before completing the form

1 Name: Deena Staff No: 15111
 Department: Computing Post: Asst Location: 1113
 Date of first appointment: 9.9.16
 Terms of service: contract / permanent / temporary
 Leave balance: (vacation, sick leave)
 Previous University sponsored staff development leading to academic award in the past 5 years or University funded certificate attendance in the last 12 months:
 Date: 11.1.1 Amount paid: 5100.00 staff development/certificate attendance:
 Date: 11.1 Amount paid: 5100.00 staff development/certificate attendance:
 Date: 11.1 Amount paid: 5100.00 staff development/certificate attendance:
 (Please use additional sheet if the above space is insufficient)

2 Title of Programme: Debate & Negotiation
 Institution / Location: HK Polytechnic (1113)
 Mode of study: Full-time / Part-time / Distance Learning /
 Other (please specify)
 Level of Award: Bachelor degree / Master degree / Doctorate /
 Other (please specify)
 Duration of programme: From 11.9.1 to 11.9.1

Source sample: 2_5

The Hong Kong Polytechnic University

Application for Attending
Staff Development Programme (Local or Overseas)
Leading to Academic Award

Please read the Notes for Candidates on the last page before completing the form

1 Name: Liza Mui E Hing Staff No: 160057
 Department: Comp Post: Head Assistant of Education
 Date of first appointment: 18.9.1
 Terms of service: contract / permanent / temporary
 Leave balance: (vacation, sick leave)
 Previous University sponsored staff development leading to academic award in the past 5 years or University funded certificate attendance in the last 12 months:
 Date: 11.11.18 Amount paid: 12000.00 staff development/certificate attendance:
 Date: 11.11.18 Amount paid: 12000.00 staff development/certificate attendance:
 Date: _____ Amount paid: _____ staff development/certificate attendance:
 (Please use additional sheet if the above space is insufficient)

2 Title of Programme: Apply through Information Science
Results Achieved in a Probation Salary
 Institution / Location: _____
 Mode of study: Full-time / Part-time / Distance Learning /
 Other (please specify)
 Level of Award: Bachelor degree / Master degree / Doctorate /
 Other (please specify)
 Duration of programme: From _____ to _____

Source sample: 2_4

The Hong Kong Polytechnic University

Application for Attending
Staff Development Programme (Local or Overseas)
Leading to Academic Award

Please read the Notes for Candidates on the last page before completing the form

1 Name: Yip Shui Shing Staff No: 962871012
 Department: Computing the Knowledge Student Location: _____
 Date of first appointment: Sept 1 1997
 Terms of service: contract / permanent / temporary
 Leave balance: (vacation, sick leave)
 Previous University sponsored staff development leading to academic award in the past 5 years or University funded certificate attendance in the last 12 months:
 Date: _____ Amount paid: 5000 staff development/certificate attendance:
 Date: _____ Amount paid: 6000 staff development/certificate attendance:
 Date: _____ Amount paid: 7000 staff development/certificate attendance:
 (Please use additional sheet if the above space is insufficient)

2 Title of Programme: HK staff work
For the VC
 Institution / Location: _____
 Mode of study: Full-time / Part-time / Distance Learning /
 Other (please specify)
 Level of Award: Bachelor degree / Master degree / Doctorate /
 Other (please specify)
 Duration of programme: From Sept 1997 to Sept 1997

Source sample: 2_6

COMPUTER SOFTWARE REQUISITION FORM

Self Name _____ Date _____

SW REQUISITION:

Product Name _____
 Number of Copies _____ Cost/yr _____ Total Cost _____

Vendor Name _____
 Vendor Address _____
 Place of installation: _____ Date required _____
 Charge Against Research Account (Y/N): _____ Account No: _____ (if applicable)

Please tick the following:
 Staff Use Student Use Research Administration
 Other - Please specify: _____

Justification (e.g. personal use, student use, research, etc.)

If the funding comes from Dept 1 Software A/C, prior approval has to be obtained from Resource Committee

Approved By: _____
 Chairman, Resource Committee

N.B. (1) Please attach relevant information/specification for consideration.
 (2) Please try to complete the form as early as possible to allow necessary delay in order processing.

Source sample: 3_1

COMPUTER SOFTWARE REQUISITION FORM

Self Name John Nichol Date 22.2.91

SW REQUISITION:

Product Name: File Transfer Engineering
 Number of Copies: 2 Cost/yr: £2,800 Total Cost: £5,600

Vendor Name: _____
 Vendor Address: _____
 Place of installation: _____ Date required: _____
 Charge Against Research Account (Y/N): Y Account No: 126124 (if applicable)

Please tick the following:
 Staff Use Student Use Research Administration
 Other - Please specify: _____

Justification (e.g. personal use, student use, research, etc.)

If the funding comes from Dept 1 Software A/C, prior approval has to be obtained from Resource Committee

Approved By: _____
 Chairman, Resource Committee

N.B. (1) Please attach relevant information/specification for consideration.
 (2) Please try to complete the form as early as possible to allow necessary delay in order processing.

Source sample: 3_3

COMPUTER SOFTWARE REQUISITION FORM

Self Name: Peterina Gough Date: 22.2.91
Image Retrieval and Masking

SW REQUISITION:

Product Name: Dayling
 Number of Copies: 3 Cost/yr: £1200 Total Cost: £3600

Vendor Name: Hampton Road Slings
 Vendor Address: 210 Boro Works, Church Lane, 1st Fl, Colchester
 Place of installation: PHS Date required: 9.2.91
 Charge Against Research Account (Y/N): Y Account No: _____ (if applicable)

Please tick the following:
 Staff Use Student Use Research Administration
 Other - Please specify: _____

Justification (e.g. personal use, student use, research, etc.)
PHS - fitting (Lindsay Lobb)

If the funding comes from Dept 1 Software A/C, prior approval has to be obtained from Resource Committee

Approved By: [Signature]
 Chairman, Resource Committee

N.B. (1) Please attach relevant information/specification for consideration.
 (2) Please try to complete the form as early as possible to allow necessary delay in order processing.

Source sample: 3_2

COMPUTER SOFTWARE REQUISITION FORM

Self Name: Dianna Date: 22.2.91

SW REQUISITION:

Product Name: ST 4.0
 Number of Copies: 2 Cost/yr: £4,000 Total Cost: £8,000

Vendor Name: IBM
 Vendor Address: IBM
 Place of installation: PHS Date required: 22.2.91
 Charge Against Research Account (Y/N): Y Account No: _____ (if applicable)

Please tick the following:
 Staff Use Student Use Research Administration
 Other - Please specify: _____

Justification (e.g. personal use, student use, research, etc.)
PHS - IBM

If the funding comes from Dept 1 Software A/C, prior approval has to be obtained from Resource Committee

Approved By: [Signature]
 Chairman, Resource Committee

N.B. (1) Please attach relevant information/specification for consideration.
 (2) Please try to complete the form as early as possible to allow necessary delay in order processing.

Source sample: 3_4

COMPUTER SOFTWARE ACQUISITION FORM

Date Recd: 16 Sept 1997 Date: Sept 1997

SOFTWARE ACQUISITION:

Product Name: Windows 95

Number of Copies: 5 Country: USA Total Cost: 25000

Vendor Name: Microsoft Corp

Vendor Address: CA, USA

Place of Installation: Office Date required: Sept 2, 1997

Charge Against: Research Account 15194 Account No. (if applicable)

Please tick the following:

Staff Use Student Use Research Administration

Other - Please specify:

Justification: D.O.K.

If the funding comes from Dept 3 Software A/C, your approval has to be obtained from Research Committee

Approved by: [Signature]
Chairman, Research Committee

Source sample: 3_5

COMPUTER SOFTWARE ACQUISITION FORM

Date Recd: 16 Sept 1997 Date: 1/9/97

NEW ACQUISITION:

Product Name: Microsoft Windows 95

Number of Copies: 1 Country: USA Total Cost: 2000

Vendor Name: Microsoft

Vendor Address: CA, USA

Place of Installation: Office Date required: 1/9/97

Charge Against: Research Account 15194 Account No. (if applicable)

Please tick the following:

Staff Use Student Use Research Administration

Other - Please specify:

Justification: D.O.K.

If the funding comes from Dept 3 Software A/C, your approval has to be obtained from Research Committee

Approved by: [Signature]
Chairman, Research Committee

N.B. (1) Please attach relevant information/justification for acquisition.
(2) Please tick to complete the form on any relevant date if it is necessary doing so under pressure.

Source sample: 3_6

The Hong Kong Polytechnic University

APPLICATION FOR STAFF DEVELOPMENT PROGRAMME LEADING TO ACADEMIC AWARD

Please refer guidelines

- Sections 1 to 3 of the form should be completed by the applicant and the Head of Department should complete Section 4.
- The completed form should be forwarded by the Head of Department to the Secretary, Staff Development Committee.
- Any available information about the proposed programme should be submitted with the application form.
- Head of Department's justification
- Justification by the Head of Department should take note of some or all of the following: -
- (a) Value of the programme to the individual and the department.
 - (b) Reasons why the particular individual has been selected.
 - (c) An evaluation (if possible) of the chosen programme.
 - (d) How the department intends to provide cover during the staff member's absence.
- For details of level of funding and approval authority, applicants may refer to the revised policy paper (DCCD 08/96).
- (a) Staff-supervised study leave falls on study leave of 12 weeks or more.
 - (b) Full pay study leave refers to study leave of less than 12 weeks.
 - (c) Partial leave - non-approved study leave and full pay study leave are leave-taking arrangements that are approved or not approved by the staff member's immediate supervisor.
 - (d) Staff-supervised study leave falls on study leave of less than 12 weeks.
 - (e) Full pay study leave refers to study leave of less than 12 weeks.
 - (f) Staff-supervised study leave falls on study leave of less than 12 weeks.
 - (g) Staff-supervised study leave falls on study leave of less than 12 weeks.
- Applications not supported by Head of Department should also be sent to the Secretary, Staff Development Committee for record purposes.
- For applicants who are applying for financial assistance under HKDCCD, please attach an updated CV to this application form for consideration by the P Committee.

Source sample: 4_1

The Hong Kong Polytechnic University

APPLICATION FOR STAFF DEVELOPMENT PROGRAMME LEADING TO ACADEMIC AWARD

Please refer guidelines

- Sections 1 to 3 of the form should be completed by the applicant and the Head of Department should complete Section 4.
- The completed form should be forwarded by the Head of Department to the Secretary, Staff Development Committee.
- Any available information about the proposed programme should be submitted with the application form.
- Head of Department's justification
- Justification by the Head of Department should take note of some or all of the following: -
- (a) Value of the programme to the individual and the department.
 - (b) Reasons why the particular individual has been selected.
 - (c) An evaluation (if possible) of the chosen programme.
 - (d) How the department intends to provide cover during the staff member's absence.
- For details of level of funding and approval authority, applicants may refer to the revised policy paper (DCCD 08/96).
- (a) Staff-supervised study leave falls on study leave of 12 weeks or more.
 - (b) Full pay study leave refers to study leave of less than 12 weeks.
 - (c) Partial leave - non-approved study leave and full pay study leave are leave-taking arrangements that are approved or not approved by the staff member's immediate supervisor.
 - (d) Staff-supervised study leave falls on study leave of less than 12 weeks.
 - (e) Full pay study leave refers to study leave of less than 12 weeks.
 - (f) Staff-supervised study leave falls on study leave of less than 12 weeks.
 - (g) Staff-supervised study leave falls on study leave of less than 12 weeks.
- Applications not supported by Head of Department should also be sent to the Secretary, Staff Development Committee for record purposes.
- For applicants who are applying for financial assistance under HKDCCD, please attach an updated CV to this application form for consideration by the P Committee.

Source sample: 4_2

The Hong Kong Polytechnic University
APPLICATION FOR STAFF DEVELOPMENT PROGRAMME
LEADING TO ACADEMIC AWARD

Notes for guidance

- 1) Sections 1 to 3 of the form should be completed by the applicant and the Head of Department should complete Section 4.
- 2) The completed form should be forwarded by the Head of Department to the Secretary, Staff Development Committee.
- 3) Any available information about the proposed programme should be submitted with the application form.
- 4) Head of Department's justification
- Justification by the Head of Department should refer to one or all of the following:
- (a) Value of the programme to the individual and the Department.
 - (b) Reasons why the particular individual has been selected.
 - (c) An evaluation (if possible) of the chosen programme.
 - (d) How the programme relates to previous work during the staff member's absence.
- 5) For details of level of funding and approval authority, reference may be made to the revised policy paper (SDCC 5/94).
- 6) (a) Leave - suggested study leave refers to study leave of 12 weeks or more.
 (b) Full pay study leave refers to study leave of less than 12 weeks.
 (c) Half pay study leave refers to study leave of less than 12 weeks.
 (d) Staff granted leave - suggested for full pay study leave of 3 days or more and required to contribute to full work before or after 12 of the period of absence for the programme up to a maximum work leave of 90 days.
- 7) Applications not supported by Head of Department should also be sent to the Secretary, Staff Development Committee for record purposes.
- 8) For applicants who are applying for financial assistance above HK\$10,000, please attach an updated CV to the application form for consideration by the President.

Ernst Ludwig
9.7.91

Source sample: 4_3

The Hong Kong Polytechnic University
APPLICATION FOR STAFF DEVELOPMENT PROGRAMME
LEADING TO ACADEMIC AWARD

Notes for guidance

- 1) Sections 1 to 3 of the form should be completed by the applicant and the Head of Department should complete Section 4.
- 2) The completed form should be forwarded by the Head of Department to the Secretary, Staff Development Committee.
- 3) Any available information about the proposed programme should be submitted with the application form.
- 4) Head of Department's justification
- Justification by the Head of Department should refer to one or all of the following:
- (a) Value of the programme to the individual and the Department.
 - (b) Reasons why the particular individual has been selected.
 - (c) An evaluation (if possible) of the chosen programme.
 - (d) How the programme relates to previous work during the staff member's absence.
- 5) For details of level of funding and approval authority, reference may be made to the revised policy paper (SDCC 5/94).
- 6) (a) Leave - suggested study leave refers to study leave of 12 weeks or more.
 (b) Full pay study leave refers to study leave of less than 12 weeks.
 (c) Half pay study leave refers to study leave of less than 12 weeks.
 (d) Staff granted leave - suggested for full pay study leave of 3 days or more and required to contribute to full work before or after 12 of the period of absence for the programme up to a maximum work leave of 90 days.
- 7) Applications not supported by Head of Department should also be sent to the Secretary, Staff Development Committee for record purposes.
- 8) For applicants who are applying for financial assistance above HK\$10,000, please attach an updated CV to the application form for consideration by the President.

[Signature]

Source sample: 4_4

The Hong Kong Polytechnic University
APPLICATION FOR STAFF DEVELOPMENT PROGRAMME
LEADING TO ACADEMIC AWARD

Notes for guidance

- 1) Sections 1 to 3 of the form should be completed by the applicant and the Head of Department should complete Section 4.
- 2) The completed form should be forwarded by the Head of Department to the Secretary, Staff Development Committee.
- 3) Any available information about the proposed programme should be submitted with the application form.
- 4) Head of Department's justification
- Justification by the Head of Department should refer to one or all of the following:
- (a) Value of the programme to the individual and the Department.
 - (b) Reasons why the particular individual has been selected.
 - (c) An evaluation (if possible) of the chosen programme.
 - (d) How the programme relates to previous work during the staff member's absence.
- 5) For details of level of funding and approval authority, reference may be made to the revised policy paper (SDCC 5/94).
- 6) (a) Leave - suggested study leave refers to study leave of 12 weeks or more.
 (b) Full pay study leave refers to study leave of less than 12 weeks.
 (c) Half pay study leave refers to study leave of less than 12 weeks.
 (d) Staff granted leave - suggested for full pay study leave of 3 days or more and required to contribute to full work before or after 12 of the period of absence for the programme up to a maximum work leave of 90 days.
- 7) Applications not supported by Head of Department should also be sent to the Secretary, Staff Development Committee for record purposes.
- 8) For applicants who are applying for financial assistance above HK\$10,000, please attach an updated CV to the application form for consideration by the President.

Ang
9.7.91

Source sample: 4_5

The Hong Kong Polytechnic University
APPLICATION FOR STAFF DEVELOPMENT PROGRAMME
LEADING TO ACADEMIC AWARD

Notes for guidance

- 1) Sections 1 to 3 of the form should be completed by the applicant and the Head of Department should complete Section 4.
- 2) The completed form should be forwarded by the Head of Department to the Secretary, Staff Development Committee.
- 3) Any available information about the proposed programme should be submitted with the application form.
- 4) Head of Department's justification
- Justification by the Head of Department should refer to one or all of the following:
- (a) Value of the programme to the individual and the Department.
 - (b) Reasons why the particular individual has been selected.
 - (c) An evaluation (if possible) of the chosen programme.
 - (d) How the programme relates to previous work during the staff member's absence.
- 5) For details of level of funding and approval authority, reference may be made to the revised policy paper (SDCC 5/94).
- 6) (a) Leave - suggested study leave refers to study leave of 12 weeks or more.
 (b) Full pay study leave refers to study leave of less than 12 weeks.
 (c) Half pay study leave refers to study leave of less than 12 weeks.
 (d) Staff granted leave - suggested for full pay study leave of 3 days or more and required to contribute to full work before or after 12 of the period of absence for the programme up to a maximum work leave of 90 days.
- 7) Applications not supported by Head of Department should also be sent to the Secretary, Staff Development Committee for record purposes.
- 8) For applicants who are applying for financial assistance above HK\$10,000, please attach an updated CV to the application form for consideration by the President.

Data Semantics: What, where and How.
A. Sheth
Yeschi Shoy

Source sample: 4_6

Department of Computing
BACS Final Year Project Proposal Form

Supervisor Name:
Project Title:
Project Area Code:
Brief Description (for student reference):
Problem Statement:
General Approach:
Expected Outcome:

Source sample: 5_1

Department of Computing
BACS Final Year Project Proposal Form

Supervisor Name: <i>CM Ng</i>
Project Title: <i>Application Special Synthesis</i>
Project Area Code: <i>3501</i>
Brief Description (for student reference): <i>Project of Special Synthesis in M.S. in Management of Technology at the University of Manitoba, Canada. It is designed to show advanced M.S. & T. in Computer Technology from Queen University in 1997 and to compare complexity of M.S. in Electrical Engineering.</i>
Problem Statement: <i>In order to remain globally competitive, it is critical that organisations establish processes that allow them to adapt to an ever-changing market place.</i>
General Approach: <i>This need for agility is particularly acute, pronounced for organisations involved in the design and/or manufacture of complex products. In manufacturing organisations to adapt in an effective and efficient manner is essential to realising the necessary agility.</i>
Expected Outcome:

Source sample: 5_3

Department of Computing
BACS Final Year Project Proposal Form

Supervisor Name: <i>Architect</i>
Project Title: <i>Software Program Architecture</i>
Project Area Code:
Brief Description (for student reference): <i>此项目旨在研究软件架构设计的重要性，并探讨如何设计一个可扩展、可维护且安全的软件系统。项目将涉及需求分析、架构设计、实现和测试等阶段。</i>
Problem Statement: <i>软件架构是软件系统的核心，它决定了系统的性能、安全性和可扩展性。然而，许多软件系统在开发过程中缺乏良好的架构设计，导致系统难以维护、扩展和升级。</i>
General Approach: <i>本项目将采用敏捷开发方法，通过迭代的方式逐步完善软件架构。我们将使用UML进行需求分析和架构设计，并使用Java进行实现。项目还将涉及性能测试和安全性测试。</i>
Expected Outcome: <i>完成一个可扩展、可维护且安全的软件系统，并撰写一份详细的软件架构设计文档。</i>

Source sample: 5_2

Department of Computing
BACS Final Year Project Proposal Form


Supervisor Name: <i>Wif. Chan</i>
Project Title: <i>Handwritten Message</i>
Project Area Code: <i>2322</i>
Brief Description (for student reference): <i>Design an app yesterday.</i>
Problem Statement: <i>You can get details from Apple news paper. Or you can use message to get it from Internet.</i>
General Approach: <i>The bus is big. She is a princess of the world. She is always beautiful, good will, and missed.</i>
Expected Outcome: <i>She is alive in everyone's heart for ever.</i>

Source sample: 5_4

Department of Computing
BACS Final Year Project Proposal Form

Supervisor Name: <u>Ye Shunmy</u>
Project Title: <u>The application of</u>
Project Area Code: <u>1411/16</u>
Brief Description (for student reference): <u>the computer application.</u>
Problem Statement: <u>To solve the problem of computers.</u>
General Approach: <u>Using software and hardware. Sat Theory.</u>
Expected Outcome: <u>Software hardware.</u>

Source sample: 5_5

Department of Computing BACS Final Year Project Proposal Form
Supervisor Name: <u>Ye Shunmy</u>
Project Title: <u>document image classification by</u>
Project Area Code: <u>1411/16</u>
Brief Description (for student reference): <u>A basic function of any graphics program is to select portions of an image to manipulate. If you know and other painting programs.</u>
Problem Statement: <u>This chapter discusses the selection tools and how they can be filled, manipulated, filtered, ignored, copied, and pasted into you, will learn how selections apply to layers, this chapter, channels and paths, and the basis of compositing's image.</u>
General Approach: <u>blending a selection: Making a selection means drawing an area in image to which, manipulate for more. Make selections in the following way: - Use most the three selection tool (Lasso) - Use the Pen tool limited in the paths - Palettes (Properties) in this commands under the edit.</u>
Expected Outcome: 

Source sample: 5_6

Department of Computing BACS Final Year Project Proposal Form
Supervisor Name: <u>Ye Shunmy</u>
Project Title: <u>to Hong Kong</u>
Project Area Code: <u>1411/16</u>
Brief Description (for student reference): <u>the computer application.</u>
Problem Statement: <u>To solve the problem of computers.</u>
General Approach: <u>Using software and hardware. Sat Theory.</u>
Expected Outcome: <u>Software hardware.</u>

This is a test.

Hong Kong Polytechnic University.

Hong Kong Post office.

Microsoft

IBM

Intel

AS

Source sample: 6_1

Department of Computing BACS Final Year Project Proposal Form
Supervisor Name: <u>Ye Shunmy</u>
Project Title: <u>to Hong Kong</u>
Project Area Code: <u>1411/16</u>
Brief Description (for student reference): <u>the computer application.</u>
Problem Statement: <u>To solve the problem of computers.</u>
General Approach: <u>Using software and hardware. Sat Theory.</u>
Expected Outcome: <u>Software hardware.</u>

This is a test.

Hong Kong Polytechnic University.

Hong Kong Post office.



Microsoft

IBM

Intel

AS

Source sample: 6_2

 Hong Kong Polytechnic 香港理工大學		
From: Department of C.	To: Dept. D.	
Ref: 216 to Chris staff		
Tel No: 2762 1177	Year Ref: 13 m. C.	
Date: January 21, 1982	Date: 6. 25. 82	



BSE course wins top award

The Hong (Hons) programme in Building Services Engineering has been chosen by UK's Chartered Institution of Building Services Engineers as the first recipient of the prestigious Haggard Award.

The course was judged by the Examination Panel of the Institution to be the most innovative and well crafted course that they viewed during 1986. Presentation of the award to Prof. John Barnwell, Hd (B.S.E.), will take place on April 2 at the Royal Society of Arts in England.

Congratulations to Prof. Barnwell and his staff!



