Journal of Molecular Imaging & Dynamics
ISSN: 2155-9937

Wang et al., J Mol Imag Dynamic 2014, 4:1
DOI: 10.4172/2155-9937.1000115

Research Article Open Access

Speeding up the Analysis of Neuron Morphology using Parallel Processing

Wang DD*, Bourke D, Domanski L and Vallotton P

Quantitative Imaging, CSIRO Mathematics, Informatics and Statistics, Locked Bag 17, North Ryde, NSW 1670, Australia

Abstract

Researchers using High Content Screening systems generate thousands of fluorescently labeled cell images from which they measure subtle but important phenotypic changes summarized by dozens of parameters. Large image datasets and fast turnaround requirement have made the efficient High Content Analysis a challenging task. This paper studies multi-core based high performance image analysis and its application to data and compute-intensive High Content Analyses. A vertical parallelization strategy is employed and an automated parallelization framework is implemented to automatically dispatch image processing tasks. The strategy is based on allocation of different images to separate processors so that each image is analyzed sequentially on a single processor and multiple images are processed by separate processors in parallel. Experiments demonstrate that this approach, of a generic character, considerably increases throughput.

Keywords: Cell image analysis; Batch processing; High content screening; High content analysis; High performance computing

Introduction

Automated image analysis is used by pharmaceutical companies to measure changes in cell morphology, both rapidly and accurately. The field itself, dubbed High Content Analysis (HCA) is emerging as one of the fastest growing sectors in drug discovery and development. It represents the convergence between cell-based assays, high-resolution imaging, and advanced quantitative image analysis [1]. There is no hard dividing line between High Content Screening (HCS) and HCA though the former is generally higher-throughput while the latter has an emphasis on gaining the maximum information from an assay, typically based on images. HCS systems achieve high throughput by rapidly capturing and processing data from entire micro-well plates. Each well in these plates contains cells or biochemical samples, which have been labeled to detect the changes induced by perturbations, such as addition of a candidate drug compound or a gene knockout. The images can be very dense, with hundreds of cells and complex cell morphology. It may take several hours or even days to process the images generated from a single experiment, which may be unacceptable in practice. The ideal image processing time lies within the same time frame as the image capturing. However, the reality is that the image processing time still tends to be much longer than industry's expectations, even on a high-end computer. Therefore, strategies bases on High Performance Computing (HPC) are necessary.

HPC comes from parallelism, fast-dense circuitry, and packaging technology [2]. Over the last decade, several studies have been conducted in the application of HPC to image processing [3-6]. Most of these studies focus on the use of HPC infrastructure or distributed processing in an application driven research environment. The solutions presented in these studies are based on supercomputers and computer clusters. With the enormous progress in computing power, computers with multi-core CPUs are now standard. The multi-core shift presents unprecedented opportunities for researchers to deal with large datasets efficiently. It has triggered some efforts towards developing parallel image analysis algorithms that take advantage of these powerful processors. Trease et al. introduced a high-performance hybrid multi-core processing framework for processing videos and images [7]. Hartley et al. illustrated a cooperative parallelization approach where multiple CPU sockets, multiple Graphical Processing Units (GPUs) and multiple cluster nodes coexist. Literature reviews have revealed few studies that explore multi-core based HCA systems. In general, high performance image analysis algorithms have not been sufficiently developed and investigated so far in the drug screening context. As most of HCA systems are not based on supercomputing facilities, studies on high performance HCA solutions based on multi-core computers appear more practical, and have potential to make a difference in terms of cost and quality in drug discovery.

In this paper, we present an enabling solution that makes the most of one's multi-core computer to achieve high throughput image processing.

Methods

Researchers attempt to make sense of massive amount of image data through HCA. For example, to identify compounds that affect neurite outgrowths, several micro-well-plate experiments need to be carried out. Eventually, hundreds of thousands of microscope images may be generated. Well-based and cell-based features are extracted from the images using HCA software packages. As some dense images may have over 1000 cells featuring complicated structures, processing large amounts of such images in a timely manner is difficult. This section outlines the case of neurite outgrowth analysis to provide some background information on the challenge.

