5-2010

DNA Staged Self-Assembly at Temperature 1

Nicholas H. Guiterrez
University of Texas-Pan American

DNA STAGED SELF-ASSEMBLY AT TEMPERATURE 1

A Thesis

by

NICOLAS H. GUTIERREZ

Submitted to the Graduate School of the
University of Texas - Pan American
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2010

Major Subject: Computer Science

DNA STAGED SELF-ASSEMBLY AT TEMPERATURE 1

A Thesis
by
NICOLAS H. GUTIERREZ

COMMITTEE MEMBERS

Dr. Robert Schweller
Chair of Committee

Dr. Richard Fowler
Committee Member

Dr. Andres Figueroa
Committee Member

Dr. Zhixiang Chen
Committee Member

May 2010

ABSTRACT

Gutierrez, Nicolas H., DNA Staged Self-Assembly at Temperature 1. Master of Science (MS), May, 2010, 25 pp., 10 figures, references, 33 titles.

We introduce alternate temperature 1 self-assembly constructions of an *n* x *n* square by efficiently utilizing bins and stages to achieve desirable results. These bins are able to contain a variety of tiles or supertiles, which are then mixed together in a pre-determined sequence of distinct stages. The basic 2D tile assembly model at temperature 1 uses 2*n*-1 tile types to construct a square. The model only utilizes one bin and occurs all in one stage. We will demonstrate how the use of bins and stages will allow for the construction of these squares more efficiently.

ACKNOWLEDGEMENTS

I would like to thank my family for all of their encouragement and support throughout my academic career. Also, I would like to thank my advisor, Dr. Robert Schweller, for allowing me to work with him and introducing me to the field of bio-informatics. Lastly, I want to thank all of the graduate and undergraduate students I have studied with, for making this a fun and memorable experience.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Research in Biological Science has allowed the manipulation of DNA by constructing replicas of traditional Wang tiles [29, 30], which computational theorists can manipulate through highly efficient algorithms to produce various self-assembly techniques such as the two dimensional model by Winfree [34] and its various generalizations [1, 2, 14, 23]. These Wang tiles are four-sided, non-rotatable tiles which have glues on one or more sides. These tiles are able to attach to other tiles depending on the glues present on the attaching sides. Any glue can bond with other glues so long as they have the same number or letter of representation. These bonding tiles form larger constructions which lead up to many desired shapes [3, 9, 16, 17, 19, 20, 33]. One of the most desired shapes is the square, which has shown to be a common benchmark problem when demonstrating various self-assembly techniques [1, 2, 3, 4, 23].

We focus on temperature 1 construction for many reasons. One of the most important reasons is because it is difficult to enforce error free constructions at higher temperatures. Differences in glue strengths would create temporary incorrect attachments which ultimately create undesired assemblies. There are several techniques used to attempt a correction to this problem [5, 6, 7, 8, 13, 18, 21, 31, 32], yet through the use of temperature 1, the tiles will either bond or not bond depending if the bonding glues match.

Many self-assembly methods involve a seed tile which serves as the starting point of the assembly. Tiles attach to this seeded structure, a tile at a time, until the assembly has completed. This method of assembly is difficult to enforce [25, 26] because tiles which have not yet attached to a seed tile may still connect to each other as long as they have matching glues. Our staged self-assembly model at temperature 1 does not require a seed tile and allows attachments to form where possible.

The Staged Assembly Model

The staged tile assembly model, as introduced in [11, 27], allows a more flexible fabrication of complex shapes by reducing the tile and glue complexities, and introducing bins and stages. Staged assembly allows for a sequence of distinct tiles to be gradually introduced into a forming shape. Throughout the changes in stages, any tiles which have not attached, or any supertile which is not fully formed, is discarded before moving on to the next stage of the assembly. This model can have any number of bins, each containing tiles or supertiles, which are stored for use in later stages. Each stage is able to either add a new set of tiles to an existing bin, or combine one or more bins together, allowing them to mix into larger supertiles.

Research Results

Between previous work and personal studies, the results offer different methods to solving the problem of efficiently creating an n x n square. Both of the results obtained in our studies are modifications of previous work to improve their functionality. The staged assembly method constructs 1 x n lines defined in [10] similar to our own method of building lines without a counter, but it does not take into account how the use of bins would affect the stages required. Even though both are O(*logn*) constructions, