International Journal of Soft Computing 1 (1) : 36-40, 2006


# Multiple Blocks Query for a Simple Context-Based 2-DGE Retrieval System

[1]Tzu-Shin T., [2]C. Jeanne, [3]C. Tung-Shou, [3]W. Tian-Shing, [3]C. Shu-Bin,
[1]C. Ming-Chih and [1]L. Shuan-Yow
[1]Institute of Medicine, Chung Shan Medical University, Taichung, 402 Taiwan
[2]Department of Computer Science Infomation Management, Hungkuang University,
Taichung, 43302 Taiwan
[3]Graduate School of Computer Science Information Technology,
Nattional Taichung Institute of Technology,
Taichung, 404 Taiwan

**Abstract:** In the proposed context-based retrieval technique, a 2-DGE image is filtered out for noises and unobvious protein spots using a threshold. The significant protein spots on the filtered 2-DGE images will then be indexed and confined within their block properties which will all be stored in the 2DGE database for query purposes. Queries could be made by retrieving a 2-DGE image and marking one or multiple protein blocks where specific searches could be made to retrieve the required information. The process to create the 2-DGE database is simple and the search is quick. The proposed technique will be helpful to biologists with needs for unrestricted access to a 2-DGE image database.

**Key words:** Protein spots, 2-D Gel Electrophoresis (2-DGE), context-based retrieval, multiple blocks query

## INTRODUCTION

In proteomics informatics, proteins in cells are isolated using the 2-D Gel Electrophoresis (2-DGE) technique[1,2]. These gels can easily become moldy at room temperature (including air conditioning) and must be kept refrigerated. For a biologist, this is cumbersome since any laboratory test would require the gel samples to be out of refrigeration. Digitizing the 2-D gel and its protein contents would allow a biologist to have free unrestricted access and for longer periods of time[3,4]. However, to accurately identify and track the protein information can be a very difficult but not impossible task.

Analysis tools such as Melanie [5], Z3[4] and Image Master 2D[5] had been used to analyze the 2-DGE images. However, keeping tracking of these 2-DGE images can be overwhelming and difficult. Chang *et al.*,[6] proposed a 2D string technique to track protein spots based on each spot's spatial location relative to the spatial locations of its neighboring protein spots. Explicit information must be stored for all the coordinate locations and their relationships. Other spatial techniques include the enhanced versions for the 2D string such as the 2D C-string[7,8], 2D G-string[9] and 2D B-string[10]. Another technique, the 2D R-tree[11] was based on the size and locations of the 2-DGE image. Similar to the 2D string, all these techniques are burdened with storage requirements to upkeep the information on the relative spatial locations of the individual protein spots and their neighbors.

For this study, a simple and novel context-based retrieval technique is proposed to create a 2-DGE search database. Filtered versions of the 2-DGE images and the defined block properties of the protein spots will be stored in the database for queries. Searches could be made by marking one or multiple protein blocks on a 2-DGE image and using the block properties as the search criteria.

## THE PROPOSED CONTEXT-BASED RETRIEVAL TECHNIQUE

The proposed context retrieval technique can be divided into two parts. The first part is to record related information of each protein spot such as the coordinate

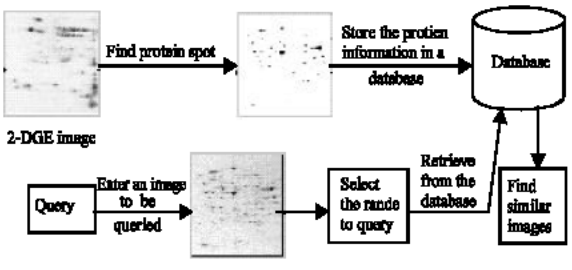
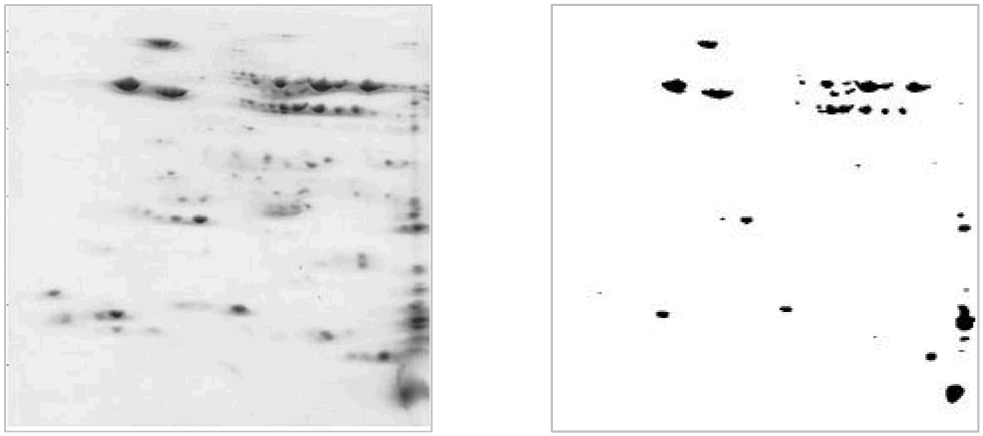