Large image datasets

When screening for neurite outgrowth, multiple plates of images are typically produced. Each micro-well plate may have 384 wells, and 12 images may be sampled from each well. This will produce 4608 images from each plate. If 5 plates of images are generated, there will be 23040 images to be processed. If each image has three channels (RGB),

***Corresponding author:** Wang DD, Quantitative Imaging, CSIRO Mathematics, Informatics and Statistics, Locked Bag 17, North Ryde, NSW 1670, Australia, Tel: +61395452176, E-mail: dadong.wang@csiro.au

Received June 26, 2014; **Accepted** August 21, 2014; **Published** October 10, 2014

Citation: Wang DD, Bourke D, Domanski L, Vallotton P (2014) Speeding up the Analysis of Neuron Morphology using Parallel Processing. J Mol Imag Dynamic 4: 115. doi:10.4172/2155-9937.1000115

Copyright: © 2014 Wang DD, et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

with one channel showing the labeled nuclei, one showing the labeled neurons and neurites, and one showing additional information such as labeled proteins in some sub-cellular compartments, the number of images to be processed will be 69120. A typical image has a dimension of 1280×1280 pixels. It turns out that the HCA for neurite outgrowth analysis is a data intensive computation process.

Compute-intensive analysis

In neurite analysis, some images may be very "dense", containing over 1000 neurons. As shown in Figure 1, many neurons may be clumped together, and some neurons may have complicated neurite structures. This increases the complexity of analysis. From Figure 1, one can see that many neurite branches overlap. De-clumping neurons and generating neurite branch-level features and assigning the correct neurites to the corresponding neurons become a challenging task (Figure 1).

Neurite analysis deals with quantitative measures and statistics of neuron and neurite structures on both a cell-by-cell and image-by-image basis. Some definitions of neurite segments are illustrated in Figure 2. They are listed below:

Neurite segments: S1 to S8; Roots: R1 to R2; Extremities: E1 to E5; Branching points: B1 to B3; Longest neurite segment: S6 + S7; Neurite field area: area of convex hull, i.e. the area enclosed by dotted lines; Primary neurite: neurite shown in red; Secondary neurite: neurite shown in cyan; Tertiary neurite: neurite shown in blue; Branch layers: layer 1 is for primary; 2 for secondary, and 3 for tertiary.

To analyze neurite outgrowth, the following measurements are required

Cell based measurements: The cell-based measurement includes 34 parameters such as cell area and perimeter, maximum and mean intensity, total neurite length, max neurite length, max and mean branch layer, number of branch points, number of roots, number of segments, number of extremities, neurite field area, neurite area, max and mean intensity of neurite etc.

Image-wide summary statistics: This includes the number of cells in the image, total and average neurite length, total and average number of segments, average longest neurite from a cell, total and average number of roots, total and average number of extreme neurites, total and average number of branch points, average branching layers etc.

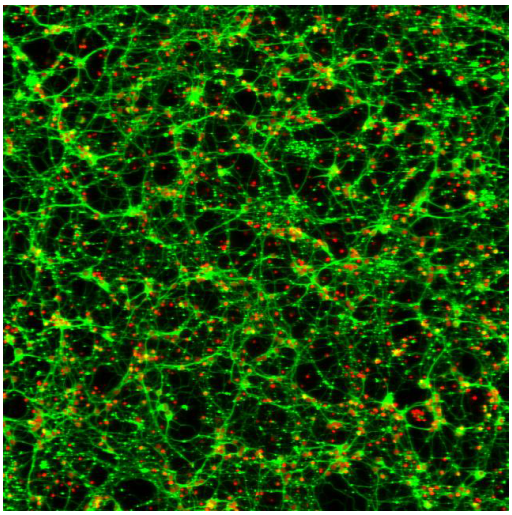

Figure 1: A typical dense image with many "clumped" neurons and overlapped neurite structures. Image acquired with IN Cell Analyzer 3000 shows a high degree of neurite branching complexity. Image courtesy of Marjo Götte, Novartis Institutes for BioMedical Research.

