MACE: Lossless Compression and Analysis of Microarray Images

Robert Bierman[1], Nidhi Maniyar[1], Charles Parsons[2], Rahul Singh[1]
[1] Dept. of Computer Science, San Francisco State University, San Francisco, CA
[2] College of Medicine, University of Vermont, Burlington, VT
bierman@sfsu.edu, nidhi@sfsu.edu, charles.parsons@uvm.edu, rsingh@cs.sfsu.edu

ABSTRACT

The ubiquity of microarray expression data in state-of-the-art biology has been well established. The widespread adoption of this technology coupled with the significant volume of image-based experimental data generated per experiment (averaging 40 MB), have led to significant challenges in storage and query-retrieval of primary data from microarray experiments. Research in the yet nascent area of microarray data-compression seeks to address this problem. In this paper, we propose a conceptually novel approach that achieves significantly better lossless compression. Unlike lossy compression, our algorithm is guaranteed against loss of information that may have potential biological relevance. The proposed method supports key operations such as automated grid and spot finding, histogram-based automatic thresholding for spot segmentation, and subsequent foreground and background separation. Based on the proposed approach, we have developed a standardized format for storing microarray data that encapsulates all the relevant information, including both the Cy3 and Cy5 expression images significantly compressed. We have also developed a software application called MACE (Microarray Compression and Extraction application) to compress-decompress microarray data and generate the aforementioned format. Compression-decompression results on a wide class of microarray experiments involving different spot layouts validate the effectiveness of our approach and its potential to significantly address the aforementioned challenges in storage and management of microarray data.

Categories and Subject Descriptors

E.4 [**Coding and Information Theory**]: *Data compaction and compression.* J.3 [**Life and Medical Sciences**]: *Biology and genetics.* H.3.3 [**Information Systems**] *Information Storage & Retrieval*.

Keywords

Microarray, microarray data compression and storage, microarray data analysis.

1. INTRODUCTION

Microarrays have become an important tool in the study of gene function, regulation, and interaction across large numbers of genes, and even entire genomes. Allowing the analysis of a broad spectrum of gene expressions in a single experiment [6, 13, 17], microarrays are essential in investigations related to Systems Biology as well as research in specific areas such as cancer [1] and metabolic pathway discovery. Paralleling this popularity, microarrays have also been the focus of significant research in Computer Science, specifically related to the analysis and mining of data emanating from such experiments. The focus of this paper is on addressing the issue of *effective compression-decompression of primary microarray data*. While this problem is critical to the success of microarray-based research (both biological and computational), it has received relatively less attention in the computer science and engineering communities.

To fully understand the technical challenge, we present a brief overview of the process through which microarray data is generally made available in repositories: The raw image-based data of a microarray experiment, henceforth referred to as the primary data, consists of a pair of 16 bits per pixels (bpp) grayscale images. A single microarray image is generally tens of megabytes in size, and a single hybridization experiment often generates greater than 40 MB of data. Once obtained, the image is analyzed using a variety of software tools to extract the relevant intensity of each spot in the microarray, and this information is used to evaluate the expression level of individual genes. Due to the large size of the (image-based) primary data, a predominant majority of online databases and repositories distribute *only the extracted expression levels and not the original microarray images*. In the following, we call the alphanumeric expression-level information obtained after processing the primary image-based output from the microarray, as secondary data. In the Stanford Microarray Database for instance, only the extracted expression data is stored in spreadsheet form. These files are approximately 12MB each or roughly ¼ the size of the actual experimental images. An analysis of this process raises a series of important issues that impact both processing as well as ease of storage and query-retrieval of microarray information. These include:

- *Dependence of downstream researchers on techniques used for generating the secondary (numeric) expression data from the primary (image-based) expression data*: It is important to note that a number of assumptions are used in the derivation of secondary data from a microarray image, and the methodology used to extract such information is the subject of considerable debate [5, 11, 12, 16]. Differences in the method of grid determination, spot allocation, and background intensity correction can significantly impact the validity of data collected from a single experiment. The process of normalizing data across microarray experiments in also non-trivial. Due to the absence of primary image-based microarray data, biological as well as algorithmic inferences have to be drawn by treating the primary-to-secondary microarray data conversion process as a black-box.