Source sample: 6_3

 Hong Kong Polytechnic 香港理工大學		
From: Peter, Lin	To: Mr. Huang	
Ref: Huang to Tom ID card		
Tel No: 2762 1177	Year Ref: 13 m. C.	
Date: Nov. 23, 1987	Date: Nov. 21, 1987	

I would like to inform you that your new ID card is ready and. Please come to the General Office (R0250) to take it. If you will have any question regarding on your ID card, feel free to contact me at 27623223.

Yours sincerely,
Peter, Lin.



Source sample: 6_5

 Hong Kong Polytechnic 香港理工大學		
From: Wu Liming	To: Hung Xudong	
Ref: in		
Tel No: 2762 1177	Year Ref: 13 m. C.	
Date: 25. 10. 87	Date:	

Form filling is very time consuming, thank you for your trusting in me. I will finish your assignment and tell you to total.

If there is anything you are not satisfied, please let me know, and if necessary I would like to do another ~~time~~ over for you.


Source sample: 6_4

 Hong Kong Polytechnic 香港理工大學		
From: Huang	To: Dora	
Ref: in 121		
Tel No: 7528	Year Ref: 22 Km	
Date: 26. 10. 87	Date:	

Nothing important, just need to make a sample to you.

I should change you by the way!

Source sample: 6_6

 To: Mr. Huang Xiu-Dong
From: Ya Shui-Shang


<input checked="" type="checkbox"/>	URGENT	急件
	FOR SIGNATURE	候簽
	FOR IMMEDIATE ACTION	敬請即辦
	FOR APPROVAL	希為核准
	FOR INFORMATION	祈為查照
	FOR RECORD	請予存錄
	FOR BRING UP	依期呈閱
	RETURNED WITH THANKS	謹此奉還
	PLEASE TELEPHONE	請即回電
	PHONE NO.	

Message: Pls call 7310

Date: Nov 24, 1997

512515

Source sample: 9_1

 To: Mr. Wang
From:


<input checked="" type="checkbox"/>	URGENT	急件
	FOR SIGNATURE	候簽
	FOR IMMEDIATE ACTION	敬請即辦
	FOR APPROVAL	希為核准
	FOR INFORMATION	祈為查照
	FOR RECORD	請予存錄
	FOR BRING UP	依期呈閱
	RETURNED WITH THANKS	謹此奉還
	PLEASE TELEPHONE	請即回電
	PHONE NO.	

Message: Please call back
27667313

Date: 24-11-97

512515

Source sample: 9_3

 To: Guzman
From: Davidson


<input checked="" type="checkbox"/>	URGENT	急件
	FOR SIGNATURE	候簽
	FOR IMMEDIATE ACTION	敬請即辦
	FOR APPROVAL	希為核准
	FOR INFORMATION	祈為查照
	FOR RECORD	請予存錄
	FOR BRING UP	依期呈閱
	RETURNED WITH THANKS	謹此奉還
	PLEASE TELEPHONE	請即回電
	PHONE NO.	<u>26167313</u>

Message: Please contact me
at soonest.

Date: 20/11/97

512515

Source sample: 9_2

 To:
From:

<input checked="" type="checkbox"/>	URGENT	急件
	FOR SIGNATURE	候簽
	FOR IMMEDIATE ACTION	敬請即辦
	FOR APPROVAL	希為核准
	FOR INFORMATION	祈為查照
	FOR RECORD	請予存錄
	FOR BRING UP	依期呈閱
	RETURNED WITH THANKS	謹此奉還
	PLEASE TELEPHONE	請即回電
	PHONE NO.	<u>27667313</u>

Message: Pls kindly call back
to me.

Date: 10/09/97

512515

Source sample: 9_4



To: Wu Liming
From: Huang Xudong

<input checked="" type="checkbox"/>	URGENT	急件
<input type="checkbox"/>	FOR SIGNATURE	候簽
<input type="checkbox"/>	FOR IMMEDIATE ACTION	敬請即辦
<input type="checkbox"/>	FOR APPROVAL	希為核准
<input type="checkbox"/>	FOR INFORMATION	祈為查照
<input type="checkbox"/>	FOR RECORD	請予存錄
<input type="checkbox"/>	FOR BRING UP	依期呈閱
<input type="checkbox"/>	RETURNED WITH THANKS	謹此奉還
<input type="checkbox"/>	PLEASE TELEPHONE	請即回電
PHONE NO. 2766 7315		

Message: All assignments have been finished, please check and collect them.

Date: 24/11/97

312513

Source sample: 9_5



To: Wang
From: XD

Hong Kong Polytechnic University

MEMO

Mr Wang,

I have had a phone call for you this afternoon, from library. You booked a proceeding and they have found it now. Please go and contact Mrs Wang for details.

XD
24-11-97

312514

Source sample: 10_1



To: 石福
From: 江國富

<input type="checkbox"/>	URGENT	急件
<input type="checkbox"/>	FOR SIGNATURE	候簽
<input type="checkbox"/>	FOR IMMEDIATE ACTION	敬請即辦
<input type="checkbox"/>	FOR APPROVAL	希為核准
<input type="checkbox"/>	FOR INFORMATION	祈為查照
<input type="checkbox"/>	FOR RECORD	請予存錄
<input type="checkbox"/>	FOR BRING UP	依期呈閱
<input type="checkbox"/>	RETURNED WITH THANKS	謹此奉還
<input type="checkbox"/>	PLEASE TELEPHONE	請即回電
PHONE NO.		

Message: 江國富 江國富 江國富 江國富 江國富 江國富 江國富 江國富 江國富 江國富

12/11/97

Date: 22.11.23

312515

Source sample: 9_6



To: Smarel
From: Andersson

Hong Kong Polytechnic University

MEMO


Some authors, usually concerned with comprehension of natural language, use 'semantic' as a vague term roughly synonymous with 'to do with meanings', where this means the same as 'not to do with grammar'. This follows a long and honourable tradition in linguistics.

Best wishes,

20/09/07

312516

Source sample: 10_2

	To: Mr. Huang	Hong Kong Polytechnic 香港理工大學
	From: Dea	


MEMO

We plan to visit the park
park next morning. Can you
go with us together? If possible,
please inform us if you would like
to do so.

Dea

24/2/18

Source sample: 10_3


	To: Huang Xudong	Hong Kong Polytechnic 香港理工大學
	From: Wu Lining	

MEMO

Hi - the morning of 24 Nov. 1917.
Huang Xudong sent Wu Lining several
forms to fill, Wu agree to finish
them before the evening of the
same day.

24/2/18

Source sample: 10_4

	To: Liao Yang	Hong Kong Polytechnic 香港理工大學
	From: Sheng shen	

MEMO


Dear Liao Yang,

This is the fifth consecutive year
when the Institution joined the Community
chest walk. Last year the Hoji
Team, for the third time, won
the Community chest's "Top Fund
Raising Team" Award (School
category) in the Hong Kong Walk
for its donation of over \$8,000.

9/2/16

21/2/18

Source sample: 10_5

	To: Mr. Huang Xudong	Hong Kong Polytechnic 香港理工大學
	From: Ye Shun'ang	

MEMO

Dear Mr Huang,

How are you going now?

This is a sample of
form and usage

Best Regards,

Ye Shun'ang

21/2/18

Source sample: 10_6

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telex/International Calls Service

To: Administrative Office, FBIS

Date: 25-07-94

From	Name / Dept.	FBIHQ1 - Sample 1
	Staff ID No.	12345677
	Extension No.	

*For Telex	Fax No. (With Area Code)	8922972
	Total No. of Pages to be transmitted	4

PURPOSE	<input type="checkbox"/> PRIVATE:	
	<input checked="" type="checkbox"/> OFFICIAL:	official

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries. 493 fax/m

Source sample: 11_1

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telex/International Calls Service

To: Administrative Office, FBIS

Date: 25-07-94

From	Name / Dept.	FBIHQ1 - Sample 1
	Staff ID No.	9229297
	Extension No.	

*For Telex	Fax No. (With Area Code)	91120729
	Total No. of Pages to be transmitted	3 pages

PURPOSE	<input type="checkbox"/> PRIVATE:	private
	<input type="checkbox"/> OFFICIAL:	

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries. 493 fax/m

Source sample: 11_2

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telex/International Calls Service

To: Administrative Office, FBIS

Date: 25-07-94

From	Name / Dept.	Setha Lam
	Staff ID No.	8525296
	Extension No.	7129

*For Telex	Fax No. (With Area Code)	20728522
	Total No. of Pages to be transmitted	4

PURPOSE	<input type="checkbox"/> PRIVATE:	
	<input type="checkbox"/> OFFICIAL:	

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries. 493 fax/m

Source sample: 11_3

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telex/International Calls Service

To: Administrative Office, FBIS

Date: 25-07-94

From	Name / Dept.	Y.C. CHAN
	Staff ID No.	85125095
	Extension No.	7103

*For Telex	Fax No. (With Area Code)	24222677
	Total No. of Pages to be transmitted	3

PURPOSE	<input type="checkbox"/> PRIVATE:	Recur.
	<input type="checkbox"/> OFFICIAL:	

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries. 493 fax/m

Source sample: 11_4

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telefax/International Calls Service

To: Administrative Offices, FBIS

Date: 23-07-94

From	Name / Dept.	<i>P. H. Kan</i>
	Staff ID No.	<i>C202451</i>
	Extension No.	<i>7276</i>

*Fax Telefax	Fax No. (With Area Code)	<i>0130460</i>
	Total No. of Pages to be transmitted	<i>0</i>

PURPOSE	<input checked="" type="checkbox"/> PRIVATE:	<i>Research</i>
	<input type="checkbox"/> OFFICIAL:	

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries 493 (extn)

Source sample: 11_5

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telefax/International Calls Service

To: Administrative Offices, FBIS

Date: 23-07-94

From	Name / Dept.	<i>JACK LO</i>
	Staff ID No.	<i>HK160172</i>
	Extension No.	<i>7274</i>

*Fax Telefax	Fax No. (With Area Code)	<i>2680419</i>
	Total No. of Pages to be transmitted	<i>2</i>

PURPOSE	<input type="checkbox"/> PRIVATE:	<i>Official</i>
	<input checked="" type="checkbox"/> OFFICIAL:	<i>Report</i>

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries 493 (extn)

Source sample: 11_7

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telefax/International Calls Service

To: Administrative Offices, FBIS

Date: 23-07-94

From	Name / Dept.	<i>JOEY CHAN</i>
	Staff ID No.	<i>9120171</i>
	Extension No.	<i>7279</i>

*Fax Telefax	Fax No. (With Area Code)	<i>2666991</i>
	Total No. of Pages to be transmitted	<i>2</i>

PURPOSE	<input type="checkbox"/> PRIVATE:	
	<input checked="" type="checkbox"/> OFFICIAL:	<i>Report</i>

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries 493 (extn)

Source sample: 11_6

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telefax/International Calls Service

To: Administrative Offices, FBIS

Date: 23-07-94

From	Name / Dept.	<i>BUNNIE CHAN</i>
	Staff ID No.	<i>9120195</i>
	Extension No.	<i>2522</i>

*Fax Telefax	Fax No. (With Area Code)	<i>8922972</i>
	Total No. of Pages to be transmitted	<i>3</i>

PURPOSE	<input type="checkbox"/> PRIVATE:	
	<input checked="" type="checkbox"/> OFFICIAL:	<i>Emergency</i>

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries 493 (extn)

Source sample: 11_8

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telefax/International Calls Service

To: Administrative Offices, FBIS

Date: 23/07/94

From	Name / Dept.	70201- Sample 2
	Staff ID No.	222922
	Extension No.	

Fax Telefax	Fax No. (With Area Code)	9170874
	Total No. of Pages to be transmitted	5

PURPOSE	<input checked="" type="checkbox"/> PRIVATE:	Private
	<input type="checkbox"/> OFFICIAL:	

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries, 493 faxin

Source sample: 11_9

HONG KONG POLYTECHNIC
FACULTY OF BUSINESS AND INFORMATION SYSTEMS

Requisition for Telefax/International Calls Service

To: Administrative Offices, FBIS

Date: 23/07/94

From	Name / Dept.	70201- Sample 3
	Staff ID No.	9123095
	Extension No.	

Fax Telefax	Fax No. (With Area Code)	9070133
	Total No. of Pages to be transmitted	10 Pages

PURPOSE	<input type="checkbox"/> PRIVATE:	
	<input checked="" type="checkbox"/> OFFICIAL:	Research

* Note: Please call Hong Kong Telephone Company at No 014 for directory enquiries, 493 faxin

Source sample: 11_10

Form No. H12-376



HONG KONG INSTITUTION OF SCIENCE

Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; application made on duplicate forms will not be accepted.

NAME IN ENGLISH	Cheng Alan (成安)	
Surname	Other names	Chinese
HKID No.	4 014014 (0)	

Notes:
(1) Names should be as they appear on Hong Kong Identity Card
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address:	Room 210, 123 Des Voeux Rd., Central.
Telephone:	3524 7812

Source sample: 12_1

Form No. H12-376



HONG KONG INSTITUTION OF SCIENCE

Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; application made on duplicate forms will not be accepted.

NAME IN ENGLISH	CHENG HARRY (成安)	
Surname	Other names	Chinese
HKID No.	K215623(0)	

Notes:
(1) Names should be as they appear on Hong Kong Identity Card
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address:	Room 210, 123 Des Voeux Rd., Central.
Telephone:	3524 7812

Source sample: 12_2



HONG KONG INSTITUTION OF SCIENCE
Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; applications made on duplicated forms will not be accepted.

NAME IN ENGLISH: FUNG Ho Bo (馮浩波)
Surname Other names Chinese
HKID No: A 01110 (7)

Notes:
(1) Names should be as they appear on Hong Kong Identity Card.
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: Tin Yee Street - WOO KEE
Telephone: _____

Source sample: 12_7



HONG KONG INSTITUTION OF SCIENCE
Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; applications made on duplicated forms will not be accepted.

NAME IN ENGLISH: POON P Sample (馮-18)
Surname Other names Chinese
HKID No: K 07659

Notes:
(1) Names should be as they appear on Hong Kong Identity Card.
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: Opp. 2nd Greenhouse, Happy Valley
Telephone: 2872 5522

Source sample: 12_9



HONG KONG INSTITUTION OF SCIENCE
Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; applications made on duplicated forms will not be accepted.

NAME IN ENGLISH: CHOW Sze on (周世安)
Surname Other names Chinese
HKID No: K 720966

Notes:
(1) Names should be as they appear on Hong Kong Identity Card.
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: 57, Fung Yee Street, CO-TRON, dr. 14
Telephone: _____

Source sample: 12_8



HONG KONG INSTITUTION OF SCIENCE
Application for Ordinary Membership

This application form is issued to: HONG KONG POLYTECHNIC UNIVERSITY
Other persons who wish to apply for membership should write to the Honorary Secretary of the Hong Kong Institution of Science to ask for an application form; applications made on duplicated forms will not be accepted.

NAME IN ENGLISH: POON P Sample (馮-18)
Surname Other names Chinese
HKID No: 626919

Notes:
(1) Names should be as they appear on Hong Kong Identity Card.
(2) If you do not hold a Hong Kong Identity Card, then enter passport number and also attach evidence showing a substantial connection with Hong Kong.

OFFICE Address: Central Bank, 2nd Greenhouse, Happy Valley
Telephone: 2892-2652

Source sample: 12_10

Appendix D

Part Source Code in C++

```
-----  
Program : Dic.h  
Function :  
-----  
  
#ifndef DIC  
#define DIC  
  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <malloc.h>  
#include <math.h>  
  
// the following three lines is used by:  
// 1. ARG.h & .c  
// 2. segment.c in function BoxAttributeWriteToFile  
#define MAXVERTEX 18 // maxnum of the vertices  
// Centre of X,Y; width, height; total number of black pixels N;  
// TH, TV, tx, width*height, N //, R, D, THx, TVx, THy, TVy.  
#define MAXV 10 // attributes of vertices  
// centre of (x1+x2)/2, (y1+y2)/2; up to Y, down to Y; left to X, right to X  
#define MAXA 6 // attributes between the two vertices  
  
// reduce parameter, used by:  
// 1. segment.c in the function BoxAttributeWriteToFile  
#define REDUCEP 3  
  
// used by the source file:  
// 1. segment.h & .c  
  
struct NODE // the abscissa of trace  
{  
    long int Xpixel; // x axis  
    long int Ypixel; // y axis  
    NODE* Next; // link to next node  
};  
  
// used by the source file:  
// 1. segment.h & .c  
  
struct BOX // saving the "enclose abscissa"  
{  
    long int LeftUpX;  
    long int LeftUpY;  
    long int RightDownX;  
    long int RightDownY;  
  
    // dx = RightDownX - LeftUpX  
    // dy = RightDownY - LeftUpY  
    int N, // total number of black pixels in a block of  
    // the original image  
    TH, // horizontal transitions of white to black pixels  
    // in a block of the original image  
    TV, // vertical transitions of white to black pixels  
    // in a block of the original image  
    tx; // when a block of the original image is projected  
    // onto x-axis, the number of columns in which  
    // black pixels exist  
    long int H; // the height of each block H = dy  
    float R, // the ratio of width to height (or aspect ratio)  
    // R = dx/dy  
    D, // the density of black pixels in a block,  
    // D = N/(dx*dy)  
    THx, // the horizontal transitions of white to black  
    // pixels per unit width, THx = TH/tx  
    TVx, // the vertical transitions of white to black  
    // pixels per unit width, TVx = TV/tx  
    THy, // the horizontal transitions of white to black  
    // pixels per unit height, THy = TH/dy  
    TVy; // the vertical transitions of white to black  
    // pixels per unit height, TVy = TV/dy  
  
    BOX* Next;  
};  
  
#include "segment.h"  
#include "ARG.h"  
  
#endif  
  
-----  
This is program for finding the optimal monomorphism between  
two attributed graphs  
Program : ARG.h
```

```

Function :
Purpose  : To match two graphs
Parms   :
Algorithm:
OS      : SunOS Release 4.1.3
Compiler: g++ -- GNU project c++ Compiler (v2.6)
History : Created date      : 7 May 1997
-----*/
#endif ARG
#define ARG

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <malloc.h>
#include <math.h>
#include "dic.h"

#define MAXVERTEX 16 defined in the dic.h
#define MAXV      12
#define MAXA      6

// used by the source file:
//      1. RAG.h & .c
class VertexAttribute
{
public:
    long int Attribute[MAXV]; // information relation to vertex
};

// used by the source file:
//      1. RAG.h & .c
class ArcAttribute
{
public:
    long int Attribute[MAXA]; // information relation to two vertices
};

// used by the source file:
//      1. RAG.h & .c
class RandomGraph
{
public:
    int          VertexNumber; // the number of vertex
    int          GrpOfVA;      // the groups of vertex attributes
    int          GrpOfAA;      // the groups of arc attributes
    VertexAttribute Vertex[MAXVERTEX+1];
    ArcAttribute   Arc[MAXVERTEX+1][MAXVERTEX+1];
};

// The class VertexAttribute, ArcAttribute and RandomGraph can be
// replaced by the struct, here, I only want to do it using the class.
// If use the class, please mind the using of "public":

//-----
// used by the source file:
//      1. RAG.h & .c
class MonomorphismGraphs
{
private:
    RandomGraph *pg, *bg;

    void ReadDataP(RandomGraph *);
    long int MatchAlgorithm(RandomGraph *, RandomGraph *);

public:
    void ARGTraining(int);
    void ARGTestting(char *);
};

#endif

/*-----
Program : ARG.c
Notes   :
Function: This is program for the match graphs
Purpose :
Parms   : the number of class
Algorithm: branch-and-bound
OS      : SunOS Release 4.1.3
Compiler: g++ -- GNU project c++ Compiler (v2.6)
History : Created date:      7 May 1997
        Last modified date: 28 Nov. 1997
-----*/

#include "ARG.h"

// The Training and Testting function are relation to the two data files.
// the structures of them is as follos:

```

```

// 1. The data file of vertex and arc - BGraph.txt
// (1)
// NumOfG - the number of the graphs
// NumOfV GrpofVA GrpofAA BelongToClass - NumofV: the no. of the vertexes;--+
// (2)
// Alv1 Alv2 ... AlvN GrpofVA: the vertex attributes
// A2v1 A2v2 ... A2vN GrpofAA: the arc attributes
// ... .. which class belongs to
// A8v1 A8v2 ... A8vN - the 8 attributes for the vertexes
// (3)
// R11v1 R11v2 ... R11vN
// R12v1 R12v2 ... R12vN
// ... ..
// Rin1 Rin2 ... RinN - the attributes for the first feature R1
//
// ... ..
// R61v1 R61v2 ... R61vN
// R62v1 R62v2 ... R62vN
// ... ..
// R6Nv1 R6Nv2 ... R6NvN - the attributes for the 6th feature R6 -----+
//
//
// 2. The file of the results for the after training - BGResults.txt
//
// LinesOfResults
// BelongToClass CostValue Balance
//

void MonomorphismGraphs::ARGTraining(int InputClass)
{
    FILE *ReadB, // open the file: BGraph.txt
        *ReadR; // open the file: BGResults.txt

    int NumOfG, // the number of the graphs
        NumOfV, // the number of the vertexes
        GrpOfVA, // the groups of vertex attributes
        GrpOfAA, // the groups of arc attributes
        BelongToClass, // which class is belong to
        LinesOfResults,
        tempData;

    int tagClass;
    int i, j, k, f, Num;
    int td1, td2, td3; // used by the reading lines in file BGResults.txt
    long int minCostValue, CostValue;

    tagClass = 0; // To record whether the InputClass is found. 0 no found.
    pg = (RandomGraph *) malloc(sizeof(RandomGraph));
    bg = (RandomGraph *) malloc(sizeof(RandomGraph));

    ReadDataP(pg);

    if((ReadB = fopen("BGraph.txt", "r")) == NULL)
    {
        // 1. the data file of base graph BGraph.txt is empty
        //
        // the pattern graph compare with itself
        // if the file BGraph.txt is empty
        //
        fclose(ReadB);
        ReadB = fopen("BGraph.txt", "a+");
        ReadR = fopen("BGResults.txt", "a+");

        // Part (1):
        //
        NumOfG = 1;
        ReadDataP(bg);
        Num = pg->VertexNumber;
        fprintf(ReadB, "%d\n", Num); // the number of graphs
        fprintf(ReadB, "%d\t%d\t%d\t%d\n", Num, MAXV, MAXA,
            InputClass);

        // Part (2):
        //
        for(i=1; i<MAXV+1; i++)
        {
            for(j=1; j<=Num; j++)
            {
                tempData = pg->Vertex[j].Attribute[i];
                fprintf(ReadB, "%d", tempData);
                if ( j == Num )
                    fprintf(ReadB, "\n");
                else
                    fprintf(ReadB, "\t");
            }
            fprintf(ReadB, "\n");
        }

        // Part (3):
        //
        for(i=1; i<MAXA+1; i++)
    {