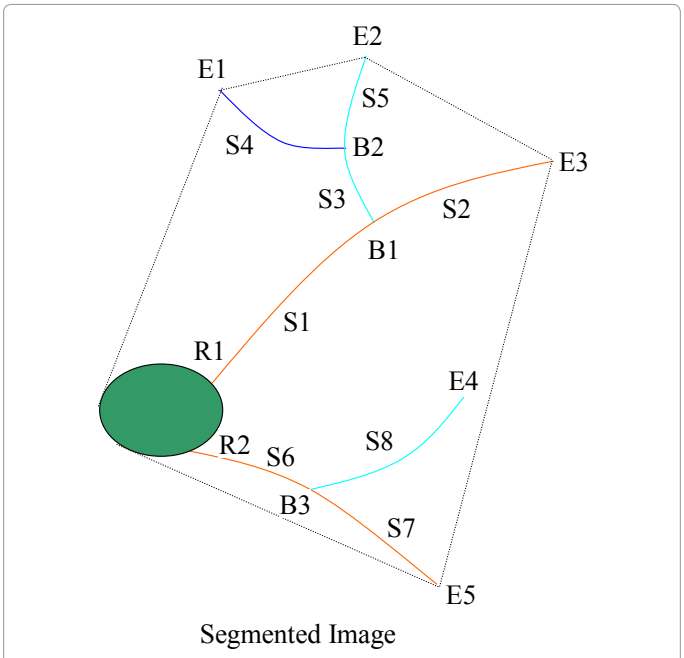

Figure 2: Segmented neurite outgrowth image.

Well-based summary for each plate: The plate summary is comprised of normalized features for each well. All features extracted from the images sampled from the same well are averaged to produce the well based normalized statistics.

To produce the above measurements, considerable computation is demanded in the image analysis. Therefore, batch processing thousands of images is a computation intensive task, taking hours even days for a single experiment.

High performance image computing

This section will address how to speed up the batch processing with multi-core based high performance image computing.

Multi-core computers have a CPU with multiple cores combining two or more independent processors into a single package composed of a single integrated circuit (IC). However, disposing of two processors does not speedup one's application automatically. According to Amdahl's law, the amount of performance gain from using a multi-core processor depends on the problem being solved and the algorithms used, as well as how they are implemented in software. Most application software packages rely on only a single core and see very limited speed improvements when run on a multi-core machine. This is because they have not been designed to take advantage of the available parallelism. In fact, developing parallel image analysis algorithms is still a challenge. In this section, a multi-core based high performance solution for neurite analysis will be described. The solution can take advantage of multiple processors and enable the conventional HCA software to process images in parallel to achieve significant performance gain.

Sequential neurite analysis procedure for a single image

Neurite analysis procedure is highly sequential, it involves three

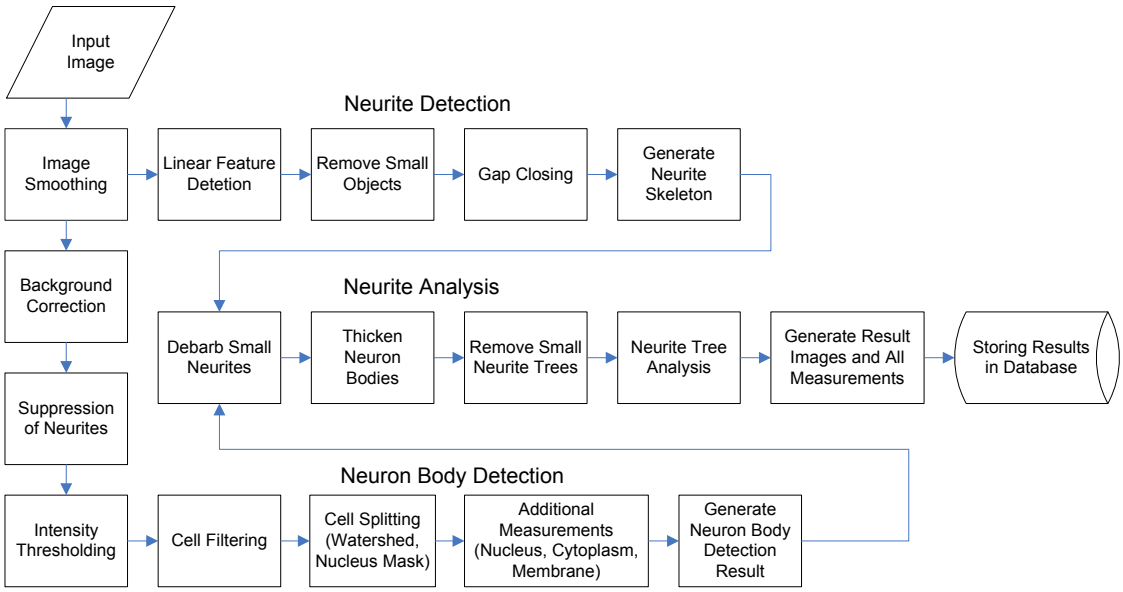

Figure 3: Sequential procedure of neurite analysis routine for a single image.