- *Inability to regenerate numeric-expression values from primary image-based microarray data using alternate algorithmic or processing techniques:* Image analysis and interpretation is a complex challenge and there is no *a priori* optimal technique. The typical lack of access to primary microarray data forces researchers to depend, without recourse, on the secondary data and eschew any opportunities for selective re-analysis of the original image or optimization of parameters employed in deriving the secondary expression values. A related problem is the inability to apply newly developed image-analysis algorithms to legacy data due to the primary data being unavailable.
- *Difficulty in comparing results across platforms and laboratories:* As alluded to earlier, normalization of microarray-data requires knowledge about specificities of imaging conditions and noise models. Such information typically gets lost once the secondary expression values are generated. Lack of access to the original image-based microarray output, therefore makes rigorous comparisons across different experimental conditions difficult.
- *Lack of an easy way to share primary microarray data:* The proliferation of microarray experiments and their utilization, have led to an increasing need to share microarray data electronically or through publicly available repositories [3, 4, 7, 8]. However, due to size constraints, primary-microarray data is typically unsuitable for transmission using commonly used information-exchange technologies. The massive size of each individual data entity also significantly complicates its storage in databases.
- *Increased experimental costs*: Inability to efficiently retrieve microarray-information, due to lack of easy access to primary-data necessitates replication of expensive experiments, especially across laboratories. This could be avoided with easy access to the primary image-based data.

An analysis of the above issues underscores the importance of maintaining and making available the original raw image output from microarray experiments. Towards this goal, we propose a lossless compression scheme that offers significantly greater compression than previously published lossless compression methods. Furthermore, we propose a new file format (extension .mai) and software application (MACE: ***M**icro**a**rray **C**ompression and **E**xtraction software) for the storage and analysis of microarray images. A primary advantage of the proposed format is the inclusion of all relevant experimental data in a single file containing both the Cy3 channel image and the Cy5 channel image, thus reducing the risk of data loss that accompanies handling multiple files. Our approach also allows on-the-fly regeneration of expression data for a varying set of assumptions, eliminating dependencies on statically generated data that may not be compatible across experiments.

The current approach towards compression of microarray images [10, 15] focuses on image-analysis for spot finding to separate a microarray image into separate channels based on pixel similarities. Once separated, the channels are compressed individually. In a critical departure from this line of reasoning, our method does not rely on spot finding to separate the channels. Instead it separates channels based solely on pixel intensities. This allows the compression scheme to function independent of any image-analysis conducted to determine biologically-relevant features. This is a major departure from other methods such as [14] where spot finding is done first, followed by lossless compression of the foreground and lossy compression of the background. The primary complication of following such a strategy lies in that spots are often noisy and difficult to clearly define algorithmically at the boundaries (Figure 1). Therefore, spot-detection based compression can inherently loose important data around spot edges or be conservative and have poor compression performance.

In contrast to the above approach, our solution methodology is completely lossless for both channels, which we refer to as the "dark" and "light" planes. Further, it does not depend on the grid determination or spot finding to separate the planes. This allows for true reconstruction of the image and for uniform comparison across different expression analysis experiments. Finally the absence of any dependence on feature (spot) extraction, leads to its robustness, even in the case of noisy spots. Another recent technique, SLOCO [10] proposes the use of lossy-compression schemes under the condition that the resulting data loss is less than the array-to-array variability in replicated experiments. While this approach results in better compression than our method, it is inherently lossy and employs a statistical criterion which can reliably work only under conditions of having multiple replicates. This may not always be possible, especially given the cost of each microarray experiment. Additionally, the bounds in [15] typically hold true at the entire-image level, and therefore allow significant local information loss, at points that contain biologically critical information.