Fig. 1: Flow for the proposed technique Creating the 2-DGE database

**Corresponding Author:** Tian-Shing, W., Graduate School of Computer Science Information Technology, Nattional Taichung Insttitute of Technology, Taichung, 404 Taiwan

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 145 | 142 | 137 | 132 | 129 | 130 | 132 | 132 |
| 112 | 108 | 101 | 123 | 110 | 121 | 100 | 107 |
| 100 | 103 | 67 | 58 | 54 | 55 | 58 | 63 |
| 111 | 110 | 67 | 56 | 51 | 52 | 55 | 60 |
| 114 | 122 | 67 | 60 | 55 | 56 | 58 | 69 |
| 118 | 111 | 105 | 110 | 124 | 120 | 114 | 147 |
| 162 | 157 | 100 | 143 | 130 | 55 | 62 | 59 |
| 195 | 193 | 149 | 150 | 132 | 45 | 56 | 60 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 67 | 58 | 54 | 55 | 58 | 63 |
| 225 | 225 | 67 | 56 | 51 | 52 | 55 | 60 |
| 225 | 225 | 67 | 60 | 55 | 56 | 58 | 69 |
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 225 | 225 | 225 | 55 | 62 | 59 |
| 225 | 225 | 225 | 225 | 225 | 45 | 56 | 60 |



(a) Before filtering (b) After filtering

Fig. 2: The pixel values before and after filtering

location of the protein spot and its size based on the two extreme coordinate locations defining a diagonal to the block kernel for the protein spot and the center point.

The second part is to create a query interface where a 2-DGE image could be loaded and one or multiple protein blocks, selectively marked for query purposes. The flow for the proposed technique is as shown in Fig. 1.

**Filtering out unobvious protein spots and noises:** Information on the 2-DGE image like the color depth, size and position of a protein spot are considered as the basis for identification. First, unobvious (or undesired) protein spots and noises will be filtered off the image. The filtered 2-DGE image with the obvious protein spots will then be stored in the database. Filtering is done by setting those regions to be filtered out to 255. The regions to filter out depends on a criteria set for a threshold value. Figure 2 illustrates an example to filter out the unobvious protein spots and the noises for a gray-level 2-DGE with pixel values between 0-255. In the example, the threshold value is set at 100 so that any pixel value greater or equal than 100 will be reset to 255 and displayed as white – this inferred the pixels as being filtered-out. On the other hand, pixel values smaller than 100 will remain unmodified and will be stored together with the newly modified pixel values as shown in Fig. 2(b) – the respective images are as illustrated in Fig. 2(c) and (d).

**Collecting information on the protein spots:** After filtering out the unobvious protein information and noises, the next step is to index each protein spot. Each protein spot makes a block. Starting from the top-left pixel value on the image, pixel values smaller than the threshold value are stacked in the First-In-Last-Out (FILO) order.

The last pixel value is popped off the stack to check the adjacent eight pixels. Pixel values less the threshold are then stored. Pixels being stored will not be stacked back. This procedure is repeated until there are no more pixels. An example of this simple process is illustrated in Fig. 3. In Fig. 3(a), scanning starts from the top-left pixel value, left to right and top to bottom to located pixel values that are not 255. As seen, the first pixel value found is 67. This pixel value is stored. Then pixels from the eight pixels which have values not equal to 255 will be stacked. The pixel values on the stack will be popped off by the FILO order and compared to the adjacent eight pixels until there are no more pixels (see Fig. 3 (b)). The blocks will be indexed and marked 1, 2, 3, … for the protein spots, respectively.

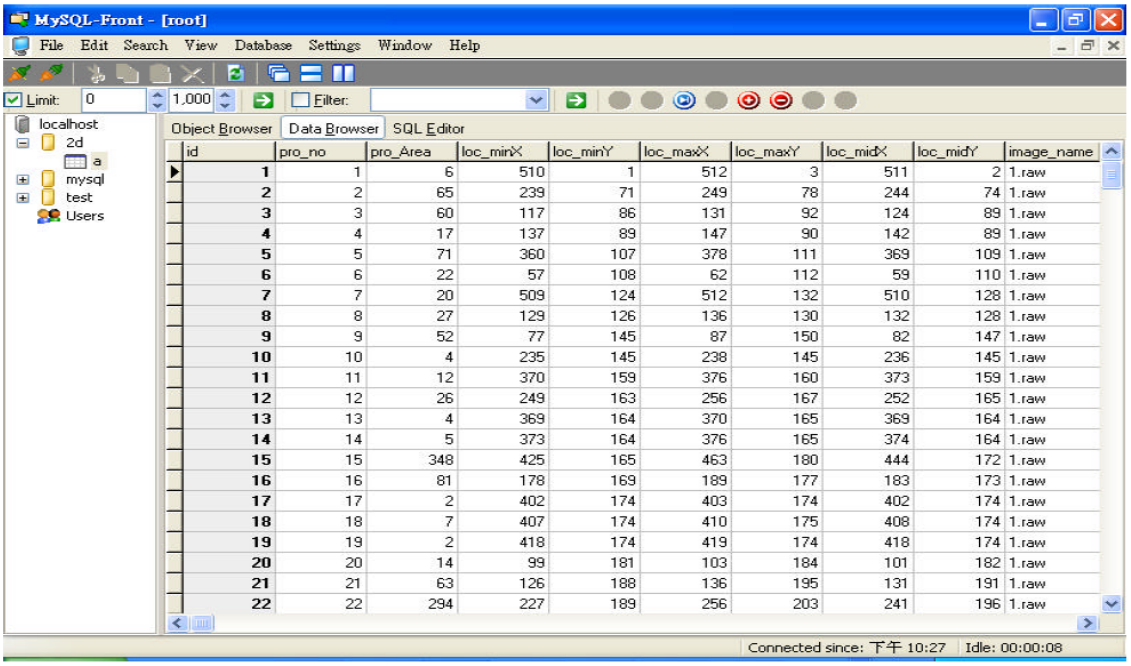
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 67 | 58 | 54 | 55 | 58 | 63 |
| 225 | 225 | 67 | 56 | 51 | 52 | 55 | 60 |
| 225 | 225 | 67 | 60 | 55 | 56 | 58 | 69 |
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 225 | 225 | 225 | 55 | 62 | 59 |
| 225 | 225 | 225 | 225 | 225 | 45 | 56 | 60 |

(a) Before an index value is set

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 1 | 1 | 1 | 1 | 1 | 1 |
| 225 | 225 | 1 | 1 | 1 | 1 | 1 | 1 |
| 225 | 225 | 1 | 1 | 1 | 1 | 1 | 1 |
| 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 |
| 225 | 225 | 225 | 225 | 225 | 2 | 2 | 2 |
| 225 | 225 | 225 | 225 | 225 | 2 | 2 | 2 |

(b) After an index alue is set

Fig. 3: Indexing the protein blocks




| id | pro_no | pro_Area | loc_minX | loc_minY | loc_maxX | loc_maxY | loc_midX | loc_midY | image_name |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 6 | 510 | 1 | 512 | 3 | 511 | 2 | 1.raw |
| 2 | 2 | 65 | 239 | 71 | 249 | 78 | 244 | 74 | 1.raw |
| 3 | 3 | 60 | 117 | 86 | 131 | 92 | 124 | 89 | 1.raw |
| 4 | 4 | 17 | 137 | 89 | 147 | 90 | 142 | 89 | 1.raw |
| 5 | 5 | 71 | 360 | 107 | 378 | 111 | 369 | 109 | 1.raw |
| 6 | 6 | 22 | 57 | 108 | 62 | 112 | 59 | 110 | 1.raw |
| 7 | 7 | 20 | 509 | 124 | 512 | 132 | 510 | 128 | 1.raw |
| 8 | 8 | 27 | 129 | 126 | 136 | 130 | 132 | 128 | 1.raw |
| 9 | 9 | 52 | 77 | 145 | 87 | 150 | 82 | 147 | 1.raw |
| 10 | 10 | 4 | 235 | 145 | 238 | 145 | 236 | 145 | 1.raw |
| 11 | 11 | 12 | 370 | 159 | 376 | 160 | 373 | 159 | 1.raw |
| 12 | 12 | 26 | 249 | 163 | 256 | 167 | 252 | 165 | 1.raw |
| 13 | 13 | 4 | 369 | 164 | 370 | 165 | 369 | 164 | 1.raw |
| 14 | 14 | 5 | 373 | 164 | 376 | 165 | 374 | 164 | 1.raw |
| 15 | 15 | 348 | 425 | 165 | 463 | 180 | 444 | 172 | 1.raw |
| 16 | 16 | 81 | 178 | 169 | 189 | 177 | 183 | 173 | 1.raw |
| 17 | 17 | 2 | 402 | 174 | 403 | 174 | 402 | 174 | 1.raw |
| 18 | 18 | 7 | 407 | 174 | 410 | 175 | 408 | 174 | 1.raw |
| 19 | 19 | 2 | 418 | 174 | 419 | 174 | 418 | 174 | 1.raw |
| 20 | 20 | 14 | 99 | 181 | 103 | 184 | 101 | 182 | 1.raw |
| 21 | 21 | 63 | 126 | 188 | 136 | 195 | 131 | 191 | 1.raw |
| 22 | 22 | 294 | 227 | 189 | 256 | 203 | 241 | 196 | 1.raw |

Fig. 4: Individual records from a 2-DGE image

After the protein blocks are indexed, the coordinates of each protein spot is calculated by finding the center between two points diagonally from the bottom-left corner and the other at the top-right corner of the smallest rectangle area that makes the protein block. The information including the coordinates of bottom-left, top-right and center points and the size are stored in a database as criteria for search.

**2-DGE image query interface:** The first part is completed after all the filtered 2-DGE images and their information are stored in the database. The final task is to create an image query interface. Coding for the interface is written with the JAVA programming language. To query, a 2-DGE image will be loaded where one or multiple protein blocks will be

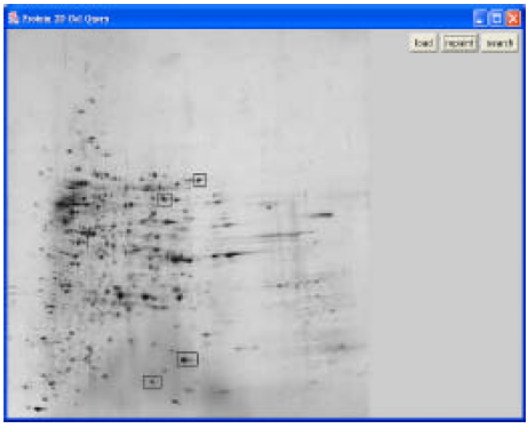


Fig. 5: Marking blocks on a 2-DGE image for query

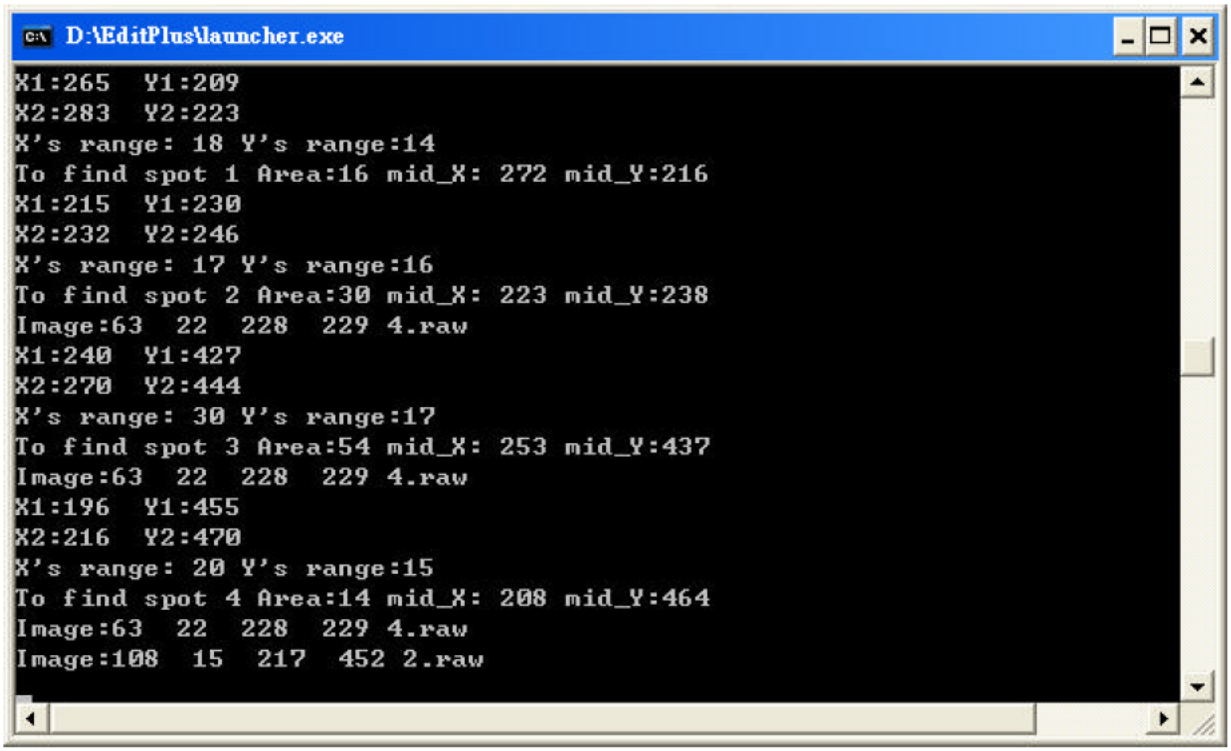



Fig. 6: Results from query on the marked protein spots

marked and submit as criteria for the search to retrieve information on the proteins including information on other 2-DGE images that matches the same query criteria.

## EXPERIMENTAL RESULTS AND ANALYSIS

The 2-DGE database is built on MySQL with Java programming. The process for storing the 2-DGE images and related information include using a threshold value of 100 to filter out unobvious protein spots and noises. Figure 4 shows sample data records for collecting information on a 2-DGE image, "1.raw".

The 2-DGE image query interface is as shown in Fig. 5. A 2-DGE image is first loaded and selective protein spots are marked as blocks for query.

Once the protein spots to be queried are marked, the sizes, the center coordinates and coordinates for the blocks will be used to search the 2-DGE image database. The search criteria are set for the size of protein spots to be around±10 and coordinate of the center to be around±20. Figure 6 shows results from the selected protein spots on the desired image such as the coordinates of the center point, size and filename of image. The coordinates of the protein spots and related information are used on the examples without going through a complicated spatial relationship operator as those in[6]. The search is simple and requires less time.