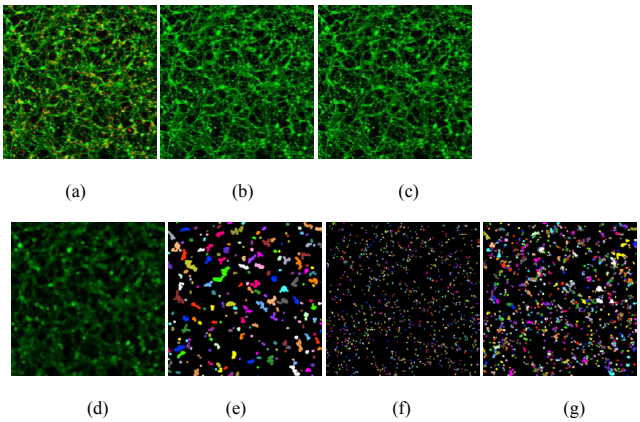

Figure 4: Neuron body detection images produced using CSIRO HCA-Vision software package. (a) The three-channel raw image, (b) Neurite channel image after smoothing, (c) Neurite image after background correction, (d) Neurite image after suppressing neuritis, (e) neurite image after thresholding, (f) segmented nucleus image in which the nuclei have been declumped and will be used as seeds for touching neuron body splitting, (g) segmented and labeled neuron bodies.

steps: (1) neuron body detection, (2) neurite detection, and (3) neurite analysis (Figure 3).

Neuron body detection

The neuron body detection aims at identifying and marking the neuron bodies. This includes smoothing the raw image to distribute the intensity more evenly within the neuron bodies; background correction to remove global trends in the background intensity; suppressing small structures such as neurites; intensity thresholding to detect neuron bodies; detecting nuclei to be used as masks to split touching neuron bodies; filtering neuron bodies when mixed cell types exist or applying certain cell selection criteria; producing neuron body detection label image and cell based measurements [8] (Figure 4a-4g).

Neurite detection

The neurite detection aims at detecting neurite structures [9]. The detection procedure includes image smoothing to remove noise within the image; linear feature detection to segment neurite structures; removing small objects which are not of interest; closing gaps between detected neurite endpoints; and generating neurite skeleton image [10] (Figure 5a-5f).

Neurite analysis

Performing a neurite analysis is to trace neurites and to associate neurites with the corresponding neurons. The tracing consists of debarbing small neurites; thickening neuron bodies to connect neighboring neurites which would otherwise be disconnected from the neuron bodies and be removed as orphan neurites; removing small neurite trees if they are not of interest; neurite tracing to generate tree statistics such as branching layers, primary, secondary and tertiary layer structures; and producing all tree analysis measurements [11] (Figure 6a-6e).

Upon completion of neurite analysis for an image, all measurements are piped into a structured database for further investigation. The whole procedure, from loading the input image to saving results in the database, is highly sequential. In the processing sequence, the input image for each processing step is the result image of its previous step. The following section will discuss how to speed up this sequential process.

Automatic parallel batch processing

In drug discovery, HCA usually involves batch processing of tens of thousands of cell images, which relies fundamentally on automation [12].

When batch processing neurite outgrowth images, the sequential procedure is repeated until all images have been processed. As the neurite analysis is highly data dependent, it is not an easy matter to parallelize the processing for each image across multi-core processors. This is because the image data is large in size, coordinating access to the intermediate result images generated from the intermediate processing steps and other shared resources, and notifying progress of the individual image processing steps across different processors require