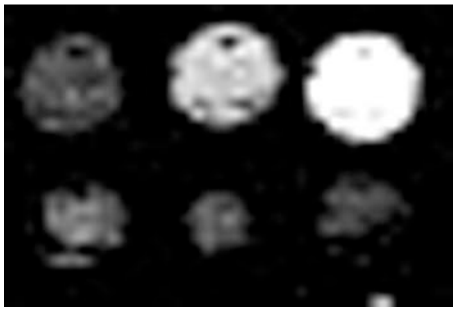

Figure 1. Actual spots showing variability in spot quality and the difficulty in demarcating the actual spot

2. COMPRESSION-DECOMPRESSION

MACE employs a unique and specific series of steps to compress the microarray images. The core idea behind the compression scheme lies in utilizing a systemic characteristic of microarray imaging that leads to highly skewed intensity distributions of microarray images as illustrated in Figure 2. Basing itself on the nature of this distribution, the compression step decomposes the image into two planes (the aforementioned dark and light planes). The characteristic nature of microarray images causes a high degree of uniformity in the dark plane, which yields highly efficient lossless compression.

The light plane support comparatively less efficient lossless compression as compared to the dark plane, however the overall compression significantly improves on other lossless compression schemes. The decompression step consists of a lossless extraction of the two planes followed by a simple overlay operation to generate the original microarray data exactly. The following subsections describe each of these steps in detail.

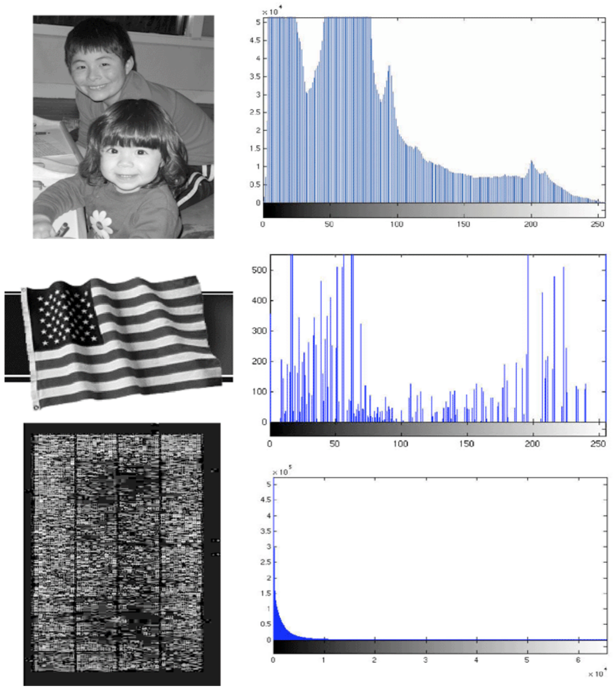

Figure 2. Histogram comparison between three common image types: natural (top), graphic (middle), and cDNA microarray (bottom). Note the prevalence of low intensity pixels in the microarray image

2.1 Compression

A common 40,000 spot microarray chip when scanned generates two TIFF files that are roughly 5500 x 1950 pixels at 16bpp (bits per pixel) which results in 21MB files, 42MB for one experiment. Microarray images are not “images” in the normal sense but are relatively fixed position data files consisting of a non-random distribution of pixel intensities. Therefore, classical image compression techniques, which focus on maintaining visual acuity, produce suboptimal compression for microarray images.

The Histograms of images (Figure 2) show an overwhelming majority of the microarray image exists with pixel values less then 1024 (for a 16 bpp image), and in fact a majority are under 256. The large number of low intensity pixels (avg. 77.02% < 256, 96.71% < 1024; sample set) is an artifact of the process used to fluoresce microarray images. In order not to lose data when fluorescing images due to pixel saturation, it is important that make sure the maximum intensity value present in the image does not exceed 65535. Thus, a scarcity of informative high intensity pixels causes the majority of remaining pixels to be captured at lower intensities. The majority of tools currently used to capture microarray image data currently account for this [2]. As a systemic artifact of the process used to develop microarrays, this feature can be expected to be maintained across microarray platforms.

