M. Tech. (Computer Science) Dissertation

Algorithms for Biological Cell Sorting

A dissertation submitted in partial fulfillment
of the requirements for the award of
M.Tech.(Computer Science) degree

by

Soumyottam Chatterjee
Roll No: CS0810

under the supervision of

Dr. Arijit Bishnu

Advanced Computing and Microelectronics Unit

Indian Statistical Institute
203 Barrackpore Trunk Road
Kolkata - 700 108

Acknowledgement

At the end of this course, it is my pleasure to thank everyone who has helped me along the way.

First of all, I want to express my sincere gratitude to my supervisor Dr. Arijit Bishnu for introducing me to the wonderful world of Discrete and Combinatorial Geometry and starting me on these fascinating problems. I have learnt a lot from him. For his patience, for all his advices and encouragement and for the way he helped me think about problems with a broader perspective, I shall always be very grateful.

I would like to thank all the professors at ISI who have made my educational life truly exciting and helped me gain a deeper and more nuanced understanding of Computer Science. I shall like to offer special thanks to Prof. Subhash Chandra Nandy who first inspired me to study Combinatorial Optimization, and Prof. Palash Sarkar to who I have always turned whenever in need of wise guidance.

I would like to thank everybody at ISI for providing a wonderful atmosphere for pursuing my studies, and more genrally, for the pursuit of happiness. I thank all my classmates who have made both the academic and non-academic experiences delightful. Special thanks are owed to my friends Ravikishore, Ambrish, Kaushik, Saurabh, Sumanta, Bahadur, Arvind, and many others who made my campus life so enjoyable. I should like to grab this opportunity to express my deep affection for and immense gratitude to two of the research scholars, Mr. Sumit Kumar Pandey and Ms. Suchismita Roy, in whom I have found an elder brother and an elder sister respectively. It has been great having them around at all times, good or bad.

My most important acknowledgement goes to my family and friends who have filled my life with joy, and made it worth living. I thank my parents whose boundless showers of love, care, and affection have always provided a haven of peace and tranquility in this otherwise ruthless world. I should not let go of this opportunity to acknowledge the incomparable magnanimosity of the person who have always given me more than I wanted, could ever hope for, or deserved. He has ceaselessly encouraged me to pursue my passions and instilled a love of knowledge in me. I shall just humbly express my gratitude to my uncle, Mr. Debshis Bhattacharyya, who has been like a second father to me. My last, but by no means the least, tribute goes to my elder brother, Mr. Debottam Chatterjee, a chidhood idol, and perhaps the principal architect in the shaping of my character and persona.

I am greatly indebted to my friends Agnik, Saurav, Saikat, Aritra, Am-

rita, Rajarshi, and Ushak for their endless supply of encouragement, moral support and real fun moments. Finally I thank Mr. Sutanu Mitra, from who I learned the beautiful language of English, and much more than that. He has been a “friend, philosopher, and guide”, in every sense of that much overused phrase.

Contents

Chapter 1

Introduction

1.1 Problem I: The Bichromatic Biological Cell Sorting Problem

1.1.1 Motivation

Rare cell population, e.g. adult stem cell, is available in very small quantities in samples that also have limited supply. Automatic cell sorting and isolation for recovery of such live cells is a challenging task. The method involves an enrichment step by magnetic or fluorescent cell sorting followed by manual or automatic cell picking or analysis [11]. Thus, applications in the medical, biological and pharmaceutical fields like stem cell research, cell therapy and cell based diagnostics need both microorganism detection and manipulation [26], [18], [14], [2], [19].

A Lab-on-a-Chip (LOC) is a device that can integrate several laboratory functions on a single chip of very small size. LOCs can handle very small fluid volumes. LOCs with sensing, processing and actuation functions can serve this purpose [17]. Microorganisms can be manipulated or displaced from their location using dielectrophoresis (or DEP). DEP is a physical phenomenon in which a force is exerted on neutral particles when it is subject to non-uniform electric field. The microorganisms can also be detected using DEP cage approach [18] and impedance sensing [17]. Differences in permittivity and conductivity between the particles and the suspending medium is used for detecting and then manipulating the microorganisms. The manipulation is done by applying electric fields using DEP. Static DEP cages have been developed that can trap live individual cells into closed potential cages [11].

To sum up the process of LOC, we have the power of detecting and manipulating or moving microorganisms. The microorganisms are manipulated

using voltage differentials (electric fields) and moved so that they can be collected at desired locations. The samples are prepared on a cell array by culturing it with some reagent so that normal or desired samples can be differentiated from undesired ones. These are then manipulated using electric fields so that they are separated and trapped and collected at the corresponding DEP cages or receptors. By manipulation, we mean the microorganisms on the cell array are displaced towards their corresponding receptor. This way the cell array is exhausted of the microorganisms. Once the microorganisms are collected at their receptors, several biological tests may be performed on them. Thus, for an abstract model of the problem, we can assume the cell array to be represented by a matrix where each cell can be any of the three types: empty, desired and undesired. We have to empty the cell array by pushing the desired and undesired cells to their respective receptors. A similar problem is studied in [12].

1.1.2 Problem Definition

We are given a matrix $\mathcal{A}$ of $M \times N$ cells. Each such cell (i, j) $(0 \leq i \leq M-1$, $0 \leq j \leq N-1)$, can have 3 possible values: Red (denoted as $\mathbf{R}$), Blue (denoted as $\mathbf{B}$) and Empty (denoted as $\mathbf{E}$). Let N_r denote the number of $\mathbf{R}$ cells, N_b denote the number of $\mathbf{B}$ cells and N_e denote the number of $\mathbf{E}$ cells. So, $N_r + N_b + N_e = M \times N$. We denote the cell in the i^{th} row and j^{th} column as (i, j). For a cell (i, j), we define its neighbourhood to be the set of cells $\mathcal{N} = \{(i, j-1), (i, j+1), (i+1, j), (i-1, j)\}$ if (i, j) does not lie on the horizontal or vertical boundaries of $\mathcal{A}$. If (i, j) lies on the horizontal or vertical boundaries or the corners, then the neighborhood of (i, j) is an appropriate subset of $\mathcal{N}$, e.g., the neighborhood of $(0, N-1)$ is $\{(0, N-2), (1, N-1)\}$ and the neighborhood of $(1, N-1)$ is $\{(0, N-1), (1, N-2), (2, N-1)\}$. A cell with value $\mathbf{E}$ can exchange its value with any of its neighbouring cells. A cell with value $\mathbf{R}$ or $\mathbf{B}$ can exchange its value only with a neighbouring $\mathbf{E}$ cell. In addition, a cell with value $\mathbf{R}$ (resp. $\mathbf{B}$) can "merge" with a neighbouring cell if and only if that cell has value $\mathbf{R}$ ($\mathbf{B}$ resp.), with the value of the cell unchanged after the "merging" operation. Such exchanges are performed with the help of a voltage differential applied across such neighbouring cells. This can be done by using a manhattan wire layout beneath the cells. Such exchanges allow a cell to reach to its corresponding receptor. An $\mathbf{R}$-receptor is located outside the matrix, adjacent to the cell $(0, 0)$. Any $\mathbf{R}$ in this cell is removed by the $\mathbf{R}$ receptor, and hence can be replaced in cell $(0, 0)$ by an $\mathbf{E}$. Similarly, a $\mathbf{B}$ receptor is located outside the matrix, adjacent to the cell $(N-1, 0)$. Any $\mathbf{B}$ in this cell gets removed by the $\mathbf{B}$-receptor, and is replaced in cell $(N-1, 0)$

by an **E**. So, the number of **E** cells increases in the matrix as the emptying process goes on. The **R** and **B** receptors provide a mechanism for emptying **R** and **B** cells from the matrix. Figure 1.1.2 shows an example.

$$\begin{array}{cc} R \sqsupset & \\ & \\ & \\ B \sqsupset & \end{array} \begin{pmatrix} E & B & B & R \\ B & R & R & B \\ B & B & R & B \\ R & R & B & R \end{pmatrix}$$

Figure 1.1: An example of a cell array with a **R**-receptor and a **B**-receptor. Here, $M = 4, N = 4, N_e = 1, N_r = 7, N_b = 8$.

We seek to minimize the number of exchanges, and hence, the number of voltage differentials, to empty the cell array. In Chapter 2, we prove that this minimization problem is NP-hard, and in Chapter 3, we propose an approximation algorithm for the special case when a cell can have values only **E** or **R**, and the goal is to push all the **R** cells to its receptor.

1.2 Problem II: The Sign Annihilation Problem

We are given a matrix $\mathcal{A}$ of $N \times N$ cells . Each such cell (i, j) $(1 \leq i \leq N, 1 \leq j \leq N)$ can have either of 2 possible values or signs: $+$ and $-$, or it can be *empty*, denoted by the value **E**. The *neighbourhood* of a cell is defined in the same fashion as in the previous problem. A cell with a $+$ or $-$ sign can exchange its value with an empty neighbouring cell. In addition if a signed cell exchanges its value with a neighbouring cell having the opposite sign, we say that an *annihilation* took place and it results in both the cells being emptied, i.e., after an annihilation both the participant cells have value **E**. We aim to minimize the total no. of exchanges needed to empty the $N \times N$ matrix $\mathcal{A}$ having the same no. n of $+$ and $-$ cells. In Chapter 4 we propose a heuristics for this problem.

Chapter 2

Hardness of the Bichromatic Biological Cell Sorting Problem

2.1 Introduction

The main theorem of this chapter is

Theorem 2.1.1. *The minimization problem as defined in 1.1, i.e., the Bichromatic Biological Cell Sorting Problem, is NP-hard.*

We prove a stronger statement, namely the problem of minimizing the no. of moves when there is only **R** or **E** cells and only one **R** receptor is NP-hard. We specifically describe this problem for the convenience of the reader.

Definition 2.1.2. (The Monochromatic Biological Cell Sorting Problem) We are given a matrix $\mathcal{A}$ of $M \times N$ cells . Each such cell (i, j) $(0 \leq i \leq M-1, 0 \leq j \leq N-1)$ can have two possible values: Red (denoted as **R**), and Empty (denoted as **E**). The *neighbourhood* of a cell is defined as in 1.1.2. A cell with value **E** can exchange its value with any of its neighbouring cells. In addition, a cell with value **R** can "merge" with a neighbouring cell having value **R**, with the value of the cell unchanged after the "merging" operation. Such exchanges allow a cell to reach to its corresponding receptor. An **R**-receptor is located outside the matrix, adjacent to the cell $(1, 1)$. So, the number of **E** cells increases in the matrix as the emptying process goes on. Any **R** in this cell is removed by the **R** receptor, and hence can be replaced in cell $(0, 0)$ by an **E**. Figure 2.1.2 shows an example.

We seek to minimize the number of exchanges, and hence, the number of voltage differentials, to empty the cell array.

$$R \sqsupset \begin{pmatrix} E & R & E & R \\ E & E & R & E \\ E & R & R & R \\ R & R & R & E \end{pmatrix}$$

Figure 2.1: An example of a cell array with an **R**-receptor

We reduce the known NP-hard problem *Rectilinear Steiner Arborescence (RSA)* [24] to this problem.

2.2 The Rectilinear Steiner Arborescence Problem

2.2.1 Introduction

Definition 2.2.1. The Rectilinear Steiner Arborescence (RSA) problem is "Given a set N of n nodes lying in the first quadrant of E^2, find a minimum length directed tree (*Rectilinear Steiner Minimum Arborescence*, or *RSMA*) rooted at the origin and containing all nodes in N, composed solely of horizontal and vertical arcs oriented only from left to right and from bottom to top."

The rectilinear Steiner arborescence problem was first studied by Nastansky, Selkow, and Stewart [21] in 1974. They proposed an integer programming formulation, which has exponential time complexity. In 1979 Laderia de Matos [16] proposed an exponential time dynamic programming algorithm. In 1985, Trubin [25] claimed the RSA problem can be solved in polynomial time. In 1992, Rao, Sadayappan, Hwang, and Shor [23] showed that Trubin's algorithm was incorrect and presented an $O(n \log n)$ time approximation algorithm that produces an RSA of length at most 2 times the optimal [23]. In 1994, Cordova and Lee [8] extended the heuristic to points in all four quadrants with the same time complexity. In 1997, Cho [5] again claimed that the RSA problem can be solved in polynomial time using a min-cost max-flow approach. Soon after, Erzin and Kahng showed Cho's claim is wrong [10]. In 2000, Lu and Ruan [15], motivated by the polynomial time approximation scheme (PTAS) of Arora [3], designed a PTAS for the RSA problem. Their PTAS runs in time $O\left(n^{O(c)} \log n\right)$ and produces an RSA of length at most $(1 + 1/c)$ times the optimal. Finally it was proved to be strongly NP-complete in 2005 by Shi and Su [24].

Because an RSMA is a shortest distance tree of minimum total length, it has important applications in VLSI routing. Cong, Leung, and Zhou [7] showed that routing trees based on RSMAs may have significantly less delay than those based on the traditional Steiner trees. Many researchers proposed efficient heuristics and exponential time exact algorithms for the RSA problem [1], [6].

2.2.2 Some Basic Results

We first fix some conventions and notations used later in the text.

Conventions / Notations:

- The cardinality of a set S is denoted by $|S|$.
- $\mathbb{R}$ denotes the set of real numbers.
- $\mathbb{Z}$ denotes the set of integers.
- $\mathbb{N}$ denotes the set of natural numbers.
- E denotes the Euclidean plane, or the two-dimensional plane on $\mathbb{R}^2$.
- Given a set of points or nodes N in an instance of the RSA problem, we call the nodes in N *sinks*.
- Unless otherwise stated, all points in N lie in the first quadrant of E^2 and distances are measured in the L_1 metric.
- A path is composed of segments which are always directed either upward or to the right.
- $||p||$ denotes the sum of coordinates $x_p + y_p$ for a point p.
- For an arc e, $l(e)$ denotes the length of the arc e in the L_1 metric, i.e., if e connects points p and q, $l(e) = |x_p - x_q| + |y_p - y_q|$.
- $l(G)$ denotes the total length of arcs in a graph G.
- For two points p and q on $G(N)$, where $G(N)$ is as defined in 2.2.11, $min(p, q)$ is the grid point with coordinates $(min\{x_p, x_q\}, min\{y_p, y_q\})$.

We now define the various terminologies related to the RSA problem.

Definition 2.2.2. An arborescence is a rooted tree with all edges directed away from the root.

Definition 2.2.3. An RSA is a directed arborescence rooted at the origin, all of whose leaves are nodes of N, containing all the nodes of N, and having the property that if an edge e joins p to q, then $x_p \leq x_q$ and $y_p \leq y_q$. That is, all edges point "northeast".