```

```

for(j=1; j<=Num; j++)
  for(k=1; k<=Num; k++)
  {
    tempData = pg->Arc[j][k].Attribute[i];
    fprintf(ReadB, "%d", tempData);
    if( k == Num )
      fprintf(ReadB, "\n");
    else
      fprintf(ReadB, "\t");
  }
  fprintf(ReadB, "\n");
}

// To calculate the cost value, and write it to the file BGResults.txt
//
CostValue = MatchAlgorithm(pg, bg);
fprintf(ReadR, "\n\n"); // the line of result
fprintf(ReadR, "%3d\t%3d\t%3d\n", InputClass, CostValue, 0);
}
else
{ // 2. the file BGraph.txt is not empty
// the pattern graph compare with all appointed
// class in the file BGraph.txt, and recorded
// all the result CostValue.
//
tagClass = 0;
fclose(ReadB);
ReadB = fopen("BGraph.txt", "r+");
ReadR = fopen("BGResults.txt", "r+");

// Part (1)
fscanf(ReadB, "%d", &NumOfG);
//printf("%5d\n\n", NumOfG);
for(i=0; i<NumOfG; i++)
{
  fscanf(ReadB, "%d %d %d %d", &NumOfV, &GrpOfVA, &GrpOfAA,
        &BelongToClass);
  //printf("%5d %5d %5d %5d\n\n", NumOfV,
  //      GrpOfVA, GrpOfAA, BelongToClass);
  bg->VertexNumber = NumOfV;
  bg->GrpOfVA = GrpOfVA;
  bg->GrpOfAA = GrpOfAA;

  // Part(2):
  //
  for(j=1; j<=GrpOfVA; j++)
  {
    for(k=1; k<=NumOfV; k++)
    {
      fscanf(ReadB, "%d", &tempData);
      bg->Vertex[k].Attribute[j] = tempData;
      //printf("%5d", bg->Vertex[k].Attribute[j]);
    }
    //printf("\n");
  }
  //printf("\n");

  // Part (3):
  //
  for(j=1; j<=GrpOfAA; j++)
  {
    for(k=1; k<=NumOfV; k++)
    {
      for(f=1; f<=NumOfV; f++)
      {
        fscanf(ReadB, "%d", &tempData);
        bg->Arc[k][f].Attribute[j] = tempData;
        //printf("%5d", bg->Arc[k][f].Attribute[j]);
      }
      //printf("\n");
    }
    //printf("\n\n");
  }

  // check whether the class is included
  // if the BelongToClass is equal to InputClass
  // caculate the CostValue, and write the CostValue to the
  // file BGResults.txt
  //
  if (BelongToClass == InputClass)
  {
    tagClass = 1;
    if ( pg->VertexNumber <= bg->VertexNumber )
      CostValue = MatchAlgorithm(pg, bg);
    else
      CostValue = MatchAlgorithm(bg, pg);

    // search the input class, and to calculate the balance
    fseek(ReadR, 0L, 0); // put the file pointer at the begin of file
    fscanf(ReadR, "%d", &LinesOfResults);
    for(j=1; j<=LinesOfResults; j++)
    {
      fscanf(ReadR, "%d %d %d", &td1, &td2, &td3);
      if ( td1 == InputClass )

```

```

        {
            td3 = abs(td2 - CostValue);
            td2 = CostValue;
            break;
        }
    }
    fseek(ReadR, 0L, 0); // put the file pointer at the begin of file
    fprintf(ReadR, "%d", LinesOfResults+1);
    fseek(ReadR, 0L, 2); // put the file pointer at the end of file
    fprintf(ReadR, "%3d\t%3d\t%3d\n", InputClass, td2, td3);
    break;
}
} // End of for i
if(tagClass == 0)
{
    // append the new pattern graph data to base graph
    fseek(ReadB, 0L, 0);
    fprintf(ReadB, "%d", NumOEG+1);
    fseek(ReadB, 0L, 2);
    fprintf(ReadB, "\n");
    Num = pg->VertexNumber;
    fprintf(ReadB, "%3d\t%3d\t%3d\t%3d\n\n",
            Num, MAXV, MAXA, InputClass);
    for(i=1; i<MAXV+1; i++)
    {
        for(j=1; j<=Num; j++)
        {
            tempData = pg->Vertex[j].Attribute[i];
            fprintf(ReadB, "%6d", tempData);
            if ( j == Num )
                fprintf(ReadB, "\n");
            else
                fprintf(ReadB, "\t");
        }
        fprintf(ReadB, "\n");
        for(i=1; i<MAXA+1; i++)
        {
            for(j=1; j<=Num; j++)
                for(k=1; k<=Num; k++)
                {
                    tempData = pg->Arc[j][k].Attribute[i];
                    fprintf(ReadB, "%6d", tempData);
                    if ( k == Num )
                        fprintf(ReadB, "\n");
                    else
                        fprintf(ReadB, "\t");
                }
            fprintf(ReadB, "\n");
        }
    }
    ReadDataP(bg);
    // To calculate the cost value, and write it to the file BGRresults.txt
    //
    CostValue = MatchAlgorithm(pg, bg);
    fseek(ReadR, 0L, 0); // put the file pointer at the begin of file
    fscanf(ReadR, "%d", &LinesOfResults);
    fseek(ReadR, 0L, 0); // put the file pointer at the begin of file
    fprintf(ReadR, "%d", LinesOfResults+1);
    fseek(ReadR, 0L, 2); // put the file pointer at the end of file
    fprintf(ReadR, "%3d\t%3d\t%3d\n", InputClass, CostValue, 0);
} // end if tagClass == 0
} // end if ReadB
fclose(ReadB);
fclose(ReadR);
}

void MonomorphismGraphs::ARGTesting(char *TestFileName)
{
    FILE *ReadB, // open the file: BGraph.txt
        *ReadR, // open the file: BGRresults.txt
        *CostValuefp;
    int NumOEG, // the number of the graphs
        NumOEV, // the number of the vertices
        GrpOfVA, // the groups of vertex attributes
        GrpOfAA, // the groups of arc attributes
        LinesOfResults,
        BelongToClass, // which class is belong to
        tempData;
    int i, j, k, f, Num,
        td1, td2, td3;
    int *LastResults[2], minCostValue, CostValue, BClass;

    pg = (RandomGraph *) malloc(sizeof(RandomGraph));
    bg = (RandomGraph *) malloc(sizeof(RandomGraph));

    ReadDataP(pg);

    if((ReadB = fopen("BGraph.txt", "r")) == NULL)
    {
        printf("Can't open the file BGraph.txt!\n");
        exit(1);
    }
    ReadR = fopen("BGRresults.txt", "r");
}

```

```

CostValuefp = fopen("Results.txt", "a+");
fprintf(CostValuefp, "\n-----\n");
//fprintf(CostValuefp, "BelongToClass minCostValue\n");

fscanf(ReadB, "%d", &NumOfG);
//printf("%5d\n\n", NumOfG);

LastResults[0] = (int *) malloc( NumOfG * sizeof(int) ); // save BelongToClass
LastResults[1] = (int *) malloc( NumOfG * sizeof(int) ); // save minCostValue
for(i=0; i<NumOfG; i++)
{
    LastResults[0][i] = 0;
    LastResults[1][i] = 0;
}

for(i=0; i<NumOfG; i++)
{
    fscanf(ReadB, "%d %d %d %d", &NumOfV, &GrpOfVA, &GrpOfAA,
    &BelongToClass);
    //printf("%5d %5d %5d %5d\n\n", NumOfV,
    //      GrpOfVA, GrpOfAA, BelongToClass);
    bg->VertexNumber = NumOfV;
    bg->GrpOfVA = GrpOfVA;
    bg->GrpOfAA = GrpOfAA;
    for(j=1; j<=GrpOfVA; j++)
    {
        for(k=1; k<=NumOfV; k++)
        {
            fscanf(ReadB, "%d", &tempData);
            bg->Vertex[k].Attribute[j] = tempData;
            //printf("%5d", bg->Vertex[k].Attribute[j]);
        }
        //printf("\n");
    }
    //printf("\n");
    for(j=1; j<=GrpOfAA; j++)
    {
        for(k=1; k<=NumOfV; k++)
        {
            for(f=1; f<=NumOfV; f++)
            {
                fscanf(ReadB, "%d", &tempData);
                bg->Arc[k][f].Attribute[j] = tempData;
                //printf("%5d", bg->Arc[k][f].Attribute[j]);
            }
            //printf("\n");
        }
        //printf("\n\n");
    }

    if ( pg->VertexNumber <= bg->VertexNumber )
        CostValue = MatchAlgorithm(pg, bg);
    else
        CostValue = MatchAlgorithm(bg, pg);

    // search the input class, and to calculate the balance
    fseek(ReadR, 0L, 0); // put the file pointer at the begin of file
    fscanf(ReadR, "%d", &LinesOfResults);
    minCostValue = 0;
    BClass = 0;
    fprintf(CostValuefp, "CostValue is %3d\n", CostValue);
    for(j=1; j<=LinesOfResults; j++)
    {
        fscanf(ReadR, "%d %d %d", &td1, &td2, &td3);
        if ( td1 == BelongToClass )
        {
            BClass++;
            minCostValue = minCostValue + abs(abs(td2-CostValue)-td3);
            fprintf(CostValuefp, "%3d\t%3d\t%3d\n", td1, td2, td3);
        }
    }
    fprintf(CostValuefp, "%3d\n\n", minCostValue/BClass);
    LastResults[0][i] = BelongToClass;
    LastResults[1][i] = minCostValue/BClass;
} // End of for i

BClass = 0;
for(i=0; i<NumOfG; i++)
    if ( LastResults[1][i] <= 1 )
    {
        BClass = 1;
        fprintf(CostValuefp, "%s to %dth class, CostValue %d\n",
            TestFileName, LastResults[0][i], LastResults[1][i]);
    }
    if ( BClass == 0 )
        fprintf(CostValuefp, "%s doesn't belong to any class:\n",
            TestFileName);

fclose(ReadB);
fclose(ReadR);
fclose(CostValuefp);
}

```



```

void MonomorphismGraphs::ReadDataP(RandomGraph *rg)
{
    FILE *ReadP;          // open the data file of pattern graph
    int NumOfV,           // the number of the vertices
        GrpOfVA,         // the groups of vertex attributes
        GrpOfAA,         // the groups of arc attributes.
        tempData;

    int i, j, k, f;

    //printf("The data of the Pattern Graph\n\n");
    if((ReadP = fopen("PGraph.txt", "r")) == NULL)
    {
        printf("Can't open the file PatternGraph.txt!\n");
        exit(1);
    }
    fscanf(ReadP, "%d %d %d", &NumOfV, &GrpOfVA, &GrpOfAA);
    //printf("%5d%5d%5d\n", NumOfV, GrpOfVA, GrpOfAA);
    rg->VertexNumber = NumOfV;
    rg->GrpOfVA = GrpOfVA;
    rg->GrpOfAA = GrpOfAA;
    for(j=1; j<=GrpOfVA; j++)
    {
        for(k=1; k<=NumOfV; k++)
        {
            fscanf(ReadP, "%d", &tempData);
            rg->Vertex[k].Attribute[j] = tempData;
            //printf("%5d", rg->Vertex[k].Attribute[j]);
        }
        //printf("\n");
    }
    //printf("\n");
    for(j=1; j<=GrpOfAA; j++)
    {
        for(k=1; k<=NumOfV; k++)
        {
            for(f=1; f<=NumOfV; f++)
            {
                fscanf(ReadP, "%d", &tempData);
                rg->Arc[k][f].Attribute[j] = tempData;
                //printf("%5d", rg->Arc[k][f].Attribute[j]);
            }
            //printf("\n");
        }
        //printf("\n\n");
    }
    fclose(ReadP);
}

long int MonomorphismGraphs::MatchAlgorithm(RandomGraph *P, RandomGraph *B)
{
    FILE *moFp;
    int m, // the vertex index for the pattern graph P
        n, // the vertex index for the base graph B
        N[MAXVERTEX+1][4], // all; p; {P, B, fN} -- Path
        N1[MAXVERTEX+1][MAXVERTEX+1][4]; // all; p; {indexP, indexB, fN}
    int M1[MAXVERTEX+1],
        M2[MAXVERTEX+1],
        NumM2;
    int matchpoint, surpluspoint[MAXVERTEX+1];
    int tagN2, i, j, k, f, p, q, r, row, d;

    long int tempg1, tempg2, tempg3, upper_bound;
    long int qrmin, minqm2, gN, aN, bN, fN;
    long int dd, Num;

    moFp = fopen("mo.txt", "w+");
    //ReadDataP();
    //ReadDataB();

    m = P->VertexNumber;
    n = B->VertexNumber;

    ////////////////////////////////////////
    // "An Algorithm for Graph Optimal Monomorphism" is selected
    // from IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS,
    // VOL. 20 NO. 3, MAY/JUNE 1990 PP.630

    NumM2 = 0;
    for(i=0; i<MAXVERTEX+1; i++)
    {
        surpluspoint[i] = 0;
        M1[i] = 0;
        M2[i] = 0;
    }

    for(i=0; i<MAXVERTEX+1; i++)
        for(k=0; k<MAXVERTEX+1; k++)
            for(j=0; j<4; j++)
            {
                N[k][j] = 0;
                N1[i][k][j] = 0;
            }
    for(i=1; i<=n; i++)

```

```

surpluspoint[i] = i;
for(p=1; p<=m; p++)
{
for(row=1; row<=n-p+1; row++)
{
N1[row][p][1] = p; // index of pattern graph
N1[row][p][2] = surpluspoint[row]; // index of base graph
N1[row][p][3] = 0; // value of f(N)

N[p][1] = p;
N[p][2] = surpluspoint[row];
N[p][3] = 0;

// M1
Num = 0;
for(i=p+1; i<=m; i++)
{
Num++;
M1[Num] = i;
}

// M2
// = {p}[1, ..., Num][1, ..., n-p]
//
Num = 0;
for(j=1; j<=n; j++)
{
tagN2 = 0;
for(k=1; k<=p; k++)
if( N[k][2] == j )
tagN2 = 1;
if(tagN2 == 0)
{
Num++;
M2[Num] = j;
}
}
NumM2 = n-p;

// Compute f(N) = g*(N)+a(N)+b(N).
//
// for g*(N)
//
gN = 0;
tempg1 = tempg2 = tempg3 = 0;

// SUM(i=1,p)c(i,qi) i.e. c(1,q1)+...+c(i,qi)
// + SUM(i=1,p)SUM(j=1,p-1)c[(j,i),(qj,qi)]
// + SUM(i=1,p)SUM(j=1,p-1)c[(i,j),(qi,qj)]
for(i=1; i<=p; i++)
{
dd = 0;
for(d=1; d<=B->GrpOfVA; d++) // B->GrpOfVA = P->GrpOfVA
dd = dd +
abs(P->Vertex[i].Attribute[d] -
B->Vertex[ N[i][2] ].Attribute[d]);
tempg1 = tempg1 + dd;

// dd = dd +
// abs((P->Vertex[i].Attribute[d] -
// B->Vertex[ N[i][2] ].Attribute[d]) *
// (P->Vertex[i].Attribute[d] -
// B->Vertex[ N[i][2] ].Attribute[d]));
//tempg1 = tempg1 + (long int)sqrt(dd);
}
printf(mofp, "%f\n", tempg1);

for(j=1; j<=p-1; j++)
{
dd = 0;
for(d=1; d<=B->GrpOfAA; d++) // B->GrpOfAA = P->GrpOfAA
dd = dd +
abs(P->Arc[j][i].Attribute[d] -
B->Arc[ N[j][2] ][ N[i][2] ].Attribute[d]);

tempg2 = tempg2 + dd;

// dd = dd +
// abs((P->Arc[j][i].Attribute[d] -
// B->Arc[ N[j][2] ][ N[i][2] ].Attribute[d]) *
// (P->Arc[j][i].Attribute[d] -
// B->Arc[ N[j][2] ][ N[i][2] ].Attribute[d]));
//tempg2 = tempg2 + (long int)sqrt(dd);

dd = 0;
for(d=1; d<=B->GrpOfAA; d++)
dd = dd +
abs(P->Arc[i][j].Attribute[d] -
B->Arc[ N[i][2] ][ N[j][2] ].Attribute[d]);
tempg3 = tempg3 + dd;
}
}
}

```

```

// dd = dd +
// abs((P->Arc[i][j].Attribute[d] -
// B->Arc[ N[i][2] ][ N[j][2] ].Attribute[d]) *
// (P->Arc[i][j].Attribute[d] -
// B->Arc[ N[i][2] ][ N[j][2] ].Attribute[d]));
//tempg3 = tempg3 + (long int)sqrt(dd);
}
}
qN = tempg1 + tempg2 + tempg3;

// for a(N)
//
aN = 0;

// SUM(i=p+1,m)min(q in M2){ c(i,q) + SUM(j=1,p)c((i,j),(q,qj))
// + SUM(j=1,p)c((j,i),(qj,q)) }
for(i=p+1; i<=m; i++)
{
minqm2 = 999999;
tempg1 = 0;
for(k=1; k<=n-p; k++) // q in the M2
{
q = M2[k];
dd = 0;
for(d=1; d<=B->GrpOfVA; d++)
dd = dd +
abs(P->Vertex[i].Attribute[d] -
B->Vertex[q].Attribute[d]);
tempg1 = dd;

// dd = dd +
// abs((P->Vertex[i].Attribute[d] -
// B->Vertex[q].Attribute[d]) *
// (P->Vertex[i].Attribute[d] -
// B->Vertex[q].Attribute[d]));
//tempg1 = (long int)sqrt(dd);

tempg2 = tempg3 = 0;
for(j=1; j<=p; j++)
{
dd = 0;
for(d=1; d<=B->GrpOfAA; d++)
dd = dd +
abs(P->Arc[i][j].Attribute[d] -
B->Arc[q][ N[j][2] ].Attribute[d]);
tempg2 = tempg2 + dd;

// dd = dd +
// abs((P->Arc[i][j].Attribute[d] -
// B->Arc[q][ N[j][2] ].Attribute[d]) *
// (P->Arc[i][j].Attribute[d] -
// B->Arc[q][ N[j][2] ].Attribute[d]));
//tempg2 = tempg2 + (long int)sqrt(dd);

dd = 0;
for(d=1; d<=B->GrpOfAA; d++)
dd = dd +
abs(P->Arc[j][i].Attribute[d] -
B->Arc[ N[j][2] ][q].Attribute[d]);
tempg3 = tempg3 + dd;

// dd = dd +
// abs((P->Arc[j][i].Attribute[d] -
// B->Arc[ N[j][2] ][q].Attribute[d]) *
// (P->Arc[j][i].Attribute[d] -
// B->Arc[ N[j][2] ][q].Attribute[d]));
//tempg3 = tempg3 + (long int)sqrt(dd);

}
if (minqm2 >= tempg1+tempg2+tempg3)
minqm2 = tempg1+tempg2+tempg3;
}
aN = aN + minqm2;
}

// for b(N)
//
// SUM(i<j; i,j in M1)min(q<r; q,r in M2){ c((i,j),(q,r))+c((j,r),(r,q))}
bN = 0;
//fprintf(mofp, ".....\n");
for(j=m; j>=M1[1]; j--) // i, j belong to M1; and i<j
for(i=M1[1]; i<j; i++) //
{
upper_bound = 999999;
tagN2 = 0;
for(k=1; k<=NumM2; k++) //for q<r, and q,r
{ // belong to M2
q = M2[k];
for(f=1; f<=NumM2; f++)
{
r = M2[f];
if( r != q )

```

```

        //fprintf(mofp, "i=%d j=%d, q=%d r=%d\n", i, j, q, r);
        dd = 0;
        for(d=1; d<=B->GrpOfAA; d++)
            dd = dd +
                abs(P->Arc[i][j].Attribute[d] -
                    B->Arc[q][r].Attribute[d]);
        qrmin = dd;

        // dd = dd +
        //     abs((P->Arc[i][j].Attribute[d] -
        //         B->Arc[q][r].Attribute[d]) *
        //         (P->Arc[i][j].Attribute[d] -
        //         B->Arc[q][r].Attribute[d]));
        //qrmin = (long int)sqrt(dd);

        //fprintf(mofp, "PA%d%d -BA%d%d = %lu\n", i, j, q, r, qrmin);
        dd = 0;
        for(d=1; d<=B->GrpOfAA; d++)
            dd = dd +
                abs(P->Arc[j][i].Attribute[d] -
                    B->Arc[r][q].Attribute[d]);
        qrmin = qrmin + dd;

        // dd = dd +
        //     abs((P->Arc[j][i].Attribute[d] -
        //         B->Arc[r][q].Attribute[d]) *
        //         (P->Arc[j][i].Attribute[d] -
        //         B->Arc[r][q].Attribute[d]));
        //qrmin = qrmin + (long int)sqrt(dd);

        if(upper_bound >= qrmin)
        {
            upper_bound = qrmin;
            tagN2 = 1;
        }
    }
}
if(tagN2 == 0)
    upper_bound = 0;
bN = bN + upper_bound;
}

fN = gN + aN + bN;

N1[row][p][3] = fN;
} // End for row

// to find the minimum f(N), then assigne the matchpoint
upper_bound = 999999;
for(i=1; i<=n-p+1; i++)
    if ( upper_bound >= N1[i][p][3] )
    {
        upper_bound = N1[i][p][3];
        matchpoint = N1[i][p][2];
        N[p][1] = N1[i][p][1];
        N[p][2] = N1[i][p][2];
        N[p][3] = N1[i][p][3];
    }

// to find the surplus point
Num=0;
for(j=1; j<=n-p+1; j++)
{
    if(matchpoint != j)
    {
        Num++;
        surpluspoint[Num] = j;
    }
}

} // End for p

// printf("\nThe result is as follows:\n");
// for(i=1; i<=n; i++)
// {
//     for(j=1; j<=m; j++)
//         printf("%3d ", N1[i][j][3]);
//         //printf("(%d,%d),%3d ", N1[i][j][1], N1[i][j][2], N1[i][j][3]);
//     printf("\n");
// }
//
// printf("\nThe optimal cost is as follows:\n");
Num = 0;
for(i=1; i<=m; i++)
{
    //printf("(%d,%d),%3d\n", N[i][1], N[i][2], N[i][3]);
    Num = Num + N[i][3];
}
return Num; // return the f(N)
}

```

Appendix E

The Pattern and Process Result Files

3) The file of pattern file

2	8	6	1
81	84		
177	63		
158	168		
99	122		
0	0		
1172	983		
1744	2324		
158	169		
0	82		
82	0		
0	120		
120	0		
0	114		
0	0		
0	0		
114	0		
0	0		
0	0		
3	0		
0	3		
0	0		

8	8	6	2						
125	132	81	81	81	81	81	21		
222	215	199	188	174	127	53	13		
18	31	104	78	110	139	139	18		
1	4	5	11	2	72	69	1		
0	0	0	0	0	0	1	0		
9	16	113	141	73	607	310	5		
8	40	127	148	95	952	905	4		
18	33	104	79	111	139	139	18		
0	129	103	103	103	103	103	73		
129	0	107	107	107	107	107	77		
103	107	0	81	81	81	81	51		
103	107	81	0	81	81	81	51		
103	107	81	81	0	81	81	51		
103	107	81	81	81	0	81	51		
103	107	81	81	81	81	0	51		
73	77	51	51	51	51	51	0		
0	218	210	205	198	174	138	118		
218	0	207	201	194	171	134	114		
210	207	0	193	186	163	126	106		
205	201	193	0	181	157	120	100		
198	194	186	181	0	150	113	93		
174	171	163	157	150	0	90	70		
138	134	126	120	111	90	0	33		
118	114	106	100	93	70	33	0		
0	7	23	14	48	95	168	208		
0	0	15	27	41	87	161	201		
0	0	0	11	25	72	145	185		
0	0	0	0	14	60	134	174		
0	0	0	0	0	46	120	160		
0	0	0	0	0	0	73	113		
0	0	0	0	0	0	0	40		
0	0	0	0	0	0	0	0		
0	0	0	0	0	0	0	0		
7	0	0	0	0	0	0	0		
23	15	0	0	0	0	0	0		
34	27	11	0	0	0	0	0		
48	41	25	14	0	0	0	0		
95	87	72	60	46	0	0	0		
168	161	145	134	120	73	0	0		
208	201	185	174	160	113	40	0		
0	0	44	44	44	43	43	103		
7	0	51	51	51	51	51	111		
0	0	0	0	0	0	0	59		
0	0	0	0	0	0	0	59		
0	0	0	0	0	0	0	60		
0	0	0	0	0	0	0	60		
0	0	0	0	0	0	0	0		
0	7	0	0	0	0	0	0		
0	0	0	0	0	0	0	0		
44	51	0	0	0	0	0	0		
44	51	0	0	0	0	0	0		
44	51	0	0	0	0	0	0		
43	51	0	0	0	0	0	0		
43	51	0	0	0	0	0	0		
103	111	59	59	59	60	60	0		

15	8	6	3												
82	112	26	82	27	71	28	28	82	80	82	82	82	108	79	
213	197	197	189	176	171	159	152	144	121	101	91	67	48	28	
84	5	12	125	16	102	18	18	125	120	125	105	126	56	118	
3	1	1	14	1	6	1	1	11	20	7	0	7	8	9	
0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	
92	2	8	44	12	45	11	12	114	129	64	0	111	50	186	
72	3	10	184	11	90	14	15	199	155	127	0	166	84	203	
84	5	13	126	16	103	18	18	126	121	126	105	126	57	118	
0	97	54	82	54	76	55	55	82	81	82	82	82	95	80	
97	0	69	97	70	91	70	70	97	96	97	97	97	110	95	
54	69	0	54	26	48	27	27	54	53	54	54	54	67	52	
82	97	54	0	54	76	55	55	82	81	82	82	82	95	80	
54	70	26	54	0	49	28	28	55	53	55	54	55	68	53	
76	91	48	76	49	0	49	49	76	75	76	76	76	89	75	
55	70	27	55	28	49	0	28	55	54	55	55	55	68	53	
55	70	27	55	28	49	28	0	55	54	55	55	55	68	53	

82	97	54	82	55	76	55	55	0	81	82	82	82	95	80
81	96	51	81	51	75	54	54	81	0	81	81	81	94	79
82	97	54	82	55	76	55	55	82	81	0	82	82	95	80
82	97	54	82	54	76	55	55	82	81	82	0	82	95	80
82	97	54	82	55	76	55	55	82	81	82	82	0	95	80
95	110	67	95	68	89	68	68	95	94	95	95	95	0	93
80	95	52	80	51	75	51	53	80	79	80	80	80	93	0

0	205	205	201	194	192	186	183	179	167	157	152	140	130	120
205	0	197	193	185	184	178	175	171	159	149	144	132	123	112
205	197	0	191	186	184	178	175	171	159	149	144	132	122	112
201	193	193	0	182	180	174	170	166	155	145	140	128	118	108
194	185	186	182	0	173	167	164	160	148	138	133	121	112	102
192	184	184	180	171	0	165	162	158	146	136	131	119	110	99
186	178	178	174	167	165	0	156	152	140	130	125	113	103	93
183	175	175	170	164	162	156	0	148	137	126	121	110	100	90
179	171	171	166	160	158	152	148	0	133	122	117	106	96	86
167	159	159	155	148	146	140	137	133	0	111	106	94	84	74
157	149	149	145	138	136	130	126	122	111	0	96	84	74	64
152	144	144	140	133	131	125	121	117	106	96	0	79	69	59
140	132	132	128	121	119	113	110	106	94	84	73	0	57	47
130	123	122	118	112	110	103	100	96	84	74	69	57	0	38
120	112	112	108	102	99	93	90	86	74	64	59	47	38	0

0	15	16	24	37	41	54	60	68	92	112	122	146	165	185
0	0	0	8	21	26	38	45	53	76	96	106	130	149	169
0	0	0	8	21	25	38	44	52	76	96	106	130	149	169
0	0	0	0	13	17	29	36	44	67	88	98	121	140	161
0	0	0	0	0	4	16	23	31	54	75	85	108	127	148
0	0	0	0	0	0	12	19	27	50	70	80	104	123	143
0	0	0	0	0	0	0	6	14	38	58	68	92	111	131
0	0	0	0	0	0	0	0	8	31	51	61	85	104	124
0	0	0	0	0	0	0	0	0	23	41	53	77	96	116
0	0	0	0	0	0	0	0	0	0	20	30	54	73	93
0	0	0	0	0	0	0	0	0	0	10	33	52	73	93
0	0	0	0	0	0	0	0	0	0	0	23	42	61	81
0	0	0	0	0	0	0	0	0	0	0	0	19	39	59
0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	8	8	0	0	0	0	0	0	0	0	0	0	0	0
37	21	21	13	0	0	0	0	0	0	0	0	0	0	0
41	26	25	17	4	0	0	0	0	0	0	0	0	0	0
54	38	38	29	16	12	0	0	0	0	0	0	0	0	0
60	45	44	36	23	19	6	0	0	0	0	0	0	0	0
68	53	52	44	31	27	14	8	0	0	0	0	0	0	0
92	76	75	67	54	50	38	31	23	0	0	0	0	0	0
112	96	95	88	75	70	58	51	43	20	0	0	0	0	0
122	106	106	98	85	80	68	61	53	30	10	0	0	0	0
146	130	130	121	108	104	92	85	77	54	33	23	0	0	0
165	149	149	140	127	123	111	104	96	73	52	42	19	0	0
185	169	169	161	148	143	131	124	116	93	73	63	39	20	0

0	0	56	0	54	11	53	53	0	2	0	0	0	0	3
10	0	86	30	85	41	84	84	30	32	30	30	30	4	33
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	56	0	54	11	53	53	0	2	0	0	0	0	3
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	45	0	43	0	42	42	0	0	0	0	0	0	0
0	0	2	0	1	0	0	0	0	0	0	0	0	0	0
0	0	2	0	1	0	0	0	0	0	0	0	0	0	0
0	0	56	0	54	11	53	53	0	2	0	0	0	0	3
0	0	54	0	52	9	51	51	0	0	0	0	0	0	1
0	0	56	0	54	11	53	53	0	2	0	0	0	0	3
0	0	56	0	54	11	53	53	0	2	0	0	0	0	3
0	0	56	0	55	11	54	54	0	2	0	0	0	0	3
26	0	82	26	81	37	80	80	26	28	26	26	26	0	29
0	0	53	0	51	8	50	50	0	0	0	0	0	0	0

0	30	0	0	0	0	0	0	0	0	0	0	0	28	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
56	86	0	56	1	45	2	2	56	94	56	56	56	82	53
0	30	0	0	0	0	0	0	0	0	0	0	0	26	0
54	85	0	54	0	43	1	1	54	92	54	54	55	81	51
11	41	0	11	0	0	0	0	11	9	11	11	11	17	8
53	84	0	53	0	42	0	0	53	91	53	53	54	80	50
53	84	0	53	0	42	0	0	53	91	53	53	54	80	50
0	30	0	0	0	0	0	0	0	0	0	0	0	26	0
2	32	0	2	0	0	0	0	2	0	2	2	2	28	0
0	30	0	0	0	0	0	0	0	0	0	0	0	26	0
0	30	0	0	0	0	0	0	0	0	0	0	0	26	0
0	30	0	0	0	0	0	0	0	0	0	0	0	26	0
0	4	0	0	0	0	0	0	0	0	0	0	0	0	0
J	33	0	3	0	0	0	0	J	1	J	J	J	29	0

82	82	82	82	83	80	14	74	17	82	39	78	78	74	38	80	75	80
205	197	193	187	182	176	172	165	161	155	148	141	128	116	112	98	82	73
104	89	51	22	138	132	24	120	10	136	51	112	112	120	32	112	122	132
5	2	2	2	0	2	2	2	2	2	2	2	2	2	2	2	2	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
106	70	40	20	0	106	18	99	28	112	40	94	262	84	27	442	141	164
132	57	30	31	0	167	25	151	31	174	61	142	291	128	38	442	193	235
104	90	51	23	139	132	24	120	31	137	51	112	113	120	32	132	123	133

0	82	82	82	82	81	58	78	60	82	60	80	80	78	60	81	79	81
82	0	82	82	82	81	58	78	60	82	60	80	80	78	60	81	79	81
82	82	0	82	81	58	78	60	82	61	80	80	80	78	60	81	79	81
82	82	82	0	82	81	58	78	60	82	61	80	80	78	60	81	79	81
81	81	81	81	81	0	57	77	58	81	59	79	79	77	59	80	77	80
58	58	58	58	58	57	0	54	15	58	16	56	56	54	16	57	54	57
78	78	78	78	78	77	54	0	55	78	56	76	76	74	56	77	74	77
60	60	60	60	60	58	35	55	0	60	38	57	58	55	38	58	56	59
82	82	82	82	82	81	58	78	60	0	60	80	80	78	60	81	79	81
60	60	61	61	61	59	36	56	38	60	0	58	59	56	38	59	57	60
80	80	80	80	80	79	56	76	57	80	58	0	79	76	58	79	76	79
80	80	80	80	80	79	56	76	58	80	59	78	0	76	58	79	77	79
78	78	78	78	78	77	54	74	55	78	56	76	76	0	56	77	74	77
60	60	60	60	60	59	36	56	38	60	38	58	58	56	0	59	57	59
81	81	81	81	81	80	57	77	58	81	59	79	79	77	59	0	77	80
79	79	79	79	79	77	54	74	56	79	57	76	77	74	57	77	0	78
81	81	81	81	81	80	57	77	59	81	60	79	79	77	59	80	78	0

0	201	199	196	193	190	188	185	183	180	176	171	166	160	158	151	143	139
201	0	195	192	1													

193	189	187	184	0	179	177	174	172	168	165	161	155	149	147	140	132	127
190	186	184	181	179	0	174	171	169	165	162	158	152	146	144	137	129	124
188	184	182	179	177	174	0	169	167	163	160	156	150	144	142	135	127	122
185	181	179	176	174	171	169	0	161	160	156	153	147	140	139	132	124	119
183	179	177	174	172	169	167	163	0	158	154	151	145	138	137	130	122	117
180	176	174	171	168	165	163	160	158	0	151	148	141	135	133	126	118	114
176	172	170	167	165	162	160	156	154	151	0	144	138	132	130	123	115	110
173	169	167	164	161	158	156	153	151	148	144	0	135	128	126	119	112	107
166	161	161	157	155	152	150	147	145	141	138	135	0	132	120	113	105	101
160	156	154	151	149	146	144	140	138	135	132	128	122	0	114	107	99	94
158	154	152	149	147	144	142	139	137	133	130	126	120	114	0	105	97	92
151	147	145	142	140	137	135	132	130	126	123	119	113	107	105	0	90	85
143	140	138	134	132	129	127	124	122	118	115	112	105	99	97	90	0	78
139	135	133	130	127	124	122	119	117	114	110	107	101	94	92	85	78	0
0	7	11	18	22	28	32	39	43	50	57	61	76	89	92	106	122	131
0	0	4	10	15	21	25	31	35	42	49	56	68	81	85	99	114	124
0	0	0	6	11	17	21	27	31	38	45	52	64	77	81	95	110	120
0	0	0	0	4	10	14	21	25	32	39	45	58	71	74	88	104	111
0	0	0	0	0	6	10	16	20	27	34	41	53	66	70	84	99	109
0	0	0	0	0	4	10	14	21	28	35	47	60	74	78	93	103	103
0	0	0	0	0	0	6	10	17	24	31	43	56	70	74	89	99	99
0	0	0	0	0	0	0	4	10	17	24	37	49	63	67	83	92	92
0	0	0	0	0	0	0	0	6	13	20	33	45	49	63	79	88	88
0	0	0	0	0	0	0	0	0	7	13	26	39	42	56	72	81	81
0	0	0	0	0	0	0	0	0	0	6	19	32	45	49	65	74	74
0	0	0	0	0	0	0	0	0	0	0	12	25	29	43	58	68	68
0	0	0	0	0	0	0	0	0	0	0	0	12	16	30	46	55	55
0	0	0	0	0	0	0	0	0	0	0	0	0	3	17	33	42	42
0	0	0	0	0	0	0	0	0	0	0	0	0	0	14	29	39	39
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	25	25
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	10	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	15	11	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	21	17	10	6	0	0	0	0	0	0	0	0	0	0	0	0	0
32	25	21	14	10	4	0	0	0	0	0	0	0	0	0	0	0	0
39	31	27	21	16	10	6	0	0	0	0	0	0	0	0	0	0	0
43	35	31	25	20	14	10	4	0	0	0	0	0	0	0	0	0	0
50	42	38	32	27	21	17	10	6	0	0	0	0	0	0	0	0	0
57	49	45	39	34	28	24	17	13	7	0	0	0	0	0	0	0	0
63	56	52	45	41	35	31	24	20	13	6	0	0	0	0	0	0	0
76	68	64	58	53	47	43	37	33	26	19	12	0	0	0	0	0	0
89	81	77	71	66	60	56	49	45	39	32	25	12	0	0	0	0	0
92	85	81	74	70	64	60	53	49	42	35	29	16	3	0	0	0	0
106	99	95	88	84	78	74	67	63	56	49	43	30	17	14	0	0	0
122	114	110	104	99	93	89	83	79	72	65	58	46	33	29	15	0	0
131	124	120	113	109	103	99	92	88	81	74	68	55	42	39	25	9	0
0	0	0	0	0	2	48	8	44	0	41	4	3	8	44	2	6	1
0	0	0	0	0	2	48	8	44	0	41	4	3	8	44	2	6	1
0	0	0	0	0	2	48	8	45	0	43	4	4	8	44	2	7	2
0	0	0	0	0	2	48	8	45	0	43	4	4	8	44	2	7	2
0	0	0	0	0	3	49	9	45	0	43	5	4	9	44	3	7	2
0	0	0	0	0	0	46	6	42	0	40	2	1	6	41	0	4	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	40	0	36	0	34	0	0	0	35	0	0	0
0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	2	48	8	44	0	41	4	3	8	44	2	6	1
0	0	0	0	0	0	5	0	1	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	44	4	40	0	38	0	0	4	39	0	2	0
0	0	0	0	0	0	44	4	41	0	39	0	0	4	40	0	3	0
0	0	0	0	0	0	40	0	36	0	34	0	0	0	35	0	0	0
0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	46	6	42	0	40	2	1	6	41	0	4	0
0	0	0	0	0	0	41	1	38	0	36	0	0	1	37	0	0	0
0	0	0	0	0	0	46	6	43	0	41	2	2	6	42	0	5	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	2	2	2	2	3	0	0	0	2	0	0	0	0	0	0	0	0
48	48	48	48	49	46	0	40	3	48	5	44	44	40	4	46	41	46
8	8	8	8	9	6	0	0	0	8	0	4	4	0	0	6	1	6
44	44	45	45	45	42	0	36	0	44	1	40	41	36	0	42	38	43
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	43	43	43	43	40	0	34	0	43	0	18	39	34	0	40	36	41
4	4	4	4	4	5	2	0	0	4	0	0	0	0	0	2	0	2
3	3	4	4	4	1	0	0	0	3	0	0	0	0	1	0	0	2
8	8	8	8	9	6	0	0	0	8	0	4	4	0	0	6	1	6
44	44	44	44	44	41	0	35	0	44	1	39	40	35	0	41	37	42
2	2	2	2	3	0	0	0	0	2	0	0	0	0	0	0	0	0
6	6	7	7	7	4	0	0	0	6	0	2	1	0	4	0	5	0
1	1	2	2	2	0	0	0	0	1	0	0	0	0	0	0	0	0

2 8 6 5

81	82
219	115
67	134
8	194
0	3
112	564
117	856
68	135
0	82
82	0
0	167
167	0
0	104
0	0
0	0
104	0
0	0
0	0
0	0
0	0

15 8 6 7

87	24	82	21	85	82	84	71	37	84	42	37	95	51	87
216	197	188	179	178	172	148	127	115	102	84	78	82	69	25
153	10	126	4	120	126	118	91	23	115	30	0	56	51	0
16	2	11	1	1	5	26	11	8	13	10	1	13	10	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
134	10	46	2	14	38	204	72	24	116	34	0	46	50	0
319	6	188	2	79	175	253	115	29	158	42	0	74	60	0
151	11	126	5	120	126	118	92	24	115	30	0	56	51	0

0	55	84	54	86	84	85	79	62	85	65	62	91	69	87
55	0	53	22	54	53	54	48	30	54	33	31	59	18	56
84	53	0	51	83	82	83	76	59	83	62	59	88	46	84
54	22	51	0	53	51	52	46	29	52	32	29	58	16	54
86	54	83	53	0	83	84	78	61	84	63	61	90	68	86
84	53	82	51	84	0	81	76	59	83	62	59	88	66	84
85	54	83	52	84	83	0	77	60	84	63	60	89	67	85
79	48	76	46	78	76	77	0	54	78	57	54	83	61	79
62	30	59	29	61	59	60	54	0	60	39	37	66	44	62
85	54	83	52	84	83	84	78	60	0	63	61	89	68	86
65	33	62	32	63	62	63	57	39	61	0	40	69	47	65
62	31	59	29	61	59	60	54	37	61	40	0	66	44	62
91	59	88	58	90	88	89	83	66	89	69	66	0	73	91
69	38	66	36	68	66	67	61	44	68	47	44	73	0	69
87	56	84	54	86	84	85	79	62	86	65	62	91	69	0

0	206	202	197	197	194	182	171	165	159	150	147	149	142	120
206	0	193	188	188	185	172	162	156	149	140	137	139	133	111
202	193	0	183	183	180	168	158	152	145	136	133	135	129	107
197	188	183	0	178	175	163	153	147	140	131	128	130	124	102
197	188	183	178	0	175	163	153	147	140	131	128	130	124	102
194	185	180	175	175	0	160	150	144	137	129	125	127	121	99
182	172	168	163	163	160	0	138	131	125	116	113	115	108	87
171	162	158	153	153	150	138	0	121	115	106	102	105	98	76
165	156	152	147	147	144	131	121	0	108	99	96	98	92	70
159	149	145	140	140	137	125	115	108	0	93	90	92	85	64
150	140	136	131	131	128	116	106	99	93	0	81	83	76	55
147	137	133	128	128	125	113	102	96	90	81	0	80	73	51
149	139	135	130	130	127	115	105	98	92	83	80	0	75	54
142	133	129	124	124	121	108	98	92	85	76	73	75	0	47
120	111	107	102	102	99	87	76	70	64	55	51	54	47	0

0	18	27	37	37	43	67	88	100	113	131	138	133	146	190
0	0	8	18	18	24	49	69	82	95	113	119	115	128	171
0	0	0	9	10	16	40	61	73	86	104	110	106	119	163
0	0	0	0	0	6	30	51	63	76	94	101	96	109	153
0	0	0	0	0	6	30	51	63	76	94	100	96	109	153
0	0	0	0	0	0	24	45	57	70	88	94	90	103	147
0	0	0	0	0	0	20	33	46	64	70	66	79	122	
0	0	0	0	0	0	0	12	25	43	49	45	58	102	
0	0	0	0	0	0	0	0	13	31	37	31	46	89	
0	0	0	0	0	0	0	0	0	18	24	20	33	76	
0	0	0	0	0	0	0	0	0	0	6	2	15	58	
0	0	0	0	0	0	0	0	0	0	0	0	8	52	
0	0	0	0	0	0	0	0	0	0	0	4	0	13	56
0	0	0	0	0	0	0	0	0	0	0	0	0	0	43
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	8	0	0	0	0	0	0	0	0	0	0	0	0	0
37	18	9	0	0	0	0	0	0	0	0	0	0	0	0
37	18	10	0	0	0	0	0	0	0	0	0	0	0	0
43	24	16	6	6	0	0	0	0	0	0	0	0	0	0
67	49	40	30	30	24	0	0	0	0	0	0	0	0	0
88	69	61	51	51	45	20	0	0	0	0	0	0	0	0
100	82	73	63	63	57	33	12	0	0	0	0	0	0	0
113	95	86	76	76	70	46	25	13	0	0	0	0	0	0
131	113	104	94	94	88	64	43	31	18	0	0	0	0	0
138	119	110	101	100	94	70	49	37	24	6	0	4	0	0
133	115	106	96	96	90	66	45	33	20	2	0	0	0	0
146	128	119	109	109	101	79	58	46	33	15	8	13	0	0
190	171	163	151	151	147	122	102	89	76	59	52	56	43	0

0	63	5	46	2	5	3	15	50	3	44	49	0	35	0
0	0	0	60	0	0	0	0	0	0	0	0	0	0	0
0	57	0	0	0	0	0	10	45	0	39	44	0	39	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	60	3	63	0	3	1	13	48	0	42	47	0	31	0
0	57	0	60	0	0	0	10	45	0	39	44	0	30	0
0	59	2	62	0	2	0	12	47	0	41	46	0	32	0
0	47	0	50	0	0	0	0	34	0	29	34	0	20	0
0	12	0	15	0	0	0	0	0	0	0	0	0	0	0
0	60	2	63	0	2	0	12	47	0	41	46	0	32	0
0	18	0	21	0	0	0	0	5	0	0	5	0	0	0
0	13	0	16	0	0	0	0	0	0	0	0	0	0	0
8	71	13	74	10	13	11	23	58	11	52	57	0	43	7
0	27	0	30	0	0	0	0	14	0	9	14	0	0	0
0	63	5	66	2	5	3	16	50	3	45	50	0	36	0

0	0	0	0	0	0	0	0	0	0	0	0	8	0	0
63	0	57	0	60	57	59	47	12	60	18	13	71	27	63
5	0	0	0	3	0	2	0	0	2	0	0	13	0	5
66	3	60	0	63	60	62	50	15	63	21	16	74	30	66
2	0	0	0	0	0	0	0	0	0	0	0	10	0	2
5	0	0	0	3	0	2	0	0	2	0	0	11	0	5
3	0	0	0	1	0	0	0	0	0	0	0	11	0	3
15	0	10	0	13	10	12	0	0	12	0	0	23	0	16
50	0	45	0	48	45	47	34	0	47	5	0	58	14	50
3	0	0	0	0	0	0	0	0	0	0	0	11	0	3
44	0	39	0	42	39	41	29	0	41	0	0	52	9	45
49	0	44	0	47	44	46	34	0	46	5	0	57	14	50
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	30	0	33	30	32	20	0	32	0	0	43	0	36
0	0	0	0	0	0	0	0	0	0	0	0	7	0	0

72 8 6 8

87	41	86	149	81	157	86	82	103	61	33	48
198	150	110	116	110	84	89	75	57	60	36	15
151	46	104	0	130	0	136	124	1	90	27	36
52	14	9	0	24	0	12	11	1	8	9	14
1	0	0	0	1	0	0	0	0	0	0	0
185	59	95	0	191	0	111	99	0	64	25	36
502	66	128	0	246	0	194	162	0	105	39	51
153	47	104	1	130	1	137	125	1	81	27	37
0	64	86	118	84	122	86	84	95	74	60	67
64	0	61	95	61	99	63	61	72	51	17	44
86	63	0	117	84	121	86	84	94	73	59	67
118	95	117	0	115	151	117	115	126	105	91	98
84	61	84	115	0	119	84	81	92	71	57	65