An overview of the proposed approach is illustrated in Figure 3. The basic idea behind our scheme is to separate the pixels of the image into two classes based on similarities among each class of pixels, and then compress the resulting classes separately. Utilizing a predetermined pixel intensity threshold of 2^8, 2^9, 2^{10}, or 2^{11} based solely on the distribution of the intensity value of the pixels in the image, we split the image into a

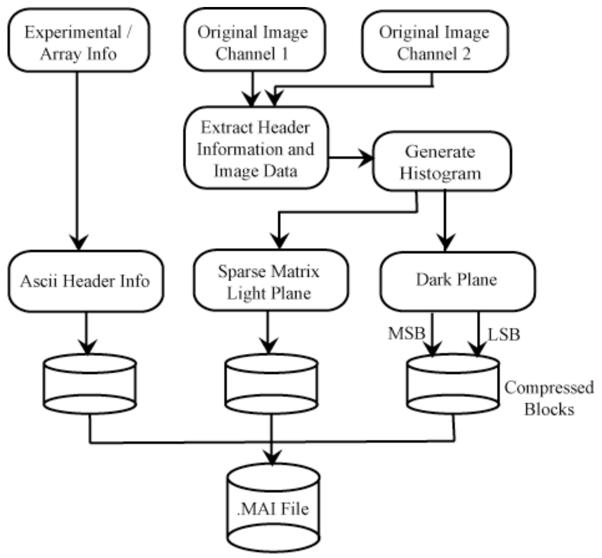

Figure 3. Flow chart for the creation of a .MAI file

"dark" plane and a "light" plane (Figure 4). Within the dark plane, all pixels with a pixel value greater then the threshold are set to a value of zero (Figure 4) (or the greatest pixel value within the plane). This is done so that the dark plane remains one continuous block. No data is lost since the pixels that are overwritten in the dark plane exist in the light plane and are replaced during reconstruction of the image. The repetitive nature of the pixel values allow for significant compression.

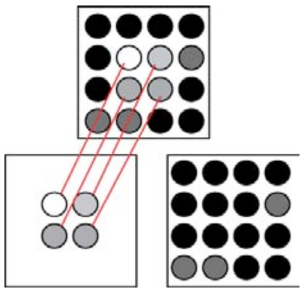

Figure 4. Division of the image into the light plane and the dark plane. The light plane becomes sparse and the dark plane replaces the light plane pixels with “dark” pixels.

The dark plane is then separated into two blocks (Figure 5), the MSB (most significant byte) in one block and the LSB (least significant byte) in a second block. Note that the MSB can only have the values of 0 - 7 (i.e. 0, 1, 2, or 3 bits) because of the maximum threshold of 2048. If 256 is used as the threshold then all of the bytes in the MSB block are zero and therefore that block is discarded resulting in an immediate 50% size reduction.

The threshold is determined by the percentage of pixels that fall within one of the four thresholds. We look for approximately 90% of the pixels to fall within the threshold. A simple scan of the data is then done to determine if there is a greater uniformity by compressing the byte string horizontally or vertically. Vertical transposition generally provides better uniformity. This is because the height of the microarray images tends to be around three times greater than their width and after vertical transposition the bands between spots tend to be more uniform.

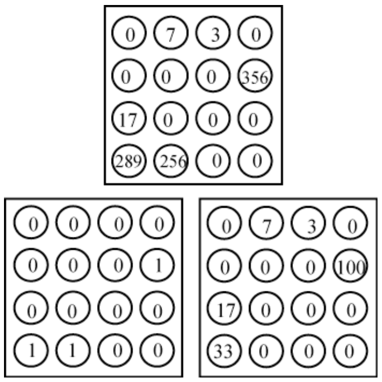

Figure 5 Division of the dark plane into the MSB (Most Significant Byte) and LSB (Least Significant Byte). Numeric Representation: One plane of 16 bit values is divided into two planes of 8 bit values.

The dark plane blocks are then compressed using the best result from GZip, Lempel-Ziv-Welch, Huffman, or variable-length encoding compression routines [19]. Because of the uniformity of the dark plane, compression is extensive, from 60% to 80%, with an average of 67% reduction for the reference set of images. The light plane consists of individual blocks of pixels all having intensity values equal to or greater than the original threshold, which roughly correspond to the spot areas on the microarray. These pixel blocks are placed together using a simple sparse matrix algorithm, followed by compression. The compression of the light plane is not a extensive as the dark plane but is still in the 45% range, with an average of 40% for the reference set of images. This yields overall compression in the 50% to 66% range. See Table 1 for various comparisons of compression performance.