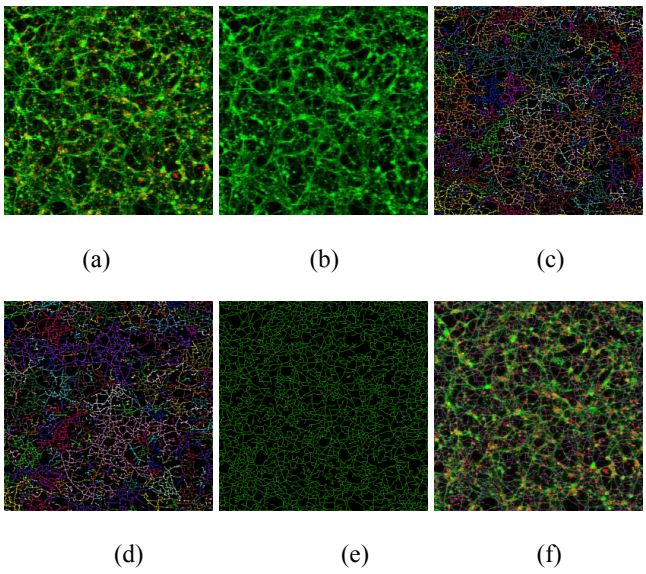

Figure 5: Neurite detection images produced using CSIRO HCA-Vision software package. (a) The three-channel raw image, (b) Neurite channel image after smoothing, (c) Neurite image after applying linear feature detection algorithms, (d) Neurite image after removing small neurites, (e) neurite image after closing gaps between neurites, (f) detected neurites overlaid on the raw image.

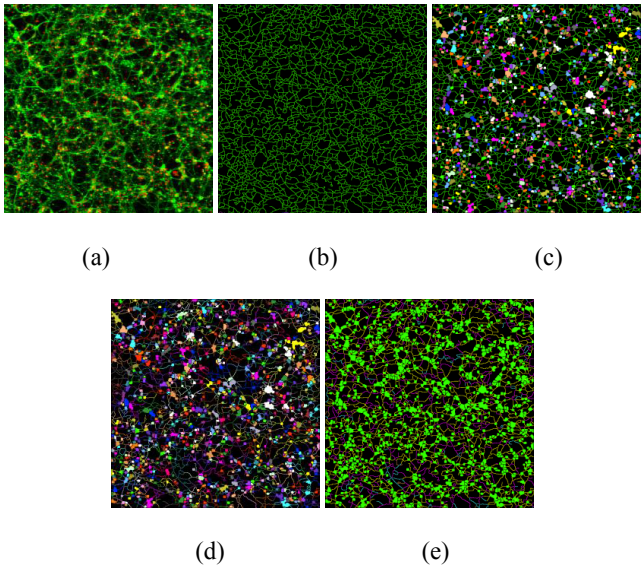

Figure 6: Neurite analysis images produced using CSIRO HCA-Vision software. (a) The three-channel raw image, (b) Neurite channel image after debarbing small neurites, (c) Neurite image after thickening neuron bodies, (d) The segmented neurons and neurite trees labeled by the neuron body to which they are deemed to belong, (e) The segmented neurons and neurite trees labeled with different branching levels.

significant amount of communications and result in heavy overhead.

Given the large problem sizes contributed by the number of images and complexity of image analysis, a vertical task partition strategy is employed to simplify the parallelization work. The main idea of the strategy is to allocate one image to a single processor at a time; therefore, each image is processed sequentially on a processor; multiple images are allocated to multi-cores and are processed in parallel.

With the proposed solution, the number of processors available on the computer on which our High Content Analysis (HCA) software runs is automatically detected, and the equal number of work threads is created to process images in parallel. The time spent in processing an image on a processor depends on the complexity of individual images. Upon completion of image processing on a processor, a new image will be automatically assigned to the processor. This process continues until all images are processed. To coordinate the access to shared resources such as the database and Graphical User Interface (GUI) components by individual threads, flow control is employed in the parallel batch processing.

In Figure 7, a 4-core computer is used as an example to illustrate the flowchart of the proposed mechanism for automatic parallel batch processing. It shows the image allocation, data management, and the batch processing loops. The batch processing starts with preparing an image processing list, then loads image processing parameter profiles. The job scheduler allocates different images in the list to separate processors. Upon completion of processing each image, the features extracted from the image are saved into the database and the batch processing progress is refreshed in the GUI to notify a user of the batch processing progress. A new image will be allocated to the processor on which the image processing is just completed. This iteration continues until all images in the list are processed.

The job scheduler is a robust, efficient, and scalable engine designed to use cooperative scheduling and work-stealing algorithms to achieve fast, efficient, and maximum CPU utilization. It can scale well on multiple processors and dynamically adapt and distribute images over the separate processors.