122	99	121	153	119	0	121	119	130	109	95	102
86	63	86	117	84	121	0	84	94	71	59	67
84	61	84	115	81	119	84	0	92	71	57	65
95	72	94	126	92	130	94	92	0	82	68	75
74	51	73	105	71	109	73	71	82	0	47	54
60	37	59	91	57	95	59	57	68	47	0	40
67	44	67	98	65	102	67	65	75	54	40	0
0	174	164	157	154	141	143	137	128	129	117	106
174	0	140	133	130	117	119	113	104	105	91	82
164	140	0	123	120	107	109	103	94	95	81	72
157	133	123	0	113	100	102	96	87	88	76	65
154	130	120	113	0	97	99	92	81	85	71	62
141	117	107	100	97	0	86	80	71	72	60	49
143	119	109	102	99	86	0	82	71	74	62	52
137	113	103	96	92	80	82	0	66	67	55	45
128	104	94	87	83	71	73	66	0	58	46	36
128	105	95	88	85	72	74	67	58	0	48	37
117	93	83	76	73	60	62	55	46	48	0	25
106	82	72	65	62	49	52	45	36	37	25	0
0	48	68	82	88	114	109	122	140	138	162	183
0	0	20	34	40	66	61	74	92	90	114	135
0	0	0	14	20	46	41	54	72	70	94	115
0	0	0	0	6	12	27	40	58	56	80	101
0	0	0	0	0	25	20	34	52	50	74	94
0	0	0	0	0	0	0	8	26	24	48	69
0	0	0	0	0	5	0	13	31	29	53	74
0	0	0	0	0	0	0	0	18	15	39	60
0	0	0	0	0	0	0	0	0	0	21	42
0	0	0	0	0	0	0	0	2	0	24	44
0	0	0	0	0	0	0	0	0	0	0	20
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0
68	20	0	0	0	0	0	0	0	0	0	0
82	34	14	0	0	0	0	0	0	0	0	0
88	40	20	6	0	0	0	0	0	0	0	0
114	66	46	32	25	0	5	0	0	0	0	0
109	61	41	27	20	0	0	0	0	0	0	0
122	74	54	40	34	8	13	0	0	0	0	0
140	92	72	58	52	26	31	18	0	2	0	0
138	90	70	56	50	24	29	15	0	0	0	0
162	114	94	80	74	48	53	39	21	24	0	0
183	135	115	101	94	69	74	60	42	44	20	0
0	46	1	0	5	0	1	5	0	26	54	39
0	0	0	0	0	0	0	0	0	0	7	0
0	45	0	0	4	0	0	4	0	25	53	38
61	108	63	0	67	0	62	67	46	88	115	100
0	40	0	0	0	0	0	0	0	20	48	33
70	116	71	8	75	0	71	75	54	96	124	109
0	45	0	0	4	0	0	4	0	25	53	38
0	41	0	0	0	0	0	0	0	21	48	33
15	62	16	0	21	0	16	21	0	42	69	54
0	20	0	0	0	0	0	0	0	0	27	12
0	0	0	0	0	0	0	0	0	0	0	0
0	7	0	0	0	0	0	0	0	0	15	0
0	0	0	61	0	70	0	0	15	0	0	0
46	0	45	108	40	116	45	41	62	20	0	7
1	0	0	62	0	71	0	0	16	0	0	0
0	0	0	0	0	8	0	0	0	0	0	0
5	0	4	67	0	75	4	0	21	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	62	0	71	0	0	16	0	0	0
5	0	4	67	0	75	4	0	21	0	0	0
0	0	0	46	0	54	0	0	0	0	0	0
26	0	25	88	20	96	25	21	42	0	0	0
54	7	53	115	48	124	53	48	69	27	0	15
39	0	38	100	33	109	38	33	54	12	0	0

5	8	6	9	
61	62	66	24	63
125	60	48	39	15
112	107	57	10	108
98	14	0	0	13
0	1	0	0	0
882	107	0	0	106
1411	152	0	0	146
112	108	57	10	108
0	61	63	42	62
61	0	64	43	62
63	64	0	45	64
42	43	45	0	43
62	62	64	43	0
0	93	86	82	70
93	0	54	49	17
86	54	0	43	11
82	49	43	0	27
70	37	31	27	0
0	64	77	86	110
0	0	12	21	45
0	0	0	9	13
0	0	0	0	24
0	0	0	0	0
0	0	0	0	0
64	0	0	0	0
77	12	0	0	0
86	21	9	0	0
110	45	33	24	0
0	0	0	37	0
1	0	0	38	0
5	3	0	42	3
0	0	0	0	0
2	0	0	19	0
0	3	5	0	2
0	0	3	0	0
0	0	0	0	0
37	38	42	0	39
0	0	3	0	0

56	55	56	22	62	59	62	92	41	55	22	78	16	20	7
170	158	146	135	123	111	97	85	86	74	64	47	36	26	11
107	107	107	27	73	100	101	29	57	87	19	6	8	17	6
0	20	0	7	8	9	8	6	6	7	5	4	5	5	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	170	0	24	59	75	74	21	18	60	20	6	12	19	4
0	206	0	32	98	136	133	19	64	111	25	8	10	22	3
108	107	108	27	80	101	101	29	57	87	19	6	8	18	7
0	55	56	39	59	57	59	74	48	55	39	67	36	38	31
55	0	55	39	59	57	59	74	48	55	39	66	36	38	31
56	55	0	39	59	57	59	74	48	55	39	67	36	38	31
39	39	39	0	42	41	42	57	31	38	22	50	19	21	14
59	59	59	42	0	61	62	77	51	58	42	70	39	41	35
57	57	57	41	61	0	61	76	50	57	41	68	38	40	33
59	59	59	42	62	61	0	77	51	58	42	70	39	41	34
74	74	74	57	77	76	77	0	66	73	57	85	54	56	50
48	48	48	31	51	50	51	66	0	48	31	59	28	30	24
55	55	55	38	58	57	58	73	48	0	18	66	35	37	31
39	39	39	22	42	41	42	57	31	38	0	50	19	21	15
67	66	67	50	70	68	70	85	59	66	50	0	47	49	42
16	16	16	19	39	38	39	54	28	35	19	47	0	18	12
18	18	18	21	41	40	41	56	30	37	21	49	18	0	13
31	31	31	14	35	33	34	50	24	31	15	42	12	13	0
0	164	158	153	147	140	134	128	128	122	117	108	103	98	91
164	0	152	146	141	134	127	122	122	116	111	102	97	92	85
158	152	0	140	134	128	121	115	116	110	105	96	91	86	78
153	146	140	0	129	123	116	110	110	104	100	91	86	81	73
147	141	134	129	0	117	110	104	104	99	94	85	80	75	67
140	124	128	123	117	0	104	98	98	92	87	79	73	68	61
134	127	121	116	110	104	0	91	91	85	81	72	67	62	54
128	122	115	110	104	98	91	0	85	80	75	66	61	56	48
128	122	116	110	104	98	91	85	0	80	75	66	61	56	48
122	116	110	104	99	92	85	80	80	0	69	60	55	50	43
117	111	105	100	94	87	81	75	75	69	0	55	50	45	38
108	102	96	91	85	79	72	66	66	60	55	0	41	36	29
103	97	91	86	80	73	67	61	61	55	50	41	0	31	24
98	92	86	81	75	68	62	56	56	50	45	16	11	0	19
91	85	78	73	67	61	54	48	48	43	38	29	24	19	0
0	12	24	35	47	59	73	85	84	96	106	123	134	144	159
0	0	12	23	34	47	61	72	72	84	93	111	121	131	146
0	0	0	10	22	35	48	60	60	71	81	99	109	119	134
0	0	0	0	11	24	38	49	49	61	70	88	98	108	123
0	0	0	0	0	12	26	38	37	49	59	76	87	97	112
0	0	0	0	0	13	25	36	36	46	56	74	84	94	109
0	0	0	0	0	0	11	11	11	23	32	50	60	70	85
0	0	0	0	0	0	0	0	0	11	21	38	49	59	74
0	0	0	0	0	0	0	0	0	11	21	39	49	59	74
0	0	0	0	0	0	0	0	0	0	9	27	37	47	62
0	0	0	0	0	0	0	0	0	0	0	17	28	38	53
0	0	0	0	0	0	0	0	0	0	0	10	20	35	50
0	0	0	0	0	0	0	0	0	0	0	0	10	25	40
0	0	0	0	0	0	0	0	0	0	0	0	0	15	30
0	0	0	0	0	0	0	0	0	0	0	0	0	0	15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	12	0	0	0	0	0	0	0	0	0	0	0	0	0
35	23	10	0	0	0	0	0	0	0	0	0	0	0	0
47	34	22	11	0	0	0	0	0	0	0	0	0	0	0
59	47	35	24	12	0	0	0	0	0	0	0	0	0	0
73	61	48	38	26	17	0	0	0	0	0	0	0	0	0
85	72	60	49	38	25	11	0	0	0	0	0	0	0	0
84	72	60	49	37	25	11	0	0	0	0	0	0	0	0
96	84	71	61	49	36	23	11	11	0	0	0	0	0	0
106	93	81	70	59	46	32	21	21	9	0	0	0	0	0
123	111	99	88	76	64	50	38	39	27	17	0	0	0	0
134	121	109	98	87	74	60	49	49	37	28	10	0	0	0
144	131	119	108	97	84	70	59	59	47	38	20	10	0	0
159	146	134	123	112	99	85	74	74	62	53	35	25	15	0
0	0	0	33	0	0	0	0	15	1	33	0	39	35	48
0	0	0	33	0	0	0	0	14	0	33	0	39	35	48
0	0	0	33	0	0	0	0	15	1	33	0	39	35	48
0	0	0	0	0	0	0	0	0	0	0	0	5	2	15
6	7	6	40	0	3	0	0	21	7	40	0	46	42	55
3	4	3	37	0	0	0	0	18	4	37	0	43	39	52
6	6	6	40	0	2	0	0	21	7	39	0	45	42	55
36	37	36	70	30	33	30	0	51	37	70	14	76	72	85
0	0	0	18	0	0	0	0	0	0	18	0	24	20	33
0	0	0	32	0	0	0	0	14	0	32	0	38	34	47
0	0	0	0	0	0	0	0	0	0	0	0	6	2	15
22	22	22	55	15	18	15	0	37	23	55	0	61	57	70
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	15	0	21	18	21	53	0	14	0	17	0	0	0
1	0	1	0	7	4	7	37	0	0	0	23	0	0	0
33	33	33	0	40	37	39	70	18	32	0	55	0	0	0
0	0	0	0	0	0	0	14	0	0	0	0	0	0	0
39	39	39	5	46	43	45	76	24	38	6	61	0	0	0
35	35	35	2	42	39	42	72	20	34	2	57	0	0	0
48	48	48	15	55	52	55	85	33	47	15	70	9	13	0
10	8	6	11											
98	98	101	160	54	100	99	100	94	28					
232	223	204	190	189	157	99	45	12	6					
77	144	111	25	57	158	158	159	150	17					
3	3	4	2	3	35	35	22	3	2					
0	0	0	0	0	1	1	0	0	0					
69	139	113	20	58	262	291	158	152	18					
60	110	158	21	80	629	563	286	209	17					
77	144	112	26	57	159	159	159	150	17					
0	98	99	129	76	99	99	99	96	63					
98	0	99	129	76	99	99	99	96	63					
99	99	0	130	77	100	100	100	97	65					
129	129	110	0	107	130	129	130	127	94					
76	76	77	107	0	77	76	77	74	41					

99	99	100	130	77	0	99	100	97	64
99	99	100	129	76	99	0	99	97	64
99	99	100	130	77	100	99	0	97	64
96	96	97	127	74	97	97	0	61	0
63	63	65	94	41	64	64	64	61	0
0	227	218	211	210	194	165	138	122	119
227	0	211	206	206	190	161	134	117	114
218	211	0	197	196	180	151	124	108	105
211	206	197	0	190	174	144	117	101	98
210	206	196	190	0	171	144	117	101	98
194	190	180	174	173	0	128	101	85	82
165	161	151	144	144	128	0	72	55	52
138	134	124	117	117	101	72	0	28	25
122	117	108	101	101	85	55	28	0	9
119	114	105	98	98	82	52	25	9	0

0	8	28	41	42	74	133	186	219	225
0	0	19	33	33	65	124	178	211	217
0	0	0	13	14	46	105	158	191	197
0	0	0	0	0	32	91	145	178	184
0	0	0	0	0	32	91	144	177	183
0	0	0	0	0	0	59	112	145	151
0	0	0	0	0	0	0	53	86	92
0	0	0	0	0	0	0	0	33	39
0	0	0	0	0	0	0	0	0	6
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
28	19	0	0	0	0	0	0	0	0
41	31	13	0	0	0	0	0	0	0
42	33	14	0	0	0	0	0	0	0
74	65	46	32	32	0	0	0	0	0
133	124	105	91	91	59	0	0	0	0
186	178	158	145	144	112	53	0	0	0
219	211	191	178	177	145	86	33	0	0
225	217	197	184	183	151	92	39	6	0

0	0	0	0	44	0	0	0	4	69
0	0	0	0	44	0	0	0	4	69
7	1	0	0	47	1	1	1	7	72
61	61	58	0	106	60	60	60	65	131
0	0	0	0	0	0	0	0	0	25
1	1	0	0	46	0	0	0	5	71
1	1	0	0	45	0	0	0	5	71
1	1	0	0	46	0	0	0	5	71
0	0	0	0	40	0	0	0	0	65
0	0	0	0	0	0	0	0	0	0

0	0	3	61	0	1	1	1	0	0
0	0	3	61	0	1	1	1	0	0
0	0	0	58	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
44	44	47	106	0	46	45	46	40	0
0	0	1	60	0	0	0	0	0	0
0	0	1	60	0	0	0	0	0	0
0	0	1	60	0	0	0	0	0	0
4	4	7	65	0	5	5	5	0	0
69	69	72	131	25	71	71	71	65	0

9 8 6 12

175	116	116	25	102	101	102	108	102	
247	231	218	223	180	160	129	90	19	
41	116	96	20	178	181	163	166	165	
5	4	5	19	20	0	39	20	55	
0	0	0	0	0	0	1	0	1	
37	124	115	51	575	0	254	438	207	
51	89	131	34	546	0	409	468	567	
42	116	96	20	178	181	163	166	166	

0	145	145	100	138	138	138	141	138	
145	0	116	71	109	108	109	112	109	
145	116	0	71	109	108	109	112	109	
100	71	71	0	64	63	64	67	64	
138	109	109	64	0	101	102	105	102	
138	108	108	63	101	0	101	104	101	
138	109	109	64	102	101	0	105	102	
141	112	112	67	105	104	105	0	105	
138	109	109	64	102	101	102	105	0	

0	239	232	235	213	201	188	168	143	
239	0	224	227	206	195	180	161	135	
232	224	0	220	199	189	173	154	128	
235	227	220	0	202	191	176	157	131	
213	206	199	202	0	170	154	135	110	
201	195	189	191	170	0	144	125	99	
188	180	173	176	154	144	0	109	84	
168	161	154	157	135	125	109	0	65	
143	135	128	131	110	99	84	65	0	

0	15	29	23	66	87	118	156	207	
0	0	13	8	50	71	102	140	192	
0	0	0	0	37	58	89	127	178	
0	0	5	0	42	63	94	132	184	
0	0	0	0	0	20	51	90	141	
0	0	0	0	0	0	31	69	120	
0	0	0	0	0	0	0	38	89	
0	0	0	0	0	0	0	0	51	
0	0	0	0	0	0	0	0	0	

0	0	0	0	0	0	0	0	0	
15	0	0	0	0	0	0	0	0	
29	13	0	5	0	0	0	0	0	
23	8	0	0	0	0	0	0	0	
66	50	17	42	0	0	0	0	0	
87	71	58	63	20	0	0	0	0	
118	102	89	94	51	11	0	0	0	
156	140	127	132	90	69	38	0	0	
207	192	178	184	141	120	89	51	0	

0	58	58	149	72	74	72	66	72	
0	0	0	91	14	15	14	8	14	
0	0	0	91	14	15	14	8	14	
0	0	0	0	0	0	0	0	0	
0	0	0	77	0	1	0	0	0	
0	0	0	75	0	0	0	0	0	
0	0	0	76	0	1	0	0	0	
0	0	0	82	5	7	6	0	6	
0	0	0	76	0	1	0	0	0	

0	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0	0
149	91	91	0	77	75	76	82	76	
72	14	14	0	0	0	0	5	0	
74	15	15	0	1	0	1	7	1	
72	14	14	0	0	0	0	6	0	
66	8	8	0	0	0	0	0	0	
72	14	14	0	0	0	0	6	0	

b) The file of process and result

```

k) 3.bmp belongs to .....
The match result of the lowest cost f(N), and quota
0th BG f(N) = 8
   quota(1) = 7      BelongToClass: 1

1th BG f(N) = 4
   quota(1) = 0      BelongToClass: 1
   quota(2) = 4      BelongToClass: 1

2th BG f(N) = 8
   quota(1) = 8      BelongToClass: 1
   quota(2) = 4      BelongToClass: 1
   quota(3) = 7      BelongToClass: 1

3th BG f(N) = 1
   quota(1) = 6      BelongToClass: 1
   quota(2) = 3      BelongToClass: 1
   quota(3) = 6      BelongToClass: 1
   quota(4) = 1      BelongToClass: 1

4th BG f(N) = 6
   quota(1) = 2      BelongToClass: 1
   quota(2) = 5      BelongToClass: 1
   quota(3) = 2      BelongToClass: 1
   quota(4) = 2      BelongToClass: 1
   quota(5) = 6      BelongToClass: 1

5th BG f(N) = 2
   quota(1) = 2      BelongToClass: 2

6th BG f(N) = 0
   quota(1) = 3      BelongToClass: 2
   quota(2) = 0      BelongToClass: 2

7th BG f(N) = 0
   quota(1) = 3      BelongToClass: 2
   quota(2) = 0      BelongToClass: 2
   quota(3) = 0      BelongToClass: 2

8th BG f(N) = 7
   quota(1) = 2      BelongToClass: 2
   quota(2) = 0      BelongToClass: 2
   quota(3) = 0      BelongToClass: 2
   quota(4) = 7      BelongToClass: 2

9th BG f(N) = 1
   quota(1) = 3      BelongToClass: 2
   quota(2) = 1      BelongToClass: 2
   quota(3) = 1      BelongToClass: 2
   quota(4) = 7      BelongToClass: 2
   quota(5) = 1      BelongToClass: 2

10th BGf(N) = 5
   quota(1) = 5      BelongToClass: 3

11th BGf(N) = 3
   quota(1) = 5      BelongToClass: 3
   quota(2) = 3      BelongToClass: 3

12th BGf(N) = 8
   quota(1) = 5      BelongToClass: 3
   quota(2) = 3      BelongToClass: 3
   quota(3) = 7      BelongToClass: 3

13th BGf(N) = 7
   quota(1) = 4      BelongToClass: 3
   quota(2) = 3      BelongToClass: 3
   quota(3) = 6      BelongToClass: 3
   quota(4) = 6      BelongToClass: 3

14th BGf(N) = 7
   quota(1) = 5      BelongToClass: 3
   quota(2) = 4      BelongToClass: 3
   quota(3) = 6      BelongToClass: 3
   quota(4) = 7      BelongToClass: 3
   quota(5) = 7      BelongToClass: 3

15th BGf(N) = 8
   quota(1) = 8      BelongToClass: 4

16th BGf(N) = 6
   quota(1) = 4      BelongToClass: 4
   quota(2) = 6      BelongToClass: 4

17th BGf(N) = 7
   quota(1) = 6      BelongToClass: 4
   quota(2) = 5      BelongToClass: 4
   quota(3) = 7      BelongToClass: 4

18th BGf(N) = 6
   quota(1) = 4      BelongToClass: 4
   quota(2) = 5      BelongToClass: 4
   quota(3) = 5      BelongToClass: 4
   quota(4) = 6      BelongToClass: 4

19th BGf(N) = 5
   quota(1) = 2      BelongToClass: 4
   quota(2) = 4      BelongToClass: 4
   quota(3) = 3      BelongToClass: 4
   quota(4) = 4      BelongToClass: 4
   quota(5) = 5      BelongToClass: 4

20th BGf(N) = 8
   quota(1) = 7      BelongToClass: 5

21th BGf(N) = 8
   quota(1) = 7      BelongToClass: 5
   quota(2) = 7      BelongToClass: 5