In addition to the image data there is also the experimental information which describes the information about each spot such as the clone ID, gene name, etc. and the tag information maintained in the TIFF files. As shown in Figure 3 this data is then combined with the two images and written to the .MAI file.

One potential area for optimization of our compression approach is a dynamic determination of the threshold for separating the light and dark planes of the microarray image. To test this hypothesis, we implemented a thresholding algorithm based on the Otsu method of image thresholding [15].

$$\sigma_B^2(k) = \frac{[\mu_T\omega(k) - \mu(k)]^2}{\omega(k)[1-\omega(k)]}$$

Where: $\omega(k)$ is the zeroth cumulative moment, $\mu(k)$ is the first cumulative moment, μ_T is the total mean level, and $\sigma_B^2(k)$ is the optimal threshold. However, our results did not show any significant improvement utilizing this technique over using the simple thresholds described above, though the resulting threshold usually was close to optimal. One possible reason for this discrepancy is the fact that the distribution of pixels is generally not bi-model and therefore not suited to automated thresholding using [15].

2.2 Decompression

Regeneration of the compressed image is straightforward, with the results of the extraction of each of the two planes, light and dark, overlaid to recreate the original image. This process is a lossless method where the dark plane is first reconstructed from the MSB and LSB blocks remerged (Figure 5). This step gives us a representation of every pixel in the original image, where certain pixels that are represented in the light plane have been zeroed. The light plane is then reconstructed with only those pixels that were part of the light plane overwriting the reconstructed dark plane. The result is a perfect pixel to pixel match with the original image.

3. AUTOMATED GRID PLACEMENT AND SPOT DETECTION

The MACE software application brings together methods in grid and spot detection into a single application and is designed to automatically determine the dimensions and locations of the subgrid and the spots of the microarray image. Towards this, we extend the techniques proposed in [10] so that gridding and spotting can be done both on spot arrangements in the matrix (simple row, column) format as well as the latest high-density "orange pack" format (see Figure 6) . For the "orange pack" format the rotation of the X axis allows the alignment along a "row" of spots and provides the intersection points with the columns to mark each of the spots.

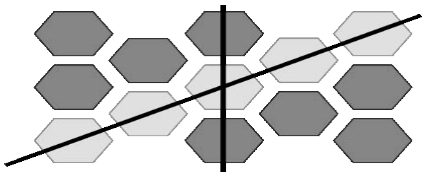

Figure 6. Diagram of a hexagonal packed array, "orange pack" also showing offset grid lines for spot detection

3.1. Grid placement analysis

The generalized technique used to find the subgrid and spot location is to compute the average pixel intensities of the microarray image in both the vertical axis (row by row) and horizontal axis (column by column). A plot of the pixel intensities in each axis yields a curve containing local maxima that roughly correspond to hybridization points (spots), and local minima that correspond to the background regions between spots (grids). This plot is smoothed and refined by using a 4 pixel window initially and then by using half of the average distance between minima. The subgrid placing can be done by referring to maxima points in the plot that differ by more than 25 pixels (in both the row and column plots). These points correspond to the subgrid coordinates (Figure 7).

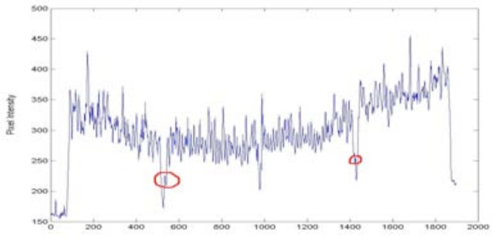

Figure 7. Plot of average column intensity in the microarray image

However, the detection of such spots gets fairly challenging owing to noise which results in peaks of very low intensity values between two maxima points. Therefore the plot for subgrid spacing is smoothed out again to exclude low intensity maxima points by using a filter of 0.95*average minimum intensity. Finally, the subgrid placing is done over this smoothed plot by using a filter of 25 pixels.

3.2. Spot location analysis

The technique of grid placement is applied recursively to determine the spacing of spots. In the refined plot, which is smoothed over 4 pixels, the maxima correspond to the spot's highest intensity pixel and minima correspond to the background between the spots. The spacing between the minima points yields average spacing between the spots. Thus the center of the spot is (approximately) midway between the minima points. Further refinement of center results in efficient spot location analysis.