Definition 2.2.4. A node which has two outlinks is called a Steiner node if it is not a sink.

Remark 2.2.5. Only sinks and Steiner nodes are legitimate nodes in an RSA.

Definition 2.2.6. If a node has two outlinks, we label the vertical one "left" and the horizontal one "right".

Definition 2.2.7. An RSMA for a set N is an RSA A with minimum $l(A)$.

Observation 2.2.8. *Let $\tau(A)$ denote the depth-first tour of a tree A which starts at the origin and goes first to the leftmost node whenever there is a choice. It is easily verified that $\tau(A)$ covers every arc twice (in opposite directions) before it comes back to the origin. Furthermore, the visiting sequence of $\tau(A)$ establishes a linear order $\pi(A)$ for nodes in N. (This is simply the preorder of the tree A.)*

Lemma 2.2.9. *For any RSMA A, if node u lies to the northwest of node v, i. e., if $u_x < v_x$ and $u_y > v_y$, then u precedes v in $\pi(A)$.*

Proof. Let w be the lowest common ancestor of u and v. Since edges in an RSMA cannot cross, the path from w to u must be the left outlink from w, and the path from w to v the right one. Thus, u is visited first and so precedes v in $\pi(A)$. □

Lemma 2.2.10. *In any RSMA A, let u and v be two nodes having the same parent node z. Then $z = min(u, v)$.*

Proof. Suppose to the contrary that $min(u, v) = w \neq z$. Then $z_x \leq w_x$ and $z_y \leq w_y$, so there is a directed path from z to w. This implies
$l(z, u) + l(z, v)$
$= l(z, w) + l(w, u) + l(z, w) + l(w, v)$
$> l(z, w) + l(w, u) + l(w, v)$.
So we can substitute the three arcs $[z, w]$, $[w, u]$, and $[w, v]$ for the two arcs $[z, u]$ and $[z, v]$ to shorten the tree, a contradiction to the assumption that A is an RSMA. □

Definition 2.2.11. For a given set N of sinks, $G(N)$ denotes the grid that includes a vertical path from the x-axis and a horizontal path from the y-axis through every sink, and is bounded by the x-axis, the y-axis, the horizontal path through the highest sink, and the vertical path through the rightmost sink.

We have the following theorem:

Theorem 2.2.12. *There exists an RSMA which uses only arcs of G(N).*

Proof. Suppose that A is an RSMA which does not use only arcs of $G(N)$. Find the node p of A which is not a node of $G(N)$ and which is farthest from the origin (i.e., maximum $||p||$). By Lemma 2.2.10, $p = min(u, v)$, where u and v are the two children of p, but this means that either u or v is not a node of $G(N)$, a contradiction. □

Definition 2.2.13. Let L_z denote the line $x + y = z$ in the first quadrant.

Definition 2.2.14. A set P of points is a cover for a set Q of points if for every $q \in Q$, $\exists$ a $p \in P$ which has a path to q. For given Q and z let Q_z denote the subset of Q lying above L_z. Then let $MC(Q, z)$ denote a minimum cardinality set of points on L_z that covers Q_z.

Lemma 2.2.15. *Any RSA for N intersects L_z at least $MC(N, z)$ times for all z.*

Proof. This follows immediately from the definition of $MC(N, z)$. □

The next lemma deals with the fact that an instance of *RSMA* can consist of points having non-integer (or even irrational) co-ordinates while in Problem 2.1.2 the cells of the matrix $\mathcal{A}$ can have only integer-coordinates.

Lemma 2.2.16. *For any instance of the RSA problem with the set N of n nodes lying in the first quadrant of E^2, we can get a mapping $f : N \to N'$ such that the linear order as stated in 2.2.8 obtained for the RSMA for N is an invariant under this mapping.*

Proof. We only note that to preserve the structure of the RSMA, it is necessary only to maintain the linear ordering of the distances between the nodes of $G(N)$. □

Henceforth we shall assume all the points in N in an instance of *RSA* to have integer coordinates.

2.3 Reduction from the RSA problem to the Monochromatic Biological Cell Sorting Problem

Definition 2.3.1. A non-deviating path is one moving through which a red cell exchanges its value with only a left neighbour or an upper one.

Before going into the actual reduction from the *RSA* problem to the Monochromatic Biological Cell Sorting Problem, we note the following.

Observation 2.3.2. *For any sequence of exchanges that empties the matrix in the Monochromatic Biological Cell Sorting Problem, there is a sequence of exchanges with a cost not more than the original where every red cell follows only* non-deviating *paths.*

The intuition is that since the receptor is at the top-left corner of the matrix, it is always "profitable" to move a red cell (by exchanging its value with an *empty* neighbour) leftwards or upwards.