```

22th	BGF(N) = 8	
	quota[1] = 7	BelongToClass: 5
	quota[2] = 7	BelongToClass: 5
	quota[3] = 7	BelongToClass: 5
23th	BGF(N) = 8	
	quota[1] = 7	BelongToClass: 5
	quota[2] = 7	BelongToClass: 5
	quota[3] = 7	BelongToClass: 5
	quota[4] = 7	BelongToClass: 5
24th	BGF(N) = 5	
	quota[1] = 5	BelongToClass: 7
25th	BGF(N) = 6	
	quota[1] = 5	BelongToClass: 7
	quota[2] = 6	BelongToClass: 7
26th	BGF(N) = 6	
	quota[1] = 5	BelongToClass: 7
	quota[2] = 7	BelongToClass: 7
	quota[3] = 6	BelongToClass: 7
27th	BGF(N) = 7	
	quota[1] = 5	BelongToClass: 7
	quota[2] = 5	BelongToClass: 7
	quota[3] = 6	BelongToClass: 7
	quota[4] = 7	BelongToClass: 7
28th	BGF(N) = 2	
	quota[1] = 2	BelongToClass: 8
29th	BGF(N) = 9	
	quota[1] = 1	BelongToClass: 8
	quota[2] = 8	BelongToClass: 8
30th	BGF(N) = 5	
	quota[1] = 5	BelongToClass: 9
31th	BGF(N) = 1	
	quota[1] = 6	BelongToClass: 9
	quota[2] = 0	BelongToClass: 9
32th	BGF(N) = 4	
	quota[1] = 6	BelongToClass: 9
	quota[2] = 1	BelongToClass: 9
	quota[3] = 4	BelongToClass: 9
33th	BGF(N) = 4	
	quota[1] = 4	BelongToClass: 9
	quota[2] = 1	BelongToClass: 9
	quota[3] = 3	BelongToClass: 9
	quota[4] = 4	BelongToClass: 9
34th	BGF(N) = 7	
	quota[1] = 5	BelongToClass: 9
	quota[2] = 1	BelongToClass: 9
	quota[3] = 4	BelongToClass: 9
	quota[4] = 3	BelongToClass: 9
	quota[5] = 7	BelongToClass: 9
35th	BGF(N) = 5	
	quota[1] = 5	BelongToClass: 10
36th	BGF(N) = 5	
	quota[1] = 5	BelongToClass: 10
	quota[2] = 5	BelongToClass: 10
37th	BGF(N) = 2	
	quota[1] = 2	BelongToClass: 10
	quota[2] = 2	BelongToClass: 10
	quota[3] = 2	BelongToClass: 10
38th	BGF(N) = 2	
	quota[1] = 6	BelongToClass: 10
	quota[2] = 5	BelongToClass: 10
	quota[3] = 1	BelongToClass: 10
	quota[4] = 2	BelongToClass: 10
39th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
40th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
	quota[2] = 0	BelongToClass: 11
41th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
	quota[2] = 0	BelongToClass: 11
	quota[3] = 0	BelongToClass: 11
42th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
	quota[2] = 0	BelongToClass: 11
	quota[3] = 0	BelongToClass: 11
	quota[4] = 0	BelongToClass: 11
43th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
	quota[2] = 0	BelongToClass: 11
	quota[3] = 0	BelongToClass: 11
	quota[4] = 0	BelongToClass: 11
	quota[5] = 0	BelongToClass: 11
44th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
	quota[2] = 0	BelongToClass: 11
	quota[3] = 0	BelongToClass: 11
	quota[4] = 0	BelongToClass: 11
	quota[5] = 0	BelongToClass: 11
	quota[6] = 0	BelongToClass: 11
45th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11
	quota[2] = 0	BelongToClass: 11
	quota[3] = 0	BelongToClass: 11
	quota[4] = 0	BelongToClass: 11
	quota[5] = 0	BelongToClass: 11
	quota[6] = 0	BelongToClass: 11
	quota[7] = 0	BelongToClass: 11
46th	BGF(N) = 0	
	quota[1] = 0	BelongToClass: 11

```

    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11
    quota[6] = 0      BelongToClass: 11
    quota[7] = 0      BelongToClass: 11
    quota[8] = 0      BelongToClass: 11

47th BG f(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11
    quota[6] = 0      BelongToClass: 11
    quota[7] = 0      BelongToClass: 11
    quota[8] = 0      BelongToClass: 11
    quota[9] = 0      BelongToClass: 11

48th BG f(N) = 4
    quota[1] = 4      BelongToClass: 12

49th BG f(N) = 4
    quota[1] = 4      BelongToClass: 12
    quota[2] = 4      BelongToClass: 12

50th BG f(N) = 4
    quota[1] = 4      BelongToClass: 12
    quota[2] = 4      BelongToClass: 12
    quota[3] = 4      BelongToClass: 12

51th BG f(N) = 4
    quota[1] = 4      BelongToClass: 12
    quota[2] = 4      BelongToClass: 12
    quota[3] = 4      BelongToClass: 12
    quota[4] = 4      BelongToClass: 12

52th BG f(N) = 4
    quota[1] = 4      BelongToClass: 12
    quota[2] = 4      BelongToClass: 12
    quota[3] = 4      BelongToClass: 12
    quota[4] = 4      BelongToClass: 12
    quota[5] = 4      BelongToClass: 12

53th BG f(N) = 4
    quota[1] = 4      BelongToClass: 12
    quota[2] = 4      BelongToClass: 12
    quota[3] = 4      BelongToClass: 12
    quota[4] = 4      BelongToClass: 12
    quota[5] = 4      BelongToClass: 12
    quota[6] = 4      BelongToClass: 12

54th BG f(N) = 9
    quota[1] = 3      BelongToClass: 12
    quota[2] = 3      BelongToClass: 12
    quota[3] = 3      BelongToClass: 12
    quota[4] = 3      BelongToClass: 12
    quota[5] = 3      BelongToClass: 12
    quota[6] = 3      BelongToClass: 12
    quota[7] = 8      BelongToClass: 12

55th BG f(N) = 1
    quota[1] = 1      BelongToClass: 12
    quota[2] = 1      BelongToClass: 12
    quota[3] = 1      BelongToClass: 12
    quota[4] = 1      BelongToClass: 12
    quota[5] = 1      BelongToClass: 12
    quota[6] = 1      BelongToClass: 12
    quota[7] = 7      BelongToClass: 12
    quota[8] = 1      BelongToClass: 12

-----
Belong to 11th group, the quota cost is 0
-----
h2_3.tmp belongs to .....
The match result of the lowest cost f(N), and quota
0th BG f(N) = 7
    quota[1] = 6      BelongToClass: 1

1th BG f(N) = 3
    quota[1] = 1      BelongToClass: 1
    quota[2] = 3      BelongToClass: 1

2th BG f(N) = 7
    quota[1] = 7      BelongToClass: 1
    quota[2] = 1      BelongToClass: 1
    quota[3] = 6      BelongToClass: 1

3th BG f(N) = 0
    quota[1] = 7      BelongToClass: 1
    quota[2] = 4      BelongToClass: 1
    quota[3] = 7      BelongToClass: 1
    quota[4] = 0      BelongToClass: 1

4th BG f(N) = 5
    quota[1] = 1      BelongToClass: 1
    quota[2] = 4      BelongToClass: 1
    quota[3] = 1      BelongToClass: 1
    quota[4] = 1      BelongToClass: 1
    quota[5] = 5      BelongToClass: 1

5th BG f(N) = 1
    quota[1] = 1      BelongToClass: 2

6th BG f(N) = 2
    quota[1] = 1      BelongToClass: 2
    quota[2] = 2      BelongToClass: 2

7th BG f(N) = 2
    quota[1] = 1      BelongToClass: 2
    quota[2] = 2      BelongToClass: 2
    quota[3] = 2      BelongToClass: 2

8th BG f(N) = 6
    quota[1] = 1      BelongToClass: 2
    quota[2] = 1      BelongToClass: 2
    quota[3] = 1      BelongToClass: 2
    quota[4] = 6      BelongToClass: 2

9th BG f(N) = 2
    quota[1] = 2      BelongToClass: 2
    quota[2] = 0      BelongToClass: 2
    quota[3] = 0      BelongToClass: 2
    quota[4] = 6      BelongToClass: 2

```



```

quota[3] = 2      BelongToClass: 12
quota[4] = 2      BelongToClass: 12
quota[5] = 2      BelongToClass: 12
quota[6] = 2      BelongToClass: 12
quota[7] = 8      BelongToClass: 12
quota[8] = 0      BelongToClass: 12

```

Belong to 12th group, the quota cost is 0

h_j.bmp belongs to

The match result of the lowest cost f(N), and quota

```

0th BG f(N) = 16
  quota[1] = 15      BelongToClass: 1
1th BG f(N) = 12
  quota[1] = 8       BelongToClass: 1
  quota[2] = 12      BelongToClass: 1
2th BG f(N) = 16
  quota[1] = 16      BelongToClass: 1
  quota[2] = 12      BelongToClass: 1
  quota[3] = 15      BelongToClass: 1
3th BG f(N) = 9
  quota[1] = 2       BelongToClass: 1
  quota[2] = 5       BelongToClass: 1
  quota[3] = 2       BelongToClass: 1
  quota[4] = 9       BelongToClass: 1
4th BG f(N) = 13
  quota[1] = 9       BelongToClass: 1
  quota[2] = 12      BelongToClass: 1
  quota[3] = 9       BelongToClass: 1
  quota[4] = 9       BelongToClass: 1
  quota[5] = 13      BelongToClass: 1
5th BG f(N) = 10
  quota[1] = 10      BelongToClass: 2
6th BG f(N) = 8
  quota[1] = 5       BelongToClass: 2
  quota[2] = 8       BelongToClass: 2
7th BG f(N) = 8
  quota[1] = 5       BelongToClass: 2
  quota[2] = 8       BelongToClass: 2
  quota[3] = 8       BelongToClass: 2
8th BG f(N) = 14
  quota[1] = 9       BelongToClass: 2
  quota[2] = 7       BelongToClass: 2
  quota[3] = 7       BelongToClass: 2
  quota[4] = 14      BelongToClass: 2
9th BG f(N) = 7
  quota[1] = 3       BelongToClass: 2
  quota[2] = 5       BelongToClass: 2
  quota[3] = 5       BelongToClass: 2
  quota[4] = 1       BelongToClass: 2
  quota[5] = 7       BelongToClass: 2
10th BGf(N) = 3
  quota[1] = 3       BelongToClass: 3
11th BGf(N) = 11
  quota[1] = 3       BelongToClass: 3
  quota[2] = 11      BelongToClass: 3
12th BGf(N) = 1
  quota[1] = 2       BelongToClass: 3
  quota[2] = 10      BelongToClass: 3
  quota[3] = 0       BelongToClass: 3
13th BGf(N) = 1
  quota[1] = 2       BelongToClass: 3
  quota[2] = 9       BelongToClass: 3
  quota[3] = 0       BelongToClass: 3
  quota[4] = 0       BelongToClass: 3
14th BGf(N) = 1
  quota[1] = 1       BelongToClass: 3
  quota[2] = 10      BelongToClass: 3
  quota[3] = 0       BelongToClass: 3
  quota[4] = 1       BelongToClass: 3
  quota[5] = 1       BelongToClass: 3
15th BGf(N) = 0
  quota[1] = 0       BelongToClass: 4
16th BGf(N) = 3
  quota[1] = 1       BelongToClass: 4
  quota[2] = 3       BelongToClass: 4
17th BGf(N) = 1
  quota[1] = 0       BelongToClass: 4
  quota[2] = 1       BelongToClass: 4
  quota[3] = 1       BelongToClass: 4
18th BGf(N) = 3
  quota[1] = 1       BelongToClass: 4
  quota[2] = 2       BelongToClass: 4
  quota[3] = 2       BelongToClass: 4
  quota[4] = 1       BelongToClass: 4
19th BGf(N) = 4
  quota[1] = 1       BelongToClass: 4
  quota[2] = 3       BelongToClass: 4
  quota[3] = 2       BelongToClass: 4
  quota[4] = 3       BelongToClass: 4
  quota[5] = 4       BelongToClass: 4
20th BGf(N) = 16
  quota[1] = 15      BelongToClass: 5
21th BGf(N) = 16
  quota[1] = 15      BelongToClass: 5
  quota[2] = 15      BelongToClass: 5
22th BGf(N) = 16
  quota[1] = 15      BelongToClass: 5
  quota[2] = 15      BelongToClass: 5
  quota[3] = 15      BelongToClass: 5

```

23th BGf(N) = 16
 quota(1) = 15 BelongToClass: 5
 quota(2) = 15 BelongToClass: 5
 quota(3) = 15 BelongToClass: 5
 quota(4) = 15 BelongToClass: 5

24th BGf(N) = 3
 quota(1) = 3 BelongToClass: 7

25th BGf(N) = 14
 quota(1) = 3 BelongToClass: 7
 quota(2) = 14 BelongToClass: 7

26th BGf(N) = 3
 quota(1) = 2 BelongToClass: 7
 quota(2) = 10 BelongToClass: 7
 quota(3) = 3 BelongToClass: 7

27th BGf(N) = 15
 quota(1) = 3 BelongToClass: 7
 quota(2) = 13 BelongToClass: 7
 quota(3) = 2 BelongToClass: 7
 quota(4) = 15 BelongToClass: 7

28th BGf(N) = 7
 quota(1) = 7 BelongToClass: 8

29th BGf(N) = 16
 quota(1) = 6 BelongToClass: 8
 quota(2) = 15 BelongToClass: 8

30th BGf(N) = 13
 quota(1) = 13 BelongToClass: 9

31th BGf(N) = 7
 quota(1) = 0 BelongToClass: 9
 quota(2) = 6 BelongToClass: 9

32th BGf(N) = 4
 quota(1) = 6 BelongToClass: 9
 quota(2) = 1 BelongToClass: 9
 quota(3) = 4 BelongToClass: 9

33th BGf(N) = 5
 quota(1) = 3 BelongToClass: 9
 quota(2) = 2 BelongToClass: 9
 quota(3) = 4 BelongToClass: 9
 quota(4) = 5 BelongToClass: 9

34th BGf(N) = 16
 quota(1) = 14 BelongToClass: 9
 quota(2) = 8 BelongToClass: 9
 quota(3) = 5 BelongToClass: 9
 quota(4) = 6 BelongToClass: 9
 quota(5) = 16 BelongToClass: 9

35th BGf(N) = 3
 quota(1) = 3 BelongToClass: 10

36th BGf(N) = 4
 quota(1) = 4 BelongToClass: 10
 quota(2) = 4 BelongToClass: 10

37th BGf(N) = 7
 quota(1) = 3 BelongToClass: 10
 quota(2) = 3 BelongToClass: 10
 quota(3) = 7 BelongToClass: 10

38th BGf(N) = 10
 quota(1) = 2 BelongToClass: 10
 quota(2) = 3 BelongToClass: 10
 quota(3) = 7 BelongToClass: 10
 quota(4) = 10 BelongToClass: 10

39th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11

40th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11

41th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11

42th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11

43th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11

44th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11
 quota(6) = 8 BelongToClass: 11

45th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11
 quota(6) = 8 BelongToClass: 11
 quota(7) = 8 BelongToClass: 11

46th BGf(N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11
 quota(6) = 8 BelongToClass: 11

```

    quota(7) = 8      BelongToClass: 11
    quota(8) = 8      BelongToClass: 11
47th BGF(N) = 8
    quota(1) = 8      BelongToClass: 11
    quota(2) = 8      BelongToClass: 11
    quota(3) = 8      BelongToClass: 11
    quota(4) = 8      BelongToClass: 11
    quota(5) = 8      BelongToClass: 11
    quota(6) = 8      BelongToClass: 11
    quota(7) = 8      BelongToClass: 11
    quota(8) = 8      BelongToClass: 11
    quota(9) = 8      BelongToClass: 11
48th BGF(N) = 12
    quota(1) = 12     BelongToClass: 12
49th BGF(N) = 12
    quota(1) = 12     BelongToClass: 12
    quota(2) = 12     BelongToClass: 12
50th BGF(N) = 12
    quota(1) = 12     BelongToClass: 12
    quota(2) = 12     BelongToClass: 12
    quota(3) = 12     BelongToClass: 12
51th BGF(N) = 12
    quota(1) = 12     BelongToClass: 12
    quota(2) = 12     BelongToClass: 12
    quota(3) = 12     BelongToClass: 12
    quota(4) = 12     BelongToClass: 12
52th BGF(N) = 12
    quota(1) = 12     BelongToClass: 12
    quota(2) = 12     BelongToClass: 12
    quota(3) = 12     BelongToClass: 12
    quota(4) = 12     BelongToClass: 12
    quota(5) = 12     BelongToClass: 12
53th BGF(N) = 12
    quota(1) = 12     BelongToClass: 12
    quota(2) = 12     BelongToClass: 12
    quota(3) = 12     BelongToClass: 12
    quota(4) = 12     BelongToClass: 12
    quota(5) = 12     BelongToClass: 12
    quota(6) = 12     BelongToClass: 12
54th BGF(N) = 16
    quota(1) = 10     BelongToClass: 12
    quota(2) = 10     BelongToClass: 12
    quota(3) = 10     BelongToClass: 12
    quota(4) = 10     BelongToClass: 12
    quota(5) = 10     BelongToClass: 12
    quota(6) = 10     BelongToClass: 12
    quota(7) = 15     BelongToClass: 12
55th BGF(N) = 9
    quota(1) = 7      BelongToClass: 12
    quota(2) = 7      BelongToClass: 12
    quota(3) = 7      BelongToClass: 12
    quota(4) = 7      BelongToClass: 12
    quota(5) = 7      BelongToClass: 12
    quota(6) = 7      BelongToClass: 12
    quota(7) = 1      BelongToClass: 12
    quota(8) = 9      BelongToClass: 12

```

Belong to 9th group, the quota cost is 0

h4_3.bmp belongs to
The match result of the lowest cost f(N), and quota

```

0th BG f(N) = 16
    quota(1) = 15     BelongToClass: 1
1th BG f(N) = 13
    quota(1) = 9      BelongToClass: 1
    quota(2) = 13     BelongToClass: 1
2th BG f(N) = 16
    quota(1) = 16     BelongToClass: 1
    quota(2) = 12     BelongToClass: 1
    quota(3) = 15     BelongToClass: 1
3th BG f(N) = 9
    quota(1) = 2      BelongToClass: 1
    quota(2) = 5      BelongToClass: 1
    quota(3) = 2      BelongToClass: 1
    quota(4) = 9      BelongToClass: 1
4th BG f(N) = 11
    quota(1) = 9      BelongToClass: 1
    quota(2) = 12     BelongToClass: 1
    quota(3) = 9      BelongToClass: 1
    quota(4) = 9      BelongToClass: 1
    quota(5) = 11     BelongToClass: 1
5th BG f(N) = 10
    quota(1) = 10     BelongToClass: 2
6th BG f(N) = 8
    quota(1) = 5      BelongToClass: 2
    quota(2) = 8      BelongToClass: 2
7th BG f(N) = 8
    quota(1) = 5      BelongToClass: 2
    quota(2) = 8      BelongToClass: 2
    quota(3) = 8      BelongToClass: 2
8th BG f(N) = 14
    quota(1) = 9      BelongToClass: 2
    quota(2) = 7      BelongToClass: 2
    quota(3) = 7      BelongToClass: 2
    quota(4) = 14     BelongToClass: 2
9th BG f(N) = 7
    quota(1) = 3      BelongToClass: 2
    quota(2) = 5      BelongToClass: 2
    quota(3) = 5      BelongToClass: 2
    quota(4) = 3      BelongToClass: 2
    quota(5) = 7      BelongToClass: 2
10th BGF(N) = 3
    quota(1) = 3      BelongToClass: 1

```

1th BGF(N) = 11		
quota[1] = 3	BelongToClass: 3	
quota[2] = 11	BelongToClass: 3	
2th BGF(N) = 1		
quota[1] = 2	BelongToClass: 3	
quota[2] = 10	BelongToClass: 3	
quota[3] = 0	BelongToClass: 3	
13th BGF(N) = 1		
quota[1] = 2	BelongToClass: 3	
quota[2] = 9	BelongToClass: 3	
quota[3] = 0	BelongToClass: 3	
quota[4] = 0	BelongToClass: 3	
14th BGF(N) = 1		
quota[1] = 1	BelongToClass: 3	
quota[2] = 10	BelongToClass: 3	
quota[3] = 0	BelongToClass: 3	
quota[4] = 1	BelongToClass: 3	
quota[5] = 1	BelongToClass: 3	
15th BGF(N) = 0		
quota[1] = 0	BelongToClass: 4	
16th BGF(N) = 2		
quota[1] = 0	BelongToClass: 4	
quota[2] = 2	BelongToClass: 4	
17th BGF(N) = 1		
quota[1] = 0	BelongToClass: 4	
quota[2] = 1	BelongToClass: 4	
quota[3] = 1	BelongToClass: 4	
18th BGF(N) = 2		
quota[1] = 0	BelongToClass: 4	
quota[2] = 1	BelongToClass: 4	
quota[3] = 1	BelongToClass: 4	
quota[4] = 2	BelongToClass: 4	
19th BGF(N) = 3		
quota[1] = 0	BelongToClass: 4	
quota[2] = 2	BelongToClass: 4	
quota[3] = 1	BelongToClass: 4	
quota[4] = 1	BelongToClass: 4	
quota[5] = 3	BelongToClass: 4	
20th BGF(N) = 16		
quota[1] = 15	BelongToClass: 5	
21th BGF(N) = 16		
quota[1] = 15	BelongToClass: 5	
quota[2] = 15	BelongToClass: 5	
22th BGF(N) = 16		
quota[1] = 15	BelongToClass: 5	
quota[2] = 15	BelongToClass: 5	
quota[3] = 15	BelongToClass: 5	
23th BGF(N) = 16		
quota[1] = 15	BelongToClass: 5	
quota[2] = 15	BelongToClass: 5	
quota[3] = 15	BelongToClass: 5	
quota[4] = 15	BelongToClass: 5	
24th BGF(N) = 3		
quota[1] = 3	BelongToClass: 7	
25th BGF(N) = 14		
quota[1] = 3	BelongToClass: 7	
quota[2] = 14	BelongToClass: 7	
26th BGF(N) = 2		
quota[1] = 1	BelongToClass: 7	
quota[2] = 11	BelongToClass: 7	
quota[3] = 2	BelongToClass: 7	
27th BGF(N) = 15		
quota[1] = 3	BelongToClass: 7	
quota[2] = 11	BelongToClass: 7	
quota[3] = 2	BelongToClass: 7	
quota[4] = 15	BelongToClass: 7	
28th BGF(N) = 6		
quota[1] = 6	BelongToClass: 8	
29th BGF(N) = 16		
quota[1] = 6	BelongToClass: 8	
quota[2] = 15	BelongToClass: 8	
30th BGF(N) = 13		
quota[1] = 13	BelongToClass: 9	
31th BGF(N) = 7		
quota[1] = 0	BelongToClass: 9	
quota[2] = 6	BelongToClass: 9	
32th BGF(N) = 5		
quota[1] = 5	BelongToClass: 9	
quota[2] = 2	BelongToClass: 9	
quota[3] = 5	BelongToClass: 9	
33th BGF(N) = 6		
quota[1] = 2	BelongToClass: 9	
quota[2] = 3	BelongToClass: 9	
quota[3] = 5	BelongToClass: 9	
quota[4] = 6	BelongToClass: 9	
34th BGF(N) = 15		
quota[1] = 13	BelongToClass: 9	
quota[2] = 7	BelongToClass: 9	
quota[3] = 4	BelongToClass: 9	
quota[4] = 5	BelongToClass: 9	
quota[5] = 15	BelongToClass: 9	
35th BGF(N) = 3		
quota[1] = 3	BelongToClass: 10	
36th BGF(N) = 4		
quota[1] = 4	BelongToClass: 10	
quota[2] = 4	BelongToClass: 10	
37th BGF(N) = 7		

```

        quota[1] = 3      BelongToClass: 10
        quota[2] = 3      BelongToClass: 10
        quota[3] = 7      BelongToClass: 10

38th BGF(N) = 10
        quota[1] = 2      BelongToClass: 10
        quota[2] = 3      BelongToClass: 10
        quota[3] = 7      BelongToClass: 10
        quota[4] = 10     BelongToClass: 10

39th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11

40th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11

41th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11

42th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11

43th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11
        quota[5] = 8      BelongToClass: 11

44th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11
        quota[5] = 8      BelongToClass: 11
        quota[6] = 8      BelongToClass: 11

45th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11
        quota[5] = 8      BelongToClass: 11
        quota[6] = 8      BelongToClass: 11
        quota[7] = 8      BelongToClass: 11

46th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11
        quota[5] = 8      BelongToClass: 11
        quota[6] = 8      BelongToClass: 11
        quota[7] = 8      BelongToClass: 11
        quota[8] = 8      BelongToClass: 11

47th BGF(N) = 8
        quota[1] = 8      BelongToClass: 11
        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11
        quota[5] = 8      BelongToClass: 11
        quota[6] = 8      BelongToClass: 11
        quota[7] = 8      BelongToClass: 11
        quota[8] = 8      BelongToClass: 11
        quota[9] = 8      BelongToClass: 11

48th BGF(N) = 12
        quota[1] = 12     BelongToClass: 12

49th BGF(N) = 12
        quota[1] = 12     BelongToClass: 12
        quota[2] = 12     BelongToClass: 12

50th BGF(N) = 12
        quota[1] = 12     BelongToClass: 12
        quota[2] = 12     BelongToClass: 12
        quota[3] = 12     BelongToClass: 12

51th BGF(N) = 12
        quota[1] = 12     BelongToClass: 12
        quota[2] = 12     BelongToClass: 12
        quota[3] = 12     BelongToClass: 12
        quota[4] = 12     BelongToClass: 12

52th BGF(N) = 12
        quota[1] = 12     BelongToClass: 12
        quota[2] = 12     BelongToClass: 12
        quota[3] = 12     BelongToClass: 12
        quota[4] = 12     BelongToClass: 12
        quota[5] = 12     BelongToClass: 12

53th BGF(N) = 12
        quota[1] = 12     BelongToClass: 12
        quota[2] = 12     BelongToClass: 12
        quota[3] = 12     BelongToClass: 12
        quota[4] = 12     BelongToClass: 12
        quota[5] = 12     BelongToClass: 12
        quota[6] = 12     BelongToClass: 12

54th BGF(N) = 16
        quota[1] = 10     BelongToClass: 12
        quota[2] = 10     BelongToClass: 12
        quota[3] = 10     BelongToClass: 12
        quota[4] = 10     BelongToClass: 12
        quota[5] = 10     BelongToClass: 12
        quota[6] = 10     BelongToClass: 12
        quota[7] = 15     BelongToClass: 12

55th BGF(N) = 9
        quota[1] = 7      BelongToClass: 12
        quota[2] = 7      BelongToClass: 12
        quota[3] = 7      BelongToClass: 12
        quota[4] = 7      BelongToClass: 12
        quota[5] = 7      BelongToClass: 12
        quota[6] = 7      BelongToClass: 12
        quota[7] = 1      BelongToClass: 12