4. SPOT SEGMENTATION

Numerous methods have been suggested to accurately extract the correct expression data from a spot on a microarray. Historically, the majority of the algorithms in use for spot segmentation assume a circular spot shape and implement techniques like fixed circle or adaptive circle segmentation methods.

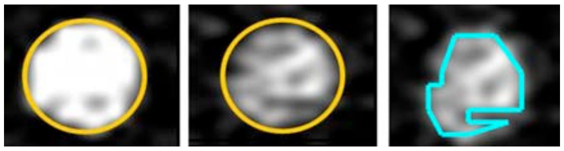

Figure 8. Circular vs. histogram threshold spot segmentation. A comparison of spot circularity segmentation (left, middle) to histogram threshold spot segmentation (right). Note the misidentified pixels (in black) captured in the middle image when assuming spot circularity

However, the main disadvantage of assuming spot circularity is that the features may include areas of low intensity which are not actually hybridization signals (Figure 8). An alternative method implemented in MACE, is the histogram threshold spot segmentation algorithm proposed in [9].

The segmentation focuses on differentiating the foreground from the background in order to store the spot. The histogram based spot segmentation technique uses a target mask that is chosen as a square area centered at the spot with side length equal to inter spot spacing. Pixels are classified as foreground or background using thresholds from the histogram of pixel values within the masked area. The foreground pixels are identified to be at the upper 35% end of the histogram and background pixels are identified to be at the lower 10% end.

Alternative methods of spot segmentation like adaptive shape segmentation using seeded region growing technique are a logical extension to be supported in future versions of the MACE application.

5. MACE APPLICATION DESIGN

The MACE application is actually broken into two parts. The first part of the application is concerned with the creation or the .mai file (and also the reconstruction of the image), the MACE application creates the .mai files without any reference to biologic distinction in the original images. The second part of the application handles automated extraction of the expression data. It is important that the extraction of the expression data be automated since you may be regenerating the expression data for a large number of experiments.

Due to the rapid nature of technological advance and the relative infancy of microarray technology, the MACE application is built around an extensible architecture to allow for the rapid assimilation of new techniques as they become available. This is accomplished through a pluggable interface in which features for spot detection, intensity quantification, background correction, and normalization can be added to the software at runtime.

6. EXPERIMENTS

To test the effectiveness of our techniques, a sample set of microarray images was obtained from a published gene expression experiment [18]. The image set consisted of 5 cDNA microarray experiments, with each experiment consisting of two TIFF images, one image each for sample and reference hybridizations.

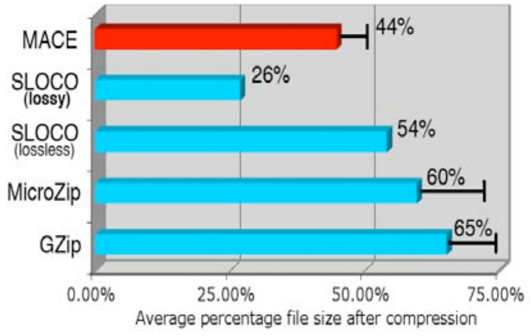

Figure 9. Average compression performance of various lossless and a lossy algorithms. Std. Dev. is shown for techniques where data was available

As shown in Figure 9 and Table 1, the average compression achieved using MACE was 44% for our sample set of images verses 64% for GZip. Also included in the figure are averages from MicroZip [14] a lossless compression for microarray images and SLOCO [10] which support both lossless and lossy compression. While our results are not a good as the SLOCO lossy compression, there is clear benefit to MACE against all of the lossless compression techniques. This results in savings of 4MB to 8MB per single 40MB experiment (a 10% – 20% additional savings).