Now we are in a position to prove the theorem

Theorem 2.3.3. *The RSA problem as defined in Definition 2.2.1 is polytime reducible to the Monochromatic Biological Cell Sorting Problem as defined in Definition 2.1.2.*

Proof. We shall look at the respective decision versions of the *RSA* problem and the Monochromatic Biological Cell Sorting Problem. Given an instance of the RSA problem, i.e. a set N of points in the first quadrant of $\mathbb{Z}^2$, we construct an $X \times Y$ matrix $\mathcal{A}(N)$, where $X = max_{p_x} \{p_x : p = (p_x, p_y) \in N\}$ and $Y = max_{p_y} \{p_x : p = (p_x, p_y) \in N\}$. The value $\mathbf{E}$ is assigned to all cells of $\mathcal{A}$ by default. For each $p = (p_x, p_y) \in N$, we now assign the value $\mathbf{R}$ to the cell at the ${p_x}^{th}$ row and ${p_y}^{th}$ column. Clearly this construction takes $O(n)$ time where $n = |N|$. We prove that for an instance N of the RSA problem, there shall exist a rectilinear Steiner arborescence of length at most k if and only if $\mathcal{A}(N)$ can be emptied by at most k-exchanges.

For the sufficiency part, we observe that by virtue of Observation 2.3.2 given a sequence of moves with k-exchanges, there is a sequence of moves with a total no. of exchanges at most k where a *red* cell exchanges its value only with a left neighbour or an upper one. Tracing the paths of the *red* cells towards the receptor and putting edges in the reverse direction give a rectilinear rooted Steiner tree with all edges directed away from the origin, i. e., precisely a rectilinear Steiner arborescence, and which has length at most k.

For the necessity part, we observe that corresponding to an RSA for N, we get a sequence of moves for the red cells in $\mathcal{A}(N)$ by identifying a cell (i, j) in the Monochromatic Biological Cell Sorting Problem with the point (i, j) in the first quadrant of the Euclidean plane in the *RSA* problem, and identifying the arc $(i_1, j_1) \longrightarrow (i_2, j_2)$ in the RSA with pushing a *red* cell from the $(i_2, j_2)^{th}$ location in the matrix $\mathcal{A}(N)$ to the $(i_1, j_1)^{th}$ location through only leftward or upward moves. Only one thing is left to be taken care of,

that is, only leaves are moved, i. e., before we move a node to its parent (location), all the chips in its subtree are brought to that node. This ensures that the same edge is not traversed more than once, and hence the total no. of exchanges does not exceed the length of the RSA.

□

the Monochromatic Biological Cell Sorting Problem having been a special case of the Bichromatic Biological Cell Sorting Problem, Theorem 2.1.1 follows from Theorem 2.3.3.

Corollary 2.3.4. *For a set of points N in the first quadrant of the Euclidean plane, all with integer coordinates, the length of the rectilinear Steiner minimum arborescence defined on that set N is equal to the minimum no. of exchanges required to empty the matrix $\mathcal{A}(N)$ as defined in the above proof of Theorem 2.3.3.*

Chapter 3

A 2-factor Approximation Algorithm for the Monochromatic Biological Cell Sorting Problem

3.1 Algorithm

In this section we propose a new heuristic which is fast and empties the matrix $\mathcal{A}$ in a short no. of exchanges l. The matrix is emptied by iteratively merging *red* cells p and q at the location $min\,(p, q)$ until a single red cell remains. The cells p and q are first brought to the cell-location $min\,(p, q)$ by the straight-line segment through the matrix $\mathcal{A}$ joining p or q and the cell at $min\,(p, q)$. If $min\,(p, q)$ is either of p or q, then that red cell need not be moved. If the position of the final red cell be the origin, i.e., the cell at (0, 0), then we are done, otherwise we just move it to the origin through a *non-deviating* path. Let N be the set of red cells. The nodes p and q are chosen to maximize $||min\,(p, q)\,||$ over all nodes in the current $||N||$.

Time complexity: This algorithm can be implemented in $O\,(\log n)$ time using a plane-sweep technique suggested by S. Fortune [23].

3.2 Proving the Approximation Ratio

From the proof of the theorem 2.3.3 it is evident that the paths traced by the *red* cells constitute a tree (an *arborescence* specifically) rooted at the cell (0, 0) and with only horizontal or vertical edges. Moreover the length of this

tree is precisely the same as the total no. of exchanges made in emptying the cell. A bound on the length of the arborescence H produced by Algorithm 3.1 is proved [23], and hence by Corollary 2.3.4 a bound on the total no. of exchanges made by Algorithm 3.1 follows.

Definition 3.2.1. We define N_z to be the set N at the point in the execution of the heuristic when we have just merged the last pair of points p, q in N with $||min\,(p,q)\,|| \geq z$.

Definition 3.2.2. W_z is defined to be the subset of points of N given by $\{p \in N : ||p|| \geq z\}$.

Lemma 3.2.3. *Let $p_i = (x_i, y_i)$ be a node in an arborescence H produced by our heuristic. There must exist a sink on the vertical path $x = x_i$ (horizontal path $y = y_i$) starting from p_i.*

Proof. Lemma 3.2.3 is trivial if p_i is a sink. We prove the general case by induction on the number of descendants of p_i. Let $p_i = min\,(p_h, p_j)$. Then $x_i = min\,(x_h, x_j)$, say x_h. By induction there exists a sink on the vertical path $x = x_h$ starting from p_h. But $y_h \geq y_i$. Hence this sink is also on the vertical path $x = x_i = x_h$ starting from p_i. A similar argument works for the y-coordinate. □

Lemma 3.2.4. *During the construction of H, for any $z \geq 0$ and $p, q \in W_z$,*

1. *$x_p \neq x_q$ and $y_p \neq y_q$,*
2. *$x_p < x_q$ if and only if $y_p > y_q$,*
3. *if $x_p < x_q$, then $x_p + y_q < z$.*

Proof.

1. Suppose that $x_p = x_q$ and, say, $y_p > y_q$. Then $min\,(p,q) = (x_q, y_q) = q$. But q lies on or above L_z and $x_q + y_q > z$, a contradiction to the assumption $p, q \in N_z$.
2. Suppose that $x_p > x_q$ and $y_p > y_q$. Then $min\,(p,q) = q$, and an argument similar to 1 leads to a contradiction.
3. Suppose that $x_p + y_q > z$. Then $min\,(p,q) = (x_p, y_q)$ lies on or above L_z, contradicting $p, q \in N_z$.

□

Lemma 3.2.5. *Let p, q be points in W_z with $x_p < x_q$. Then there exists a sink either on the horizontal path between p and (x_q, y_p) or on the vertical path between (x_q, y_p) and q.*

Proof. By Lemma 3.2.3, there is a sink p' on the horizontal path $y = y_p$ starting from p and a sink q' on the vertical path $x = x_q$ starting from q. Suppose that both $x_{p'} > x_q$ and $y_{p'} > y_q$. Then the paths $[p, p']$ and $[q, q']$ cross at (x_q, y_p) in the arborescence produced by H, a contradiction. □

We now prove

Theorem 3.2.6. $l(H) \leq 2l(RSMA)$.

Proof. First, observe that, for any RSA A, $l(A) = \int |A \cap L_z| dz$. Any RSA must intersect L_z at least $|MC(N, z)|$ times. We will show that H intersects L_z at most $2 \cdot |MC(N, z)|$ times for any z. Hence $l(H) \leq 2l(A)$ for any RSA A, implying Theorem 3.2.6.

Order the nodes in W_z according to the increasing order of the x-coordinate into $p_1, \ldots p_{m(z)}$, where $m(z) = |W_z|$. By Lemma 3.2.3 there exists a sink on the vertical path $x = x_1$ starting at p_1. Hence $MC(N, z)$ must contain a point whose x-coordinate does not exceed x_1. By Lemma 3.2.5, there exists either a sink on the horizontal path $y = y_2$ between p_2 and (x_3, y_2) or a sink on the vertical path $x = x_3$ between (x_3, y_2) and p_3. Hence $MC(N, z)$ must contain a point whose x-coordinate lies between $z - y_2$ and x_3. But $x_1 + y_2 < z$ by Lemma 3.2.4. Hence $z - y_2 > x_l$ and the above two points in $MC(N, z)$ are distinct. Similarly, $MC(N, z)$ contains a point whose x-coordinate lies between $z - y_{2j}$ and x_{2j+l} for $j = 2, 3, \ldots \lfloor m/2 \rfloor$, and these points are all distinct. Hence W_z contains at most $2|MC(N, z)|$ points, which implies H interesects L_z at most $2|MC(N, z)|$ times. Since z is arbitrary, Theorem 3.2.6 is proved. □

Chapter 4

An Efficient Heuristics for The Sign Annihilation Problem

We recall the Sign Annihilation Problem.

Definition 4.0.7. (The Sign Annihilation Problem) We are given a matrix $\mathcal{A}$ of $N \times N$ cells . Each such cell (i, j) $(1 \leq i \leq N, 1 \leq j \leq N)$ can have either of 2 possible values or signs: $+$ and $-$, or it can be *empty*, denoted by the value **E**. The *neighbourhood* of a cell is defined in the same fashion as in the previous problem. A cell with a $+$ or $-$ sign can exchange its value with an empty neighbouring cell. In addition if a signed cell exchanges its value with a neighbouring cell having the opposite sign, we say that an *annihilation* took place and it results in both the cells being emptied, i.e., after an annihilation both the participant cells have value **E**. We aim to minimize the *cost of annihilation*, which is defined as the total no. of exchanges needed to empty the $N \times N$ matrix $\mathcal{A}$ having the same no. n of $+$ and $-$ cells.