```

```

        quota[8] = 9          BelongToClass: 12
Belong to 9th group, the quota cost is 0
-----
As 1.bmp belongs to .....
The match result of the lowest cost f(N), and quota
0th BG f(N) = 1
    quota[1] = 0          BelongToClass: 1
1st BG f(N) = 4
    quota[1] = 0          BelongToClass: 1
    quota[2] = 4          BelongToClass: 1
2nd BG f(N) = 0
    quota[1] = 0          BelongToClass: 1
    quota[2] = 4          BelongToClass: 1
    quota[3] = 1          BelongToClass: 1
3rd BG f(N) = 7
    quota[1] = 0          BelongToClass: 1
    quota[2] = 3          BelongToClass: 1
    quota[3] = 0          BelongToClass: 1
    quota[4] = 7          BelongToClass: 1
4th BG f(N) = 3
    quota[1] = 1          BelongToClass: 1
    quota[2] = 2          BelongToClass: 1
    quota[3] = 1          BelongToClass: 1
    quota[4] = 1          BelongToClass: 1
    quota[5] = 3          BelongToClass: 1
5th BG f(N) = 6
    quota[1] = 6          BelongToClass: 2
6th BG f(N) = 8
    quota[1] = 5          BelongToClass: 2
    quota[2] = 8          BelongToClass: 2
7th BG f(N) = 0
    quota[1] = 5          BelongToClass: 2
    quota[2] = 8          BelongToClass: 2
    quota[3] = 8          BelongToClass: 2
8th BG f(N) = 3
    quota[1] = 2          BelongToClass: 2
    quota[2] = 4          BelongToClass: 2
    quota[3] = 4          BelongToClass: 2
    quota[4] = 3          BelongToClass: 2
9th BG f(N) = 9
    quota[1] = 5          BelongToClass: 2
    quota[2] = 7          BelongToClass: 2
    quota[3] = 7          BelongToClass: 2
    quota[4] = 1          BelongToClass: 2
    quota[5] = 9          BelongToClass: 2
10th BGf(N) = 13
    quota[1] = 13         BelongToClass: 3
11th BGf(N) = 5
    quota[1] = 3          BelongToClass: 3
    quota[2] = 5          BelongToClass: 3
12th BGf(N) = 16
    quota[1] = 13         BelongToClass: 3
    quota[2] = 5          BelongToClass: 3
    quota[3] = 15         BelongToClass: 3
13th BGf(N) = 15
    quota[1] = 12         BelongToClass: 3
    quota[2] = 5          BelongToClass: 3
    quota[3] = 14         BelongToClass: 3
    quota[4] = 14         BelongToClass: 3
14th BGf(N) = 15
    quota[1] = 13         BelongToClass: 3
    quota[2] = 4          BelongToClass: 3
    quota[3] = 14         BelongToClass: 3
    quota[4] = 15         BelongToClass: 3
    quota[5] = 15         BelongToClass: 3
15th BGf(N) = 16
    quota[1] = 16         BelongToClass: 4
16th BGf(N) = 14
    quota[1] = 12         BelongToClass: 4
    quota[2] = 14         BelongToClass: 4
17th BGf(N) = 15
    quota[1] = 14         BelongToClass: 4
    quota[2] = 13         BelongToClass: 4
    quota[3] = 15         BelongToClass: 4
18th BGf(N) = 14
    quota[1] = 12         BelongToClass: 4
    quota[2] = 13         BelongToClass: 4
    quota[3] = 13         BelongToClass: 4
    quota[4] = 14         BelongToClass: 4
19th BGf(N) = 13
    quota[1] = 10         BelongToClass: 4
    quota[2] = 12         BelongToClass: 4
    quota[3] = 11         BelongToClass: 4
    quota[4] = 12         BelongToClass: 4
    quota[5] = 13         BelongToClass: 4
20th BGf(N) = 1
    quota[1] = 0          BelongToClass: 5
21th BGf(N) = 1
    quota[1] = 0          BelongToClass: 5
    quota[2] = 0          BelongToClass: 5
22th BGf(N) = 1
    quota[1] = 0          BelongToClass: 5
    quota[2] = 0          BelongToClass: 5
    quota[3] = 0          BelongToClass: 5
23th BGf(N) = 1
    quota[1] = 0          BelongToClass: 5
    quota[2] = 0          BelongToClass: 5
    quota[3] = 0          BelongToClass: 5
    quota[4] = 0          BelongToClass: 5

```

24th BGF (N) = 13
 quota(1) = 13 BelongToClass: 7

25th BGF (N) = 2
 quota(1) = 9 BelongToClass: 7
 quota(2) = 2 BelongToClass: 7

26th BGF (N) = 14
 quota(1) = 13 BelongToClass: 7
 quota(2) = 1 BelongToClass: 7
 quota(3) = 14 BelongToClass: 7

27th BGF (N) = 2
 quota(1) = 10 BelongToClass: 7
 quota(2) = 0 BelongToClass: 7
 quota(3) = 11 BelongToClass: 7
 quota(4) = 2 BelongToClass: 7

28th BGF (N) = 10
 quota(1) = 10 BelongToClass: 8

29th BGF (N) = 0
 quota(1) = 10 BelongToClass: 8
 quota(2) = 1 BelongToClass: 8

30th BGF (N) = 3
 quota(1) = 3 BelongToClass: 9

31th BGF (N) = 9
 quota(1) = 2 BelongToClass: 9
 quota(2) = 8 BelongToClass: 9

32th BGF (N) = 12
 quota(1) = 2 BelongToClass: 9
 quota(2) = 9 BelongToClass: 9
 quota(3) = 12 BelongToClass: 9

33th BGF (N) = 11
 quota(1) = 3 BelongToClass: 9
 quota(2) = 8 BelongToClass: 9
 quota(3) = 10 BelongToClass: 9
 quota(4) = 11 BelongToClass: 9

34th BGF (N) = 1
 quota(1) = 1 BelongToClass: 9
 quota(2) = 7 BelongToClass: 9
 quota(3) = 10 BelongToClass: 9
 quota(4) = 9 BelongToClass: 9
 quota(5) = 1 BelongToClass: 9

35th BGF (N) = 13
 quota(1) = 13 BelongToClass: 10

36th BGF (N) = 13
 quota(1) = 13 BelongToClass: 10
 quota(2) = 13 BelongToClass: 10

37th BGF (N) = 9
 quota(1) = 5 BelongToClass: 10
 quota(2) = 5 BelongToClass: 10
 quota(3) = 9 BelongToClass: 10

38th BGF (N) = 6
 quota(1) = 2 BelongToClass: 10
 quota(2) = 1 BelongToClass: 10
 quota(3) = 1 BelongToClass: 10
 quota(4) = 6 BelongToClass: 10

39th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11

40th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11

41th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11

42th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11

43th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11

44th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11
 quota(6) = 8 BelongToClass: 11

45th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 8 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11
 quota(6) = 8 BelongToClass: 11
 quota(7) = 8 BelongToClass: 11

46th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11
 quota(2) = 8 BelongToClass: 11
 quota(3) = 9 BelongToClass: 11
 quota(4) = 8 BelongToClass: 11
 quota(5) = 8 BelongToClass: 11
 quota(6) = 8 BelongToClass: 11
 quota(7) = 9 BelongToClass: 11
 quota(8) = 8 BelongToClass: 11

47th BGF (N) = 8
 quota(1) = 8 BelongToClass: 11

```

        quota[2] = 8      BelongToClass: 11
        quota[3] = 8      BelongToClass: 11
        quota[4] = 8      BelongToClass: 11
        quota[5] = 8      BelongToClass: 11
        quota[6] = 8      BelongToClass: 11
        quota[7] = 8      BelongToClass: 11
        quota[8] = 8      BelongToClass: 11
        quota[9] = 8      BelongToClass: 11

49th BG f(N) = 5
        quota[1] = 5      BelongToClass: 12

49th BG f(N) = 5
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12

50th BG f(N) = 5
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12
        quota[3] = 5      BelongToClass: 12

51th BG f(N) = 5
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12
        quota[3] = 5      BelongToClass: 12
        quota[4] = 5      BelongToClass: 12

52th BG f(N) = 5
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12
        quota[3] = 5      BelongToClass: 12
        quota[4] = 5      BelongToClass: 12
        quota[5] = 5      BelongToClass: 12

53th BG f(N) = 5
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12
        quota[3] = 5      BelongToClass: 12
        quota[4] = 5      BelongToClass: 12
        quota[5] = 5      BelongToClass: 12
        quota[6] = 5      BelongToClass: 12

54th BG f(N) = 0
        quota[1] = 6      BelongToClass: 12
        quota[2] = 6      BelongToClass: 12
        quota[3] = 6      BelongToClass: 12
        quota[4] = 6      BelongToClass: 12
        quota[5] = 6      BelongToClass: 12
        quota[6] = 6      BelongToClass: 12
        quota[7] = 1      BelongToClass: 12

55th BG f(N) = 7
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12
        quota[3] = 5      BelongToClass: 12
        quota[4] = 5      BelongToClass: 12
        quota[5] = 5      BelongToClass: 12
        quota[6] = 5      BelongToClass: 12
        quota[7] = 1      BelongToClass: 12
        quota[8] = 7      BelongToClass: 12

```

Belong to 7th group, the quota cost is 0

h9_.bnp belongs to

The match result of the lowest cost f(N), and quota

```

0th BG f(N) = 1
        quota[1] = 0      BelongToClass: 1

1th BG f(N) = 4
        quota[1] = 0      BelongToClass: 1
        quota[2] = 4      BelongToClass: 1

2th BG f(N) = 1
        quota[1] = 1      BelongToClass: 1
        quota[2] = 3      BelongToClass: 1
        quota[3] = 0      BelongToClass: 1

3th BG f(N) = 7
        quota[1] = 0      BelongToClass: 1
        quota[2] = 1      BelongToClass: 1
        quota[3] = 0      BelongToClass: 1
        quota[4] = 7      BelongToClass: 1

4th BG f(N) = 3
        quota[1] = 1      BelongToClass: 1
        quota[2] = 2      BelongToClass: 1
        quota[3] = 1      BelongToClass: 1
        quota[4] = 1      BelongToClass: 1
        quota[5] = 1      BelongToClass: 1

5th BG f(N) = 7
        quota[1] = 7      BelongToClass: 2

6th BG f(N) = 9
        quota[1] = 6      BelongToClass: 2
        quota[2] = 9      BelongToClass: 2

7th BG f(N) = 9
        quota[1] = 6      BelongToClass: 2
        quota[2] = 9      BelongToClass: 2
        quota[3] = 9      BelongToClass: 2

8th BG f(N) = 2
        quota[1] = 3      BelongToClass: 2
        quota[2] = 5      BelongToClass: 2
        quota[3] = 5      BelongToClass: 2
        quota[4] = 2      BelongToClass: 2

9th BG f(N) = 10
        quota[1] = 6      BelongToClass: 2
        quota[2] = 8      BelongToClass: 2
        quota[3] = 8      BelongToClass: 2
        quota[4] = 2      BelongToClass: 2
        quota[5] = 10     BelongToClass: 2

10th BG f(N) = 14
        quota[1] = 14     BelongToClass: 3

11th BG f(N) = 5
        quota[1] = 3      BelongToClass: 3
        quota[2] = 5      BelongToClass: 3

12th BG f(N) = 14

```


quota[1] = 13	BelongToClass: 3
quota[2] = 5	BelongToClass: 3
quota[3] = 15	BelongToClass: 3
13th BGF(N) = 15	
quota[1] = 12	BelongToClass: 3
quota[2] = 5	BelongToClass: 3
quota[3] = 14	BelongToClass: 3
quota[4] = 14	BelongToClass: 3
14th BGF(N) = 15	
quota[1] = 13	BelongToClass: 3
quota[2] = 4	BelongToClass: 3
quota[3] = 14	BelongToClass: 3
quota[4] = 15	BelongToClass: 3
quota[5] = 15	BelongToClass: 3
15th BGF(N) = 16	
quota[1] = 16	BelongToClass: 4
16th BGF(N) = 14	
quota[1] = 12	BelongToClass: 4
quota[2] = 14	BelongToClass: 4
17th BGF(N) = 15	
quota[1] = 14	BelongToClass: 4
quota[2] = 13	BelongToClass: 4
quota[3] = 15	BelongToClass: 4
18th BGF(N) = 14	
quota[1] = 12	BelongToClass: 4
quota[2] = 13	BelongToClass: 4
quota[3] = 13	BelongToClass: 4
quota[4] = 14	BelongToClass: 4
19th BGF(N) = 14	
quota[1] = 11	BelongToClass: 4
quota[2] = 13	BelongToClass: 4
quota[3] = 12	BelongToClass: 4
quota[4] = 13	BelongToClass: 4
quota[5] = 14	BelongToClass: 4
20th BGF(N) = 0	
quota[1] = 1	BelongToClass: 5
21th BGF(N) = 0	
quota[1] = 1	BelongToClass: 5
quota[2] = 1	BelongToClass: 5
22th BGF(N) = 0	
quota[1] = 1	BelongToClass: 5
quota[2] = 1	BelongToClass: 5
quota[3] = 1	BelongToClass: 5
23th BGF(N) = 0	
quota[1] = 1	BelongToClass: 5
quota[2] = 1	BelongToClass: 5
quota[3] = 1	BelongToClass: 5
quota[4] = 1	BelongToClass: 5
24th BGF(N) = 13	
quota[1] = 13	BelongToClass: 7
25th BGF(N) = 3	
quota[1] = 8	BelongToClass: 7
quota[2] = 3	BelongToClass: 7
26th BGF(N) = 14	
quota[1] = 13	BelongToClass: 7
quota[2] = 1	BelongToClass: 7
quota[3] = 14	BelongToClass: 7
27th BGF(N) = 1	
quota[1] = 11	BelongToClass: 7
quota[2] = 1	BelongToClass: 7
quota[3] = 12	BelongToClass: 7
quota[4] = 1	BelongToClass: 7
28th BGF(N) = 10	
quota[1] = 10	BelongToClass: 8
29th BGF(N) = 1	
quota[1] = 9	BelongToClass: 8
quota[2] = 0	BelongToClass: 8
30th BGF(N) = 4	
quota[1] = 4	BelongToClass: 9
31th BGF(N) = 10	
quota[1] = 3	BelongToClass: 9
quota[2] = 9	BelongToClass: 9
32th BGF(N) = 12	
quota[1] = 2	BelongToClass: 9
quota[2] = 9	BelongToClass: 9
quota[3] = 12	BelongToClass: 9
33th BGF(N) = 11	
quota[1] = 3	BelongToClass: 9
quota[2] = 8	BelongToClass: 9
quota[3] = 10	BelongToClass: 9
quota[4] = 11	BelongToClass: 9
34th BGF(N) = 2	
quota[1] = 0	BelongToClass: 9
quota[2] = 5	BelongToClass: 9
quota[3] = 9	BelongToClass: 9
quota[4] = 8	BelongToClass: 9
quota[5] = 2	BelongToClass: 9
35th BGF(N) = 13	
quota[1] = 13	BelongToClass: 10
36th BGF(N) = 13	
quota[1] = 13	BelongToClass: 10
quota[2] = 13	BelongToClass: 10
37th BGF(N) = 9	
quota[1] = 5	BelongToClass: 10
quota[2] = 5	BelongToClass: 10
quota[3] = 9	BelongToClass: 10
38th BGF(N) = 6	

```

    quota[1] = 2      BelongToClass: 10
    quota[2] = 1      BelongToClass: 10
    quota[3] = 3      BelongToClass: 10
    quota[4] = 6      BelongToClass: 10

39th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11

40th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11

41th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11

42th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11
    quota[4] = 8      BelongToClass: 11

43th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11
    quota[4] = 8      BelongToClass: 11
    quota[5] = 8      BelongToClass: 11

44th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11
    quota[4] = 8      BelongToClass: 11
    quota[5] = 8      BelongToClass: 11
    quota[6] = 8      BelongToClass: 11

45th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11
    quota[4] = 8      BelongToClass: 11
    quota[5] = 8      BelongToClass: 11
    quota[6] = 8      BelongToClass: 11
    quota[7] = 8      BelongToClass: 11

46th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11
    quota[4] = 8      BelongToClass: 11
    quota[5] = 8      BelongToClass: 11
    quota[6] = 8      BelongToClass: 11
    quota[7] = 8      BelongToClass: 11
    quota[8] = 8      BelongToClass: 11

47th BGF(N) = 8
    quota[1] = 8      BelongToClass: 11
    quota[2] = 8      BelongToClass: 11
    quota[3] = 8      BelongToClass: 11
    quota[4] = 8      BelongToClass: 11
    quota[5] = 8      BelongToClass: 11
    quota[6] = 8      BelongToClass: 11
    quota[7] = 8      BelongToClass: 11
    quota[8] = 8      BelongToClass: 11
    quota[9] = 8      BelongToClass: 11

48th BGF(N) = 6
    quota[1] = 6      BelongToClass: 12

49th BGF(N) = 6
    quota[1] = 6      BelongToClass: 12
    quota[2] = 6      BelongToClass: 12

50th BGF(N) = 6
    quota[1] = 6      BelongToClass: 12
    quota[2] = 6      BelongToClass: 12
    quota[3] = 6      BelongToClass: 12

51th BGF(N) = 6
    quota[1] = 6      BelongToClass: 12
    quota[2] = 6      BelongToClass: 12
    quota[3] = 6      BelongToClass: 12
    quota[4] = 6      BelongToClass: 12

52th BGF(N) = 6
    quota[1] = 6      BelongToClass: 12
    quota[2] = 6      BelongToClass: 12
    quota[3] = 6      BelongToClass: 12
    quota[4] = 6      BelongToClass: 12
    quota[5] = 6      BelongToClass: 12

53th BGF(N) = 6
    quota[1] = 6      BelongToClass: 12
    quota[2] = 6      BelongToClass: 12
    quota[3] = 6      BelongToClass: 12
    quota[4] = 6      BelongToClass: 12
    quota[5] = 6      BelongToClass: 12
    quota[6] = 6      BelongToClass: 12

54th BGF(N) = 5
    quota[1] = 5      BelongToClass: 12
    quota[2] = 5      BelongToClass: 12
    quota[3] = 5      BelongToClass: 12
    quota[4] = 5      BelongToClass: 12
    quota[5] = 5      BelongToClass: 12
    quota[6] = 5      BelongToClass: 12
    quota[7] = 0      BelongToClass: 12

55th BGF(N) = 7
    quota[1] = 5      BelongToClass: 12
    quota[2] = 5      BelongToClass: 12
    quota[3] = 5      BelongToClass: 12
    quota[4] = 5      BelongToClass: 12
    quota[5] = 5      BelongToClass: 12
    quota[6] = 5      BelongToClass: 12
    quota[7] = 1      BelongToClass: 12
    quota[8] = 7      BelongToClass: 12

```

Belong to 12th group, the quota cost is 0

h10_3.bmp belongs to

The match result of the lowest cost f(N), and quota

0th BG f(N) = 1	quota[1] = 0	BelongToClass: 1
1th BG f(N) = 3	quota[1] = 1 quota[2] = 1	BelongToClass: 1 BelongToClass: 1
2th BG f(N) = 1	quota[1] = 1 quota[2] = 3 quota[3] = 0	BelongToClass: 1 BelongToClass: 1 BelongToClass: 1
3th BG f(N) = 6	quota[1] = 1 quota[2] = 2 quota[3] = 1 quota[4] = 6	BelongToClass: 1 BelongToClass: 1 BelongToClass: 1 BelongToClass: 1
4th BG f(N) = 3	quota[1] = 1 quota[2] = 2 quota[3] = 1 quota[4] = 1 quota[5] = 3	BelongToClass: 1 BelongToClass: 1 BelongToClass: 1 BelongToClass: 1 BelongToClass: 1
5th BG f(N) = 5	quota[1] = 5	BelongToClass: 2
6th BG f(N) = 7	quota[1] = 4 quota[2] = 7	BelongToClass: 2 BelongToClass: 2
7th BG f(N) = 7	quota[1] = 4 quota[2] = 7 quota[3] = 7	BelongToClass: 2 BelongToClass: 2 BelongToClass: 2
8th BG f(N) = 1	quota[1] = 4 quota[2] = 6 quota[3] = 6 quota[4] = 1	BelongToClass: 2 BelongToClass: 2 BelongToClass: 2 BelongToClass: 2
9th BG f(N) = 8	quota[1] = 4 quota[2] = 6 quota[3] = 6 quota[4] = 0 quota[5] = 8	BelongToClass: 2 BelongToClass: 2 BelongToClass: 2 BelongToClass: 2 BelongToClass: 2
10th BGf(N) = 12	quota[1] = 12	BelongToClass: 3
11th BGf(N) = 4	quota[1] = 4 quota[2] = 4	BelongToClass: 3 BelongToClass: 3
12th BGf(N) = 15	quota[1] = 12 quota[2] = 4 quota[3] = 14	BelongToClass: 3 BelongToClass: 3 BelongToClass: 3
13th BGf(N) = 14	quota[1] = 11 quota[2] = 4 quota[3] = 13 quota[4] = 13	BelongToClass: 3 BelongToClass: 3 BelongToClass: 3 BelongToClass: 3
14th BGf(N) = 14	quota[1] = 12 quota[2] = 3 quota[3] = 13 quota[4] = 14 quota[5] = 14	BelongToClass: 3 BelongToClass: 3 BelongToClass: 3 BelongToClass: 3 BelongToClass: 3
15th BGf(N) = 15	quota[1] = 15	BelongToClass: 4
16th BGf(N) = 13	quota[1] = 11 quota[2] = 13	BelongToClass: 4 BelongToClass: 4
17th BGf(N) = 14	quota[1] = 11 quota[2] = 12 quota[3] = 14	BelongToClass: 4 BelongToClass: 4 BelongToClass: 4
18th BGf(N) = 11	quota[1] = 11 quota[2] = 12 quota[3] = 12 quota[4] = 13	BelongToClass: 4 BelongToClass: 4 BelongToClass: 4 BelongToClass: 4
19th BGf(N) = 12	quota[1] = 9 quota[2] = 11 quota[3] = 10 quota[4] = 11 quota[5] = 12	BelongToClass: 4 BelongToClass: 4 BelongToClass: 4 BelongToClass: 4 BelongToClass: 4
20th BGf(N) = 1	quota[1] = 0	BelongToClass: 5
21th BGf(N) = 1	quota[1] = 0 quota[2] = 0	BelongToClass: 5 BelongToClass: 5
22th BGf(N) = 1	quota[1] = 0 quota[2] = 0 quota[3] = 0	BelongToClass: 5 BelongToClass: 5 BelongToClass: 5
23th BGf(N) = 1	quota[1] = 0 quota[2] = 0 quota[3] = 0 quota[4] = 0	BelongToClass: 5 BelongToClass: 5 BelongToClass: 5 BelongToClass: 5
24th BGf(N) = 12	quota[1] = 12	BelongToClass: 7
25th BGf(N) = 1		

	quota[1] = 10	BelongToClass: 7
	quota[2] = 1	BelongToClass: 7
26th	BGF(N) = 13	
	quota[1] = 12	BelongToClass: 7
	quota[2] = 0	BelongToClass: 7
	quota[3] = 13	BelongToClass: 7
27th	BGF(N) = 1	
	quota[1] = 11	BelongToClass: 7
	quota[2] = 1	BelongToClass: 7
	quota[3] = 12	BelongToClass: 7
	quota[4] = 1	BelongToClass: 7
28th	BGF(N) = 10	
	quota[1] = 10	BelongToClass: 8
29th	BGF(N) = 1	
	quota[1] = 9	BelongToClass: 8
	quota[2] = 0	BelongToClass: 8
30th	BGF(N) = 3	
	quota[1] = 3	BelongToClass: 9
31th	BGF(N) = 8	
	quota[1] = 1	BelongToClass: 9
	quota[2] = 7	BelongToClass: 9
32th	BGF(N) = 11	
	quota[1] = 1	BelongToClass: 9
	quota[2] = 8	BelongToClass: 9
	quota[3] = 11	BelongToClass: 9
33th	BGF(N) = 10	
	quota[1] = 2	BelongToClass: 9
	quota[2] = 7	BelongToClass: 9
	quota[3] = 9	BelongToClass: 9
	quota[4] = 10	BelongToClass: 9
34th	BGF(N) = 0	
	quota[1] = 2	BelongToClass: 9
	quota[2] = 8	BelongToClass: 9
	quota[3] = 11	BelongToClass: 9
	quota[4] = 10	BelongToClass: 9
	quota[5] = 0	BelongToClass: 9
35th	BGF(N) = 12	
	quota[1] = 12	BelongToClass: 10
36th	BGF(N) = 13	
	quota[1] = 13	BelongToClass: 10
	quota[2] = 13	BelongToClass: 10
37th	BGF(N) = 8	
	quota[1] = 4	BelongToClass: 10
	quota[2] = 4	BelongToClass: 10
	quota[3] = 8	BelongToClass: 10
38th	BGF(N) = 5	
	quota[1] = 1	BelongToClass: 10
	quota[2] = 2	BelongToClass: 10
	quota[3] = 2	BelongToClass: 10
	quota[4] = 5	BelongToClass: 10
39th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
40th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
41th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
42th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
	quota[4] = 7	BelongToClass: 11
43th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
	quota[4] = 7	BelongToClass: 11
	quota[5] = 7	BelongToClass: 11
44th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
	quota[4] = 7	BelongToClass: 11
	quota[5] = 7	BelongToClass: 11
	quota[6] = 7	BelongToClass: 11
45th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
	quota[4] = 7	BelongToClass: 11
	quota[5] = 7	BelongToClass: 11
	quota[6] = 7	BelongToClass: 11
	quota[7] = 7	BelongToClass: 11
46th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
	quota[4] = 7	BelongToClass: 11
	quota[5] = 7	BelongToClass: 11
	quota[6] = 7	BelongToClass: 11
	quota[7] = 7	BelongToClass: 11
	quota[8] = 7	BelongToClass: 11
47th	BGF(N) = 7	
	quota[1] = 7	BelongToClass: 11
	quota[2] = 7	BelongToClass: 11
	quota[3] = 7	BelongToClass: 11
	quota[4] = 7	BelongToClass: 11
	quota[5] = 7	BelongToClass: 11
	quota[6] = 7	BelongToClass: 11

```

        quota[7] = 7      BelongToClass: 11
        quota[8] = 7      BelongToClass: 11
        quota[9] = 7      BelongToClass: 11

49th BGF(N) = 4
        quota[1] = 4      BelongToClass: 12

49th BGF(N) = 4
        quota[1] = 4      BelongToClass: 12
        quota[2] = 4      BelongToClass: 12

50th BGF(N) = 4
        quota[1] = 4      BelongToClass: 12
        quota[2] = 4      BelongToClass: 12
        quota[3] = 4      BelongToClass: 12

51th BGF(N) = 4
        quota[1] = 4      BelongToClass: 12
        quota[2] = 4      BelongToClass: 12
        quota[3] = 4      BelongToClass: 12
        quota[4] = 4      BelongToClass: 12

52th BGF(N) = 4
        quota[1] = 4      BelongToClass: 12
        quota[2] = 4      BelongToClass: 12
        quota[3] = 4      BelongToClass: 12
        quota[4] = 4      BelongToClass: 12
        quota[5] = 4      BelongToClass: 12

53th BGF(N) = 4
        quota[1] = 4      BelongToClass: 12
        quota[2] = 4      BelongToClass: 12
        quota[3] = 4      BelongToClass: 12
        quota[4] = 4      BelongToClass: 12
        quota[5] = 4      BelongToClass: 12
        quota[6] = 4      BelongToClass: 12

54th BGF(N) = 1
        quota[1] = 5      BelongToClass: 12
        quota[2] = 5      BelongToClass: 12
        quota[3] = 5      BelongToClass: 12
        quota[4] = 5      BelongToClass: 12
        quota[5] = 5      BelongToClass: 12
        quota[6] = 5      BelongToClass: 12
        quota[7] = 0      BelongToClass: 12

55th BGF(N) = 6
        quota[1] = 4      BelongToClass: 12
        quota[2] = 4      BelongToClass: 12
        quota[3] = 4      BelongToClass: 12
        quota[4] = 4      BelongToClass: 12
        quota[5] = 4      BelongToClass: 12
        quota[6] = 4      BelongToClass: 12
        quota[7] = 2      BelongToClass: 12
        quota[8] = 6      BelongToClass: 12

-----
Belong to 12th group, the quota cost is 0
-----
h11_3.bmp belongs to .....
The match result of the lowest cost f(N), and quota
3th BG f(N) = 8
        quota[1] = 7      BelongToClass: 1

1th BG f(N) = 4
        quota[1] = 0      BelongToClass: 1
        quota[2] = 4      BelongToClass: 1

2th BG f(N) = 8
        quota[1] = 8      BelongToClass: 1
        quota[2] = 4      BelongToClass: 1
        quota[3] = 7      BelongToClass: 1

3th BG f(N) = 1
        quota[1] = 6      BelongToClass: 1
        quota[2] = 3      BelongToClass: 1
        quota[3] = 6      BelongToClass: 1
        quota[4] = 1      BelongToClass: 1

4th BG f(N) = 5
        quota[1] = 1      BelongToClass: 1
        quota[2] = 4      BelongToClass: 1
        quota[3] = 1      BelongToClass: 1
        quota[4] = 1      BelongToClass: 1
        quota[5] = 5      BelongToClass: 1

5th BG f(N) = 2
        quota[1] = 2      BelongToClass: 2

6th BG f(N) = 0
        quota[1] = 3      BelongToClass: 2
        quota[2] = 0      BelongToClass: 2

7th BG f(N) = 0
        quota[1] = 3      BelongToClass: 2
        quota[2] = 0      BelongToClass: 2
        quota[3] = 0      BelongToClass: 2

8th BG f(N) = 6
        quota[1] = 1      BelongToClass: 2
        quota[2] = 1      BelongToClass: 2
        quota[3] = 1      BelongToClass: 2
        quota[4] = 6      BelongToClass: 2

9th BG f(N) = 3
        quota[1] = 3      BelongToClass: 2
        quota[2] = 1      BelongToClass: 2
        quota[3] = 1      BelongToClass: 2
        quota[4] = 7      BelongToClass: 2
        quota[5] = 1      BelongToClass: 2

10th BGF(N) = 5
        quota[1] = 5      BelongToClass: 3

11th BGF(N) = 3
        quota[1] = 5      BelongToClass: 3
        quota[2] = 3      BelongToClass: 3

12th BGF(N) = 8
        quota[1] = 5      BelongToClass: 3
        quota[2] = 1      BelongToClass: 3
        quota[3] = 7      BelongToClass: 3

13th BGF(N) = 7

```

	quota[1] = 4	BelongToClass: 3
	quota[2] = 3	BelongToClass: 3
	quota[3] = 6	BelongToClass: 3
	quota[4] = 6	BelongToClass: 3
14th BGF(N) = 8		
	quota[1] = 6	BelongToClass: 3
	quota[2] = 3	BelongToClass: 3
	quota[3] = 7	BelongToClass: 3
	quota[4] = 8	BelongToClass: 3
	quota[5] = 8	BelongToClass: 3
15th BGF(N) = 8		
	quota[1] = 8	BelongToClass: 4
16th BGF(N) = 6		
	quota[1] = 4	BelongToClass: 4
	quota[2] = 6	BelongToClass: 4
17th BGF(N) = 8		
	quota[1] = 7	BelongToClass: 4
	quota[2] = 6	BelongToClass: 4
	quota[3] = 8	BelongToClass: 4
18th BGF(N) = 7		
	quota[1] = 5	BelongToClass: 4
	quota[2] = 6	BelongToClass: 4
	quota[3] = 6	BelongToClass: 4
	quota[4] = 7	BelongToClass: 4
19th BGF(N) = 6		
	quota[1] = 3	BelongToClass: 4
	quota[2] = 5	BelongToClass: 4
	quota[3] = 4	BelongToClass: 4
	quota[4] = 5	BelongToClass: 4
	quota[5] = 6	BelongToClass: 4
20th BGF(N) = 8		
	quota[1] = 7	BelongToClass: 5
21th BGF(N) = 8		
	quota[1] = 7	BelongToClass: 5
	quota[2] = 7	BelongToClass: 5
22th BGF(N) = 8		
	quota[1] = 7	BelongToClass: 5
	quota[2] = 7	BelongToClass: 5
	quota[3] = 7	BelongToClass: 5
23th BGF(N) = 8		
	quota[1] = 7	BelongToClass: 5
	quota[2] = 7	BelongToClass: 5
	quota[3] = 7	BelongToClass: 5
	quota[4] = 7	BelongToClass: 5
24th BGF(N) = 5		
	quota[1] = 5	BelongToClass: 7
25th BGF(N) = 6		
	quota[1] = 5	BelongToClass: 7
	quota[2] = 6	BelongToClass: 7
26th BGF(N) = 6		
	quota[1] = 5	BelongToClass: 7
	quota[2] = 7	BelongToClass: 7
	quota[3] = 6	BelongToClass: 7
27th BGF(N) = 7		
	quota[1] = 5	BelongToClass: 7
	quota[2] = 5	BelongToClass: 7
	quota[3] = 6	BelongToClass: 7
	quota[4] = 7	BelongToClass: 7
28th BGF(N) = 2		
	quota[1] = 2	BelongToClass: 8
29th BGF(N) = 8		
	quota[1] = 2	BelongToClass: 8
	quota[2] = 7	BelongToClass: 8
30th BGF(N) = 6		
	quota[1] = 6	BelongToClass: 9
31th BGF(N) = 1		
	quota[1] = 6	BelongToClass: 9
	quota[2] = 0	BelongToClass: 9
32th BGF(N) = 5		
	quota[1] = 5	BelongToClass: 9
	quota[2] = 2	BelongToClass: 9
	quota[3] = 5	BelongToClass: 9
33th BGF(N) = 3		
	quota[1] = 5	BelongToClass: 9
	quota[2] = 0	BelongToClass: 9
	quota[3] = 2	BelongToClass: 9
	quota[4] = 3	BelongToClass: 9
34th BGF(N) = 7		
	quota[1] = 5	BelongToClass: 9
	quota[2] = 1	BelongToClass: 9
	quota[3] = 4	BelongToClass: 9
	quota[4] = 3	BelongToClass: 9
	quota[5] = 7	BelongToClass: 9
35th BGF(N) = 5		
	quota[1] = 5	BelongToClass: 10
36th BGF(N) = 5		
	quota[1] = 5	BelongToClass: 10
	quota[2] = 5	BelongToClass: 10
37th BGF(N) = 1		
	quota[1] = 3	BelongToClass: 10
	quota[2] = 3	BelongToClass: 10
	quota[3] = 1	BelongToClass: 10
38th BGF(N) = 2		
	quota[1] = 6	BelongToClass: 10
	quota[2] = 5	BelongToClass: 10
	quota[3] = 1	BelongToClass: 10
	quota[4] = 2	BelongToClass: 10

```

39th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11

40th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11

41th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11

42th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11

43th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11

44th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11
    quota[6] = 0      BelongToClass: 11

45th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11
    quota[6] = 0      BelongToClass: 11
    quota[7] = 0      BelongToClass: 11

46th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11
    quota[6] = 0      BelongToClass: 11
    quota[7] = 0      BelongToClass: 11
    quota[8] = 0      BelongToClass: 11

47th BGF(N) = 0
    quota[1] = 0      BelongToClass: 11
    quota[2] = 0      BelongToClass: 11
    quota[3] = 0      BelongToClass: 11
    quota[4] = 0      BelongToClass: 11
    quota[5] = 0      BelongToClass: 11
    quota[6] = 0      BelongToClass: 11
    quota[7] = 0      BelongToClass: 11
    quota[8] = 0      BelongToClass: 11
    quota[9] = 0      BelongToClass: 11

48th BGF(N) = 3
    quota[1] = 3      BelongToClass: 12

49th BGF(N) = 3
    quota[1] = 3      BelongToClass: 12
    quota[2] = 3      BelongToClass: 12

50th BGF(N) = 3
    quota[1] = 3      BelongToClass: 12
    quota[2] = 3      BelongToClass: 12
    quota[3] = 3      BelongToClass: 12

51th BGF(N) = 3
    quota[1] = 3      BelongToClass: 12
    quota[2] = 3      BelongToClass: 12
    quota[3] = 3      BelongToClass: 12
    quota[4] = 3      BelongToClass: 12

52th BGF(N) = 3
    quota[1] = 3      BelongToClass: 12
    quota[2] = 3      BelongToClass: 12
    quota[3] = 3      BelongToClass: 12
    quota[4] = 3      BelongToClass: 12
    quota[5] = 3      BelongToClass: 12

53th BGF(N) = 3
    quota[1] = 3      BelongToClass: 12
    quota[2] = 3      BelongToClass: 12
    quota[3] = 3      BelongToClass: 12
    quota[4] = 3      BelongToClass: 12
    quota[5] = 3      BelongToClass: 12
    quota[6] = 3      BelongToClass: 12

54th BGF(N) = 8
    quota[1] = 2      BelongToClass: 12
    quota[2] = 2      BelongToClass: 12
    quota[3] = 2      BelongToClass: 12
    quota[4] = 2      BelongToClass: 12
    quota[5] = 2      BelongToClass: 12
    quota[6] = 2      BelongToClass: 12
    quota[7] = 7      BelongToClass: 12

55th BGF(N) = 1
    quota[1] = 1      BelongToClass: 12
    quota[2] = 1      BelongToClass: 12
    quota[3] = 1      BelongToClass: 12
    quota[4] = 1      BelongToClass: 12
    quota[5] = 1      BelongToClass: 12
    quota[6] = 1      BelongToClass: 12
    quota[7] = 7      BelongToClass: 12
    quota[8] = 1      BelongToClass: 12

-----
Belong to 11th group, the quota cost is 0
-----
h12_1.bmp belongs to .....
The match result of the lowest cost f(N), and quota
0th BG f(N) = 5
    quota[1] = 4      BelongToClass: 1

1th BG f(N) = 1

```

	quota[1] = 3	BelongToClass: 1
	quota[2] = 1	BelongToClass: 1
2th BG f(N) = 5		
	quota[1] = 5	BelongToClass: 1
	quota[2] = 1	BelongToClass: 1
	quota[3] = 4	BelongToClass: 1
3th BG f(N) = 7		
	quota[1] = 4	BelongToClass: 1
	quota[2] = 1	BelongToClass: 1
	quota[3] = 4	BelongToClass: 1
	quota[4] = 3	BelongToClass: 1
4th BG f(N) = 2		
	quota[1] = 2	BelongToClass: 1
	quota[2] = 1	BelongToClass: 1
	quota[3] = 2	BelongToClass: 1
	quota[4] = 2	BelongToClass: 1
	quota[5] = 2	BelongToClass: 1
5th BG f(N) = 2		
	quota[1] = 2	BelongToClass: 2
6th BG f(N) = 4		
	quota[1] = 1	BelongToClass: 2
	quota[2] = 4	BelongToClass: 2
7th BG f(N) = 4		
	quota[1] = 1	BelongToClass: 2
	quota[2] = 4	BelongToClass: 2
	quota[3] = 4	BelongToClass: 2
8th BG f(N) = 3		
	quota[1] = 2	BelongToClass: 2
	quota[2] = 4	BelongToClass: 2
	quota[3] = 4	BelongToClass: 2
	quota[4] = 3	BelongToClass: 2
9th BG f(N) = 5		
	quota[1] = 1	BelongToClass: 2
	quota[2] = 3	BelongToClass: 2
	quota[3] = 3	BelongToClass: 2
	quota[4] = 3	BelongToClass: 2
	quota[5] = 5	BelongToClass: 2
10th BGf(N) = 8		
	quota[1] = 8	BelongToClass: 3
11th BGf(N) = 0		
	quota[1] = 8	BelongToClass: 3
	quota[2] = 0	BelongToClass: 3
12th BGf(N) = 11		
	quota[1] = 8	BelongToClass: 3
	quota[2] = 0	BelongToClass: 3
	quota[3] = 10	BelongToClass: 3
13th BGf(N) = 11		
	quota[1] = 8	BelongToClass: 3
	quota[2] = 1	BelongToClass: 3
	quota[3] = 10	BelongToClass: 3
	quota[4] = 10	BelongToClass: 3
14th BGf(N) = 10		
	quota[1] = 8	BelongToClass: 3
	quota[2] = 1	BelongToClass: 3
	quota[3] = 9	BelongToClass: 3
	quota[4] = 10	BelongToClass: 3
	quota[5] = 10	BelongToClass: 3
15th BGf(N) = 11		
	quota[1] = 11	BelongToClass: 4
16th BGf(N) = 9		
	quota[1] = 7	BelongToClass: 4
	quota[2] = 9	BelongToClass: 4
17th BGf(N) = 10		
	quota[1] = 9	BelongToClass: 4
	quota[2] = 8	BelongToClass: 4
	quota[3] = 10	BelongToClass: 4
18th BGf(N) = 9		
	quota[1] = 7	BelongToClass: 4
	quota[2] = 8	BelongToClass: 4
	quota[3] = 8	BelongToClass: 4
	quota[4] = 9	BelongToClass: 4
19th BGf(N) = 8		
	quota[1] = 5	BelongToClass: 4
	quota[2] = 7	BelongToClass: 4
	quota[3] = 6	BelongToClass: 4
	quota[4] = 7	BelongToClass: 4
	quota[5] = 8	BelongToClass: 4
20th BGf(N) = 5		
	quota[1] = 4	BelongToClass: 5
21th BGf(N) = 5		
	quota[1] = 4	BelongToClass: 5
	quota[2] = 4	BelongToClass: 5
22th BGf(N) = 5		
	quota[1] = 4	BelongToClass: 5
	quota[2] = 4	BelongToClass: 5
	quota[3] = 4	BelongToClass: 5
23th BGf(N) = 5		
	quota[1] = 4	BelongToClass: 5
	quota[2] = 4	BelongToClass: 5
	quota[3] = 4	BelongToClass: 5
	quota[4] = 4	BelongToClass: 5
24th BGf(N) = 8		
	quota[1] = 8	BelongToClass: 7
25th BGf(N) = 3		
	quota[1] = 8	BelongToClass: 7
	quota[2] = 3	BelongToClass: 7
26th BGf(N) = 10		
	quota[1] = 9	BelongToClass: 7

quota[2] = 3	BelongToClass: 7
quota[3] = 10	BelongToClass: 7
27th BGF(N) = 5	
quota[1] = 7	BelongToClass: 7
quota[2] = 3	BelongToClass: 7
quota[3] = 8	BelongToClass: 7
quota[4] = 5	BelongToClass: 7
28th BGF(N) = 6	
quota[1] = 6	BelongToClass: 8
29th BGF(N) = 6	
quota[1] = 4	BelongToClass: 8
quota[2] = 5	BelongToClass: 8
30th BGF(N) = 2	
quota[1] = 2	BelongToClass: 9
31th BGF(N) = 4	
quota[1] = 3	BelongToClass: 9
quota[2] = 1	BelongToClass: 9
32th BGF(N) = 7	
quota[1] = 1	BelongToClass: 9
quota[2] = 4	BelongToClass: 9
quota[3] = 7	BelongToClass: 9
33th BGF(N) = 7	
quota[1] = 1	BelongToClass: 9
quota[2] = 4	BelongToClass: 9
quota[3] = 6	BelongToClass: 9
quota[4] = 7	BelongToClass: 9
34th BGF(N) = 4	
quota[1] = 2	BelongToClass: 9
quota[2] = 4	BelongToClass: 9
quota[3] = 7	BelongToClass: 9
quota[4] = 6	BelongToClass: 9
quota[5] = 4	BelongToClass: 9
35th BGF(N) = 9	
quota[1] = 9	BelongToClass: 10
36th BGF(N) = 8	
quota[1] = 8	BelongToClass: 10
quota[2] = 8	BelongToClass: 10
37th BGF(N) = 4	
quota[1] = 0	BelongToClass: 10
quota[2] = 0	BelongToClass: 10
quota[3] = 4	BelongToClass: 10
38th BGF(N) = 1	
quota[1] = 7	BelongToClass: 10
quota[2] = 6	BelongToClass: 10
quota[3] = 2	BelongToClass: 10
quota[4] = 1	BelongToClass: 10
39th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
40th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
41th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
42th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 1	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
quota[4] = 3	BelongToClass: 11
43th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
quota[4] = 3	BelongToClass: 11
quota[5] = 3	BelongToClass: 11
44th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
quota[4] = 3	BelongToClass: 11
quota[5] = 3	BelongToClass: 11
quota[6] = 3	BelongToClass: 11
45th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
quota[4] = 3	BelongToClass: 11
quota[5] = 3	BelongToClass: 11
quota[6] = 3	BelongToClass: 11
quota[7] = 3	BelongToClass: 11
46th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
quota[4] = 3	BelongToClass: 11
quota[5] = 3	BelongToClass: 11
quota[6] = 3	BelongToClass: 11
quota[7] = 3	BelongToClass: 11
quota[8] = 3	BelongToClass: 11
47th BGF(N) = 3	
quota[1] = 3	BelongToClass: 11
quota[2] = 3	BelongToClass: 11
quota[3] = 3	BelongToClass: 11
quota[4] = 3	BelongToClass: 11
quota[5] = 3	BelongToClass: 11
quota[6] = 3	BelongToClass: 11
quota[7] = 3	BelongToClass: 11
quota[8] = 3	BelongToClass: 11
quota[9] = 3	BelongToClass: 11
48th BGF(N) = 0	

```

    quota[1] = 0      BelongToClass: 12
49th BGF(N) = 0
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
50th BGF(N) = 0
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
    quota[3] = 0      BelongToClass: 12
51th BGF(N) = 0
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
    quota[3] = 0      BelongToClass: 12
    quota[4] = 0      BelongToClass: 12
52th BGF(N) = 0
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
    quota[3] = 0      BelongToClass: 12
    quota[4] = 0      BelongToClass: 12
    quota[5] = 0      BelongToClass: 12
53th BGF(N) = 0
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
    quota[3] = 0      BelongToClass: 12
    quota[4] = 0      BelongToClass: 12
    quota[5] = 0      BelongToClass: 12
    quota[6] = 0      BelongToClass: 12
54th BGF(N) = 6
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
    quota[3] = 0      BelongToClass: 12
    quota[4] = 0      BelongToClass: 12
    quota[5] = 0      BelongToClass: 12
    quota[6] = 0      BelongToClass: 12
    quota[7] = 5      BelongToClass: 12
55th BGF(N) = 2
    quota[1] = 0      BelongToClass: 12
    quota[2] = 0      BelongToClass: 12
    quota[3] = 0      BelongToClass: 12
    quota[4] = 0      BelongToClass: 12
    quota[5] = 0      BelongToClass: 12
    quota[6] = 0      BelongToClass: 12
    quota[7] = 6      BelongToClass: 12
    quota[8] = 2      BelongToClass: 12

```

Belong to 12th group, the quota cost is 0