Table 1. Comparative compression across multiple experiments and grid layouts

Experiment	Channel	Pixel Ct.	Uncompressed File Size	GZip using Stuff-it	% GZip	bpp Gzip	MACE File Size	% MACE	bpp MACE
13729	Ch 1	10,374,912	20,751,810	14,555,007	70.14%	11.22	10,672,026	51.43%	8.23
	Ch2		20,751,810	14,436,843	69.57%	11.13	10,249,428	49.39%	7.90
14250	Ch 1	10,448,208	20,898,410	14,438,079	69.09%	11.05	9,290,322	44.45%	7.11
	Ch2		20,898,410	14,528,323	69.52%	11.12	10,580,495	50.63%	8.10
36429	Ch 1	10,664,640	21,331,322	13,696,590	64.21%	10.27	7,276,684	34.11%	5.46
	Ch2		21,331,322	13,782,781	64.61%	10.34	9,592,236	44.97%	7.20
36430	Ch 1	10,664,640	21,331,322	13,520,549	63.38%	10.14	9,572,663	44.88%	7.18
	Ch2		21,331,323	13,699,537	64.22%	10.28	9,574,970	44.89%	7.18
36431	Ch 1	10,664,640	21,331,322	11,524,421	54.03%	8.64	8,167,073	38.29%	6.13
	Ch2		21,331,323	13,430,575	62.96%	10.07	9,332,889	43.75%	7.00
MeeboP1099	Orange Pack	10,374,912	21,218,373	11,105,102	52.34%	8.56	8,068,208	38.02%	6.22
Averages		**10,531,992**	**21,136,977**	**13,519,801**	**63.96%**	**10.27**	**9,306,999**	**44.03%**	**7.07**

An issue that arose with the compression is that "dictionary" based compression algorithms such as LZW degrade as the file size increases because the space for the dictionary is quickly filled leaving patterns found later in the file uncompressed. In addition very long runs of the same value are not captured well by the dictionary. This is why we first apply a simple run-length-encoding before applying the dictionary based compression. We also evaluate the compression result from the image in its native form as well as in its transposed form. Table 2 lists the longest uniform runs in the LSB block of the dark plane. We see orders of magnitude difference in the transposed block.

To test the thresolding, a subset of images was chosen at random and each image was compressed for each of the thresholds including the Otsu threshold. The Otsu threshold was not the best but usually close to the best. We believe that further research along this direction may eliminate the need for running multiple compressions to determine the optimal threshold. Table 2 shows the data from these experiments.

Table 2. Threshold and RLE information

Filename	Otsu Threshold	Best Threshold	Longest RLE	Longest RLE on Transposed
13729_ch1.tif	18908	2048	8846	263460
13729_ch2.tif	20168	2048	4635	166762
14250_ch1.tif	10735	8192	5390	175463
14250_ch2.tif	21077	2048	4871	167831
36429_ch1.tif	6999	1024	3682	162205
36429_ch2.tif	9322	512	2673	181515
36430_ch1.tif	2017	256	3341	167997
36430_ch2.tif	1387	256	2586	173792
36431_ch1.tif	1218	512	2193	166070
36431_ch2.tif	8011	1024	1858	195034

7. CONCLUSION AND FUTURE WORK

This work presents a conceptually novel approach to the problem of Microarray image compression. Using this compression-decompression idea as the core, we have developed a complete software application that supports the tasks of microarray compression, gridding, and spotting in a comprehensive end-to-end fashion. The research presented in this paper distinguishes itself from prior state-of-the-art on three significant fronts. First, it provides lossless compression that does not depend on significantly complex and potentially error-prone image analysis for spot finding. Second, it can be applied to arbitrarily array layouts, including regular and orange-pack configurations. Finally, the compression performance significantly improves upon those obtained with other lossless algorithms. Experimental results indicate that the MACE application offers a compelling alternative to currently available techniques in microarray image storage and analysis and can significantly ameliorate the critical challenges associated with storage, analysis, and exchange of primary (image-based) microarray data.

ACKNOWLEDGEMENTS

We would like to thank Dr. Jonathan Pollack, of the Pollack Lab and Janos Demeter, the Microarray Curator, both of Stanford University for their valuable assistance in providing sample microarray images.

REFERENCES

[1] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack and A. J. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Proc Natl Acad Sci U S A, 96 (1999), pp. 6745-50.

[2] I. Axon Instruments, GenePix Pro 6.0: Microarray Aquisition and Analysis Software for GenePix Microarray Scanners: User's Guide & Tutorial, (2004), pp. 24.