Definition 4.0.8. Given a matrix $\mathcal{A}$ of $N \times N$ cells with either $+$ or $-$ values, we define a weighted bipartite graph $G(A) = (U, V, E)$, where $|U| = |V| = n$, with a weight function $W : E \to \mathbb{N}$ on A as follows:

- $U \stackrel{\text{def}}{=}$ the set of '+'-signed cells.
- $V \stackrel{\text{def}}{=}$ the set of '−'-signed cells.
- $E = \{(u, v) : u \in U, v \in V\}$, i. e., G is a complete bipartite graph.
- $W = \{w_{i,j}\}$, where $e = (u_i, v_j)$ $(u \in U,\ v \in V)$ maps to $w_{i,j}$ under W. $w_{i,j}$ is given by $w_{i,j} \stackrel{\text{def}}{=}$ the length (in the L_1 metric) of the shortest free path between u_i and v_j. If no such path exists, then $w_{i,j} \stackrel{\text{def}}{=} \infty$.

Algorithm: We find out the *minimum weight bipartite matching* on G as defined above by the Hungarian method [13], [20]. The perfect matching thus obtained pairs each $+$ with a distinct $-$, and the annihilations follow this pairing.

Time complexity: The construction of G takes $O(|E|) = O\left(|V|^2\right)$ time. The Hungarian method solves the weighted matching problem for a complete bipartite graph with $2 \cdot |V|$ nodes in $O\left(|V|^3\right)$ arithmetic operations [22]. Hence our heuristics runs in $O\left(|V|^3\right) = O(n^3)$ time.

Remark 4.0.9. The solution obtained by our heuristic may not be the optimum, because with each annihilation, the matrix gradually becomes empty, and hence paths between opposite-signed cells may become free, thus altering (reducing, to be specific) the length of a shortest path between the cells. Thus the pairing (matching) obtained at the initial stage of the annihilation process may not be the optimum pairing.

Chapter 5

Conclusion

We studied two combinatorial optimization problems — The Bichromatic Biological Cell Sorting Problem and the Sign Annihilation Problem.

We proved that the Bichromatic Biological Cell Sorting Problem is NP-hard, moreover we proved the computational hardness for the simpler Monochromatic Cell Sorting Problem. In doing so we studied in some detail the Rectiliear Steiner Arborescence Problem. We provided a constant factor approximation algorithm for the Monochromatic version of the Biological Cell Sorting Problem.

Then we focussed on the Sign Annihilation Problem, and described an $O(n^3)$-time heuristics for this problem.

5.1 Scope of this work

5.1.1 Desired work to do in Problem I: The Bichromatic Biological Cell Sorting Problem

We note that the approximation algorithm provied in 3.1 can be extended to give a *constant-factor approximation algorithm* for the more general Bichromatic Biological Cell Sorting Problem, for which the existence of no such algorithm is known.

5.1.2 Possible avenues of progess in Problem II: The Sign Annihilation Problem

There is much left to do in the case of the Sign Annihilation Problem. The apparent simplicity of the problem hides the extremely large no. of combinatorial possibilities that lie in finding out the desired pairing. A proof of

correctness for any purported "exact" algorithm will have to deal with the hitch that as the annihilation process progresses, the configuration of the matrix changes, and so what seems like an optimal choice at any *one particular step* of the algorithm may *not* be the optimal choice *globally*.

While investigating the effect of various initial configurations on the cost of annihilation, we started with the simple case when no cell of the matrix is initially empty, i .e., for an $N \times N$ matrix $\mathcal{A}$, there are initially $\frac{N^2}{2}$ cells with $+$ sign and exactly $\frac{N^2}{2}$ cells with $-$ sign.

It is immediately observed that the lower bound on the cost is the initial no. of $+$ cells, i. e., $\frac{N^2}{2}$, and this is achieved with several initial configurations. Two examples are shown in Figure 5.1.

$$\begin{pmatrix} + & - & + & - \\ + & - & + & - \\ + & - & + & - \\ + & - & + & - \end{pmatrix}$$

$$\begin{pmatrix} + & - & + & - \\ - & + & - & + \\ - & + & + & - \\ + & - & - & + \end{pmatrix}$$

Figure 5.1: Here $N = 4, n = 8$, and the cost of annihilation is 8

Unfortunately it proves difficult to determine the upper bound on the cost of annihilation in this case, i. e., for $n = \frac{N^2}{2}$. We conjecture it to be $O\left(\frac{N^3}{4}\right)$, which is achieved in the case when the $+$ and - signs are separated into above the main diagonal and below it. Figure 5.2 gives an example.

$$\begin{pmatrix} + & + & + & - \\ + & + & - & - \\ + & + & - & - \\ + & - & - & - \end{pmatrix}$$

Figure 5.2: Here $N = 4, n = 8$, and the cost of annihilation is 17

We also conjecture that this diagonal separation gives the "worst" initial configuration, but again a proof alludes us, at least for now.

We also looked into the problem of finding the equivalence of two such matrices. Given two matrices of the same size, but with different configurations both having the same number of (initial) +-signed cells (or equivalently, the initial number of −-signed cells), we ask the question whether one can be transformed into the other by a sequence of moves. The previous work done in this area indicates that any two configurations with the same distribution of cells of two different signs can be transformed into one another by a sequence of moves so that all intermediate configurations are connected [9].