To implement the automatic parallelization, a parallel programming tool for data and task parallelism is adopted [13]. The tool enables software developers to build multi-core capable applications using existing code and compilers. It provides library based support for building parallel version of an application using existing code.

Optimization of database operations

The batch processing results for individual images are eventually piped into the database. When parallelizing the batch processing, we also identified that the database manipulation represented a bottleneck. As multi-processors cannot access the database at the same time, flow control is incorporated in the parallel processing. When a processor is accessing the database, a "lock" is obtained and released when the database operation is completed. To minimize the lock time, the database operation is optimized to reduce the number of locks.

As aforementioned, some images have over 1000 neurons. At the end of the processing of each image, all cell-based measurements are grouped into cell-based records and saved into the database. Each record has 34 fields representing 34 cell based features. Inserting these cell-based records individually is very slow and consumes considerable CPU time. We have conducted some experiments on two high end computers to optimize the database operation performance. The computers include Dell T7400 with 4-cores and 4GB RAM, and a high end Dell Xeon computer with 4 Intel® Xeon™ 3.2 Ghz processors and 8GB of RAM. The experimental results in Figure 8 show that inserting different numbers of cell based records using one insert statement takes different amounts of time. The database operation time is optimal for the two high end computers when saving about 25 records per insert statement. This is due to the fact that database connection, sending and parsing a query takes 5-7 times of the actual data insertion, depending on the record size (Figure 8).

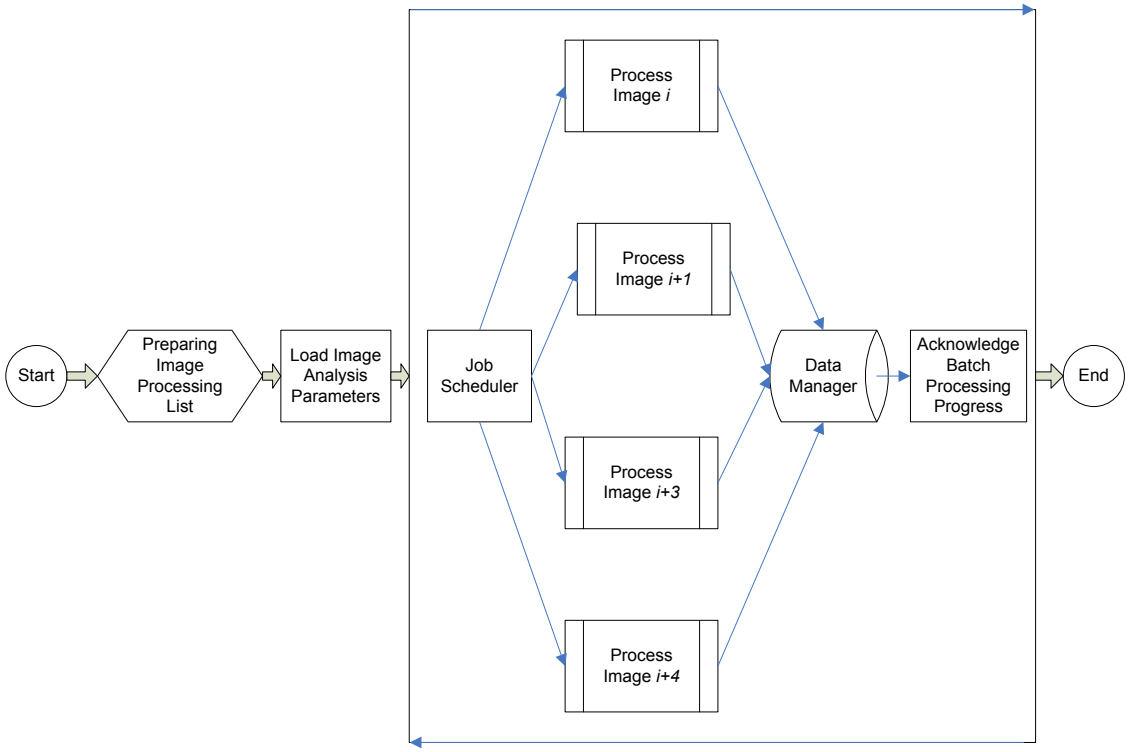

Figure 7: Flowchart of the Proposed Parallel batch processing.