## CONCLUSION

The 2-DGE image displays information about the variability and distribution of proteins. Analysis can be carried out on the extent of variability and distribution of these proteins which might help identify possible sicknesses or to develop a cure for. The analysis can aid a biologist in designing a routine for a patient to control his/her protein distribution until it gets down to those of a normal person's.

In the proposed context-based retrieval technique, multiple blocks can be marked on a 2-DGE image as criteria to retrieve all 2-DGE images with similar information. The search is simple and takes less time to retrieve the required information. Also, the database requires very little storage space. The context-based retrieval system can be useful to the biologist who requires unrestricted access to the 2-DGE image and the need to find out the protein information in other 2-DGE images.

## REFERENCES

1. Gibas, C. and P. Jambeck, 2001. Developing Bioinformatics Computer Skills, O'Reilly.

2. Liebler, D.C., 2001. Introduction to Proteomics-Tools for the New Biology, Humana Press Inc., pp: 31-56.
3. O'Farrell, P.H., 1975. High Resolution Two-dimensional electrophoresis of proteins. J. Biol. Chem., pp: 4007-4021.
4. Schneider, T.D., 2000. Evolution of Biolo. Information. Nucleic Acids Research, 28:14, pp: 2794-2799.
5. ImageMaster 2D Platinum ver. 5, http://www1 .amershambiosciences.com.
6. Chang, S.K. Q.Y. Shi and C.W. Yan, 1987. Iconic indexing by 2-D Strings, IEEE Transactions on Pattern Analysis and Machine Intelligence, 3: 413-428.
7. Huang, P.W. and Y.R. Jean, 1994. Using 2D C+-string as spatial knowledge representation for image database systems. Pattern Recognition, pp: 1249-1257.
8. Lee, S.Y. and F.J. Hsu, 1990. 2D C-string: A New Spatial Knowledge Representation for Image Database Systems. Pattern Recognition, pp: 1077-1087.
9. Chang, S.K. E. Jungert and Y. Li, 1998. Representation and Retrieval of Symbolic Pictures Using Generalized 2D String, Technical Report, University of Pittsburgh.
10. Lee, S.Y., M.C. Yang and J.W. Chen, 1992. 2D B-string: A Spatial Knowledge Representation for Image Database Systems, Proc. ICSC92 Second Int. Computer Sci. Conf., pp: 609-615.
11. Guttman, A., 1984. R-Trees: A dynamic index structure for spatial searching, in Proc. of the ACM SIGMOD Conference on Management of Data, pp: 47-57.