[3] C. A. Ball, I. A. Awad, J. Demeter, J. Gollub, J. M. Hebert, T. Hernandez-Boussard, H. Jin, J. C. Matese, M. Nitzberg, F. Wymore, Z. K. Zachariah, P. O. Brown and G. Sherlock, The Stanford Microarray Database accommodates additional microarray platforms and data formats, Nucleic Acids Res, 33 (2005), pp. D580-2.

[4] T. Barrett, T. O. Suzek, D. B. Troup, S. E. Wilhite, W.-C. Ngau, P. Ledoux, D. Rudnev, A. E. Lash, W. Fujibuchi and R. Edgar, NCBI GEO: mining millions of expression profiles--database and tools, Nucl. Acids Res., 33 (2005), pp. D562-566.

[5] D. E. Bassett, Jr., M. B. Eisen and M. S. Boguski, Gene expression informatics--it's all in your mine, Nat Genet, 21 (1999), pp. 51-5.

[6] J. DeRisi, L. Penland, P. O. Brown, M. L. Bittner, P. S. Meltzer, M. Ray, Y. Chen, Y. A. Su and J. M. Trent, Use of a cDNA microarray to analyse gene expression patterns in human cancer, Nat Genet, 14 (1996), pp. 457-60.

[7] R. Edgar, M. Domrachev and A. E. Lash, Gene Expression Omnibus: NCBI gene expression and hybridization array data repository, Nucl. Acids Res., 30 (2002), pp. 207-210.

[8] J. Gollub, C. A. Ball, G. Binkley, J. Demeter, D. B. Finkelstein, J. M. Hebert, T. Hernandez-Boussard, H. Jin, M. Kaloper, J. C. Matese, M. Schroeder, P. O. Brown, D. Botstein and G. Sherlock, The Stanford Microarray Database: data access and quality assessment tools, Nucleic Acids Res, 31 (2003), pp. 94-6.

[9] A. N. Jain, T. A. Tokuyasu, A. M. Snijders, R. Segraves, D. G. Albertson and D. Pinkel, Fully automatic quantification of microarray image data, Genome Res, 12 (2002), pp. 325-32.

[10] R. Jornsten, W. Wang, B. Yu and K. Ramchandran, Microarray image compression: SLOCO and the effect of information loss, Signal Processing, 83 (2003), pp. 859.

[11] R. Kothapalli, S. Yoder, S. Mane and T. Loughran, Microarray results: how accurate are they? BMC Bioinformatics, 3 (2002), pp. 22.

[12] Y. F. Leung and D. Cavalieri, Fundamentals of cDNA micro-array data analysis, Trends Genet, 19 (2003), pp. 649-59.

[13] D. J. Lockhart, H. Dong, M. C. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Horton and E. L. Brown, Expression monitoring by hybridization to high-density oligonucleotide arrays, Nat Biotechnol, 14 (1996), pp. 1675-80.

[14] S. Lonardi and Y. Luo, Gridding and Compression of Microarray Images, Proc. of the 2004 IEEE Comp. Sys. Bioinformatics Conference (2004).

[15] N. Otsu, A threshold selection method from gray-level histogram, IEEE Transactions on System Man Cybernetics, SMC-9 (1979), pp. 62-66.

[16] R. Sasik, C. H. Woelk and J. Corbeil, Microarray truths and consequences, J Mol Endocrinol, 33 (2004), pp. 1-9.

[17] M. Schena, D. Shalon, R. W. Davis and P. O. Brown, Quantitative monitoring of gene expression patterns with a complementary DNA microarray, Science, 270 (1995), pp. 467-70.

[18] R. Shyamsundar, Y. Kim, J. Higgins, K. Montgomery, M. Jorden, A. Sethuraman, M. van de Rijn, D. Botstein, P. Brown and J. Pollack, A DNA microarray survey of gene expression in normal human tissues, Genome Biology, 6 (2005), pp. R22.

[19] J. Ziv and A. Lempel, Compression of individual sequences via variable-rate coding, IEEE Trans Inf Theory, IT-24 (1978), pp. 530.