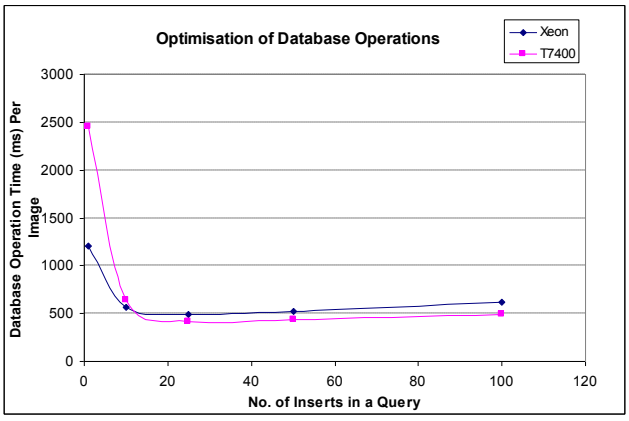

Figure 8: Experimental results of multiple cell records per query on different computers.

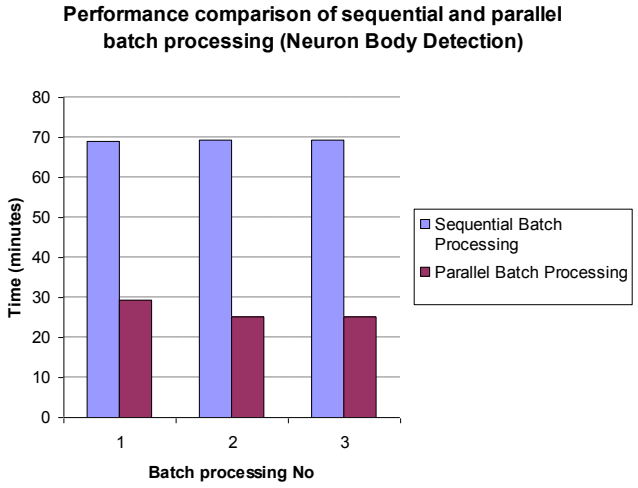

Figure 9: Execution time comparison of sequential and parallel batch processing for neuron body detection.

Results

To evaluate the performance of the parallel batch processing, experiments have been carried out on the high end Dell Xeon computer. Both neuron body detection and neurite analysis were tested for a 96-well plate of images with 6 images per well, altogether, 396 images. These images have a dimension of 1280×1280, and two channels. The first is the neuron body and neurite channel, and the other is the nucleus channel.

The test was conducted three times for both sequential and parallel processing. With the proposed parallel batch processing, significant performance improvement has been achieved. Overall, the execution time of the parallel batch processing has been reduced to 38% of the original sequential batch processing for neuron body detection, and 46% for neurite analysis. Figures 9 and 10 shows the performance comparison of the sequential and parallel batch processing for the neuron body detection and neurite analysis, respectively. The time difference among three different executions for parallel and sequential batch processing may be caused by other background Operating System tasks running on the computer.

Discussion and Conclusions

In this paper, we have presented a multi-core based batch

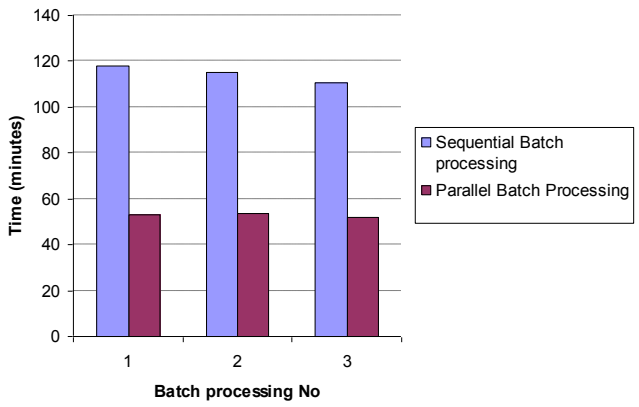

Figure 10: Execution time comparison of sequential and parallel batch processing for neurite analysis.

processing solution that can significantly improve the drug discovery efficiency. The solution provides an efficient way to process large image datasets. The solution can automatically scale to additional cores and future multi-core processors. By identifying the bottleneck of the batch processing and implementing a parallel image analysis procedure, we have established a solid and efficient HCA framework.