Bibliography

[1] M. J. Alexander and G. Robins, *New performance-driven FPGA routing algorithms*, IEEE Trans. Computer-Aided Design, **15** (1996), 1505–1517.

[2] S. Archer, T. T. Li, A. T. Evans, S. T. Britland, H. Morgan, *Cell Reactions to Dielectrophoretic Manipulation*, Biochemical and Biophysical Research Communications, **257** (1999), 687-698.

[3] S. Arora, *Polynomial time approximation scheme for Euclidean traveling salesman and other geometric problems*, J. ACM, **45** (1998), 753–782.

[4] Gruia Calinescu, Adrian Dumitrescu, and Janos Pach, *Reconfigurations in Graphs and Grids*, SIAM Journal on Discrete Mathematics, **22** (2008), 124-138.

[5] J. D. Cho, *Min-cost flow based min-cost rectilinear Steiner distance preserving tree construction*, Proceedings of the International Symposium on Physical Design, ACM Press, New York, 1997, 82–87.

[6] J. Cong, A. B. Kahng, and K. S. Leung, *Efficient algorithms for the minimum shortest path Steiner arborescence problem with applications to VLSI physical design*, IEEE Trans. Computer-Aided Design, **17** (1998), 24–39.

[7] J. Cong, K. S. Leung, and D. Zhou, *Performance driven interconnect design based on distributed RC delay model*, Proceedings of the 30th ACM/IEEE Design Automation Conference, 1993, 606–611.

[8] J. Cordova and Y. H. Lee, *A Heuristic Algorithm for the Rectilinear Steiner Arborescence Problem*, Technical Report TR-94-025, Department of Computer Science, University of Florida, Gainesville, FL, 1994.

[9] A. Dumitrescu and J. Pach, *Pushing squares around*, Graphs and Combinatorics, **22** (2006), 37-50.

[10] A. Erzin and A. Kahng, *private communication.*

[11] A. B. Fuchs, A. Romani, D. Freida, G. Medoro, M. Abonnenc, L. Altomare, I. Chartier, D. Guergour, C. Villiers, P. N. Marche, M. Tartagni, R. Guerrieri, F. Chatelain and N. Manaresi, *Electronic sorting and recovery of single live cells from microlitre sized samples*, Lab on a Chip, RSC Publishing, **6** (2000), 121-126.

[12] Arijit Ghosh, Rushin Shah, Arijit Bishnu, and Bhargab Bhattacharya, *Algorithms for Biological Cell Sorting with a Lab-on-a-chip*, World Congress on Nature and Biologically Inspired Computing (NaBIC '09), Coimbatore, India, 2009.

[13] Harold W. Kuhn, *The Hungarian Method for the Assignment Problem*, Naval Research Logistics Quarterly, **2** (1955), 83-97.

[14] R. Lovell-Badge, *The future for stem cell research*, Nature, **414** (2001), 88-91.

[15] B. Lu and L. Ruan, *Polynomial time approximation scheme for rectilinear Steiner arborescence problem*, Journal of Combinatorial Optimization, **4** (2000), 357–363.

[16] R. R. Laderia de Matos, *A Rectilinear Arborescence Problem*, Ph.D. dissertation, University of Alabama, Tuscaloosa, AL, 1979.

[17] G. Medoro, N. Manaresi, A. Leonardi, L. Altomare, M. Tartagni, and R. Guerrieri, *A Lab-on-a-Chip for Cell Detection and Manipulation*, IEEE Sensors Journal, **3** (2003), 317-325.

[18] G. Medoro, N. Manaresi, M. Tartagni and R. Guerrieri, *CMOS-only sensor and manipulation for microorganisms*, in proc. IEDM, (200), 415-418.

[19] T. Muller, G. Gradl, S. Howitz, S. Shirley, Th. Schnelle, G. Fuhr, *A 3-D microelectrode system for handling and caging single cells and particles*, Biosensors and Bioelectronics, **14** (1999), 247-256.

[20] J. Munkres, *Algorithms for the Assignment and Transportation Problem*, Journal of the SIAM, **5** (1957), 32-38.

[21] L. Nastansky, S. M. Selkow, and N. F. Stewart, *Cost-minima trees in directed acyclic graphs*, Z. Operations Res. Ser. A-B, **18** (1974), 59–67.

[22] Christos H. Papadimitriou and Kenneth Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall Inc., 1982.

[23] Sailesh K. Rao, P. Sadayappan, Frank K. Hwang, and Peter W. Shor, *The Rectilinear Steiner Arborescence Problem*, Algorithmica, **7** (1992), 277-288.

[24] Weiping Shi and Chen Su, *The Rectilinear Steiner Arborescence Problem is NP-complete*, SIAM Journal on Computing, **35** (2005), 729-740.

[25] V. A. Trubin, *Subclass of the Steiner problems on a plane with rectilinear metric*, Cybernetics, **21** (1985), 320–322.

[26] X-B. Wang, Y. Huang, P. R. C. Gascoyne and F. F. Becker, *Dielectrophoretic manipulation of particles*, IEEE Trans. on Industry Applications, **33** (1997).