The proposed solution employs an automatic parallelization engine that automatically dispatches the batch processing tasks. The one-image-per-processor protocol can simplify the parallelization for data dependent computation problems and minimize the development effort in migrating sequential image analysis algorithms to a parallel form.

All statistical features extracted from the image are piped into a structured database for more sophisticated data analysis. To improve the database operation performance, some database manipulations, such as multiple data records insertion, have been optimized to maximize the batch processing throughput.

To verify the proposed solution, a full plate of images, with 96 wells and 6 images per well, have been screened. The experimental results are validated and evaluated by comparing the performance of the proposed approach with the conventional batch processing. With the proposed approach on a quad-core machine, the batch processing time for neuron body detection has been reduced to 38% of the original, and 46% for neurite analysis.

The parallelization strategy and subsequent optimizations have yielded considerable speedup and excellent resource utilization. There is no doubt that the proposed solution has potential to increase the throughput of High Content Screening, improve the workflow in HCS laboratories and reduce the cost in drug development.

The multi-core based solution has some limitations and conditions. First of all, all image processing routines have to be made thread safe. No global or static variables can be used in the routines, which depend only on the arguments passed in. No logical and data dependence is allowed among different work threads running on different cores. Flow control shall be applied when accessing shared resources such as file I/O, GUI components and databases. The amount of speedup achieved depends on how many cores are available, but is not strictly proportional to the number of cores.

The proposed approach can be applied in other data and compute-intensive applications as well. This can bring high performance to a single desktop computer and has the potential to make significant difference in the cost and quality of scientific computations and simulations.

Acknowledgement

The authors would like to thank Marjo Götte, Novartis Institutes for Bio Medical Research, for permission to use the neurite outgrowth images presented in this paper.

References

1. Sun C, Vallotton P (2009) Fast Linear Feature Detection Using Multiple Directional Non-Maximum Suppression. J Microsc 234: 147-157.
2. Lewis T, El-Rewini H (1992) Introduction to Parallel Computing, Prentice-Hall, Inc.
3. Beynon MD, Catalyurek U, Chang C, Sussman A, Saltz J (2001) Distributed Processing of Very Large Datasets with DataCutter. Parallel Computing 27: 1457-1478.
4. Blumofe R, Leiserson C (1994) Scheduling Multithreaded Computations by Work Stealing .Proceedings of the 35th Annual Symposium on Foundations of Computer Science: 356-368.
5. Cambazoglu B, Sertel O, Kong J, Saltz J, Gurcan M, et al. (2007) Efficient Processing of Pathological Images using the Grid: Computer-Aided Prognosis on Neuronblastoma. Proceedings of the 5th IEE Workshop on Challenges of large Applications in distributed environments: 35-41.
6. Wang D, Lagerstrom R, Sun C, Bischof, Vallotton P, et al. (2010) HCA-Vision: Automated Neurite Outgrowth Analysis. J Biomol Screen 15: 1165-1170.
7. Rao A, Cecchi G, Magnasco M (2007) High performance computing environment for multidimensional image analysis. BMC Cell Biology 8: 1471-2121.
8. Hartley T, Catalyurek U, Ruiz A, Francisco Igual, Rafael Mayo, et al. (2008) Biomedical Image Analysis on a Cooperative Cluster of GPUs and Multicores. Proceedings of the 22nd annual international conference on Supercomputing: 15-25.
9. CSIRO HCA-Vision website: http://www.hca-vision.com.
10. Vallotton P, Lagerstrom R, Sun C, Buckley M, Wang D, et al. (2008) Automated analysis of neurite branching in cultured cortical neurons using HCA-Vision. Cytometry A 71: 889-895.
11. Wang D, Bourke D, Domanski L, Vallotton P (2009) Multicore-Based High Performance Image Analysis for Batch Processing in Drug Discovery, 18th World IMACS/MODSIM Congress, Cairns, Australia: 1080-1086.
12. Kikinis R, Warfield S, Westin C (1998) High Performance Computing(HPC)in Medical Image Analysis (MIA) at the Surgical Planning Laboratory (SPL). High Performance Computing Asia 98: 290-297.
13. Trease H, Farber R, Wynne A, Trease L (2008) High-Performance Video Content Analysis using Hybrid, Multi-Core Processors. IASTED Proceedings of Signal and Image Processing: 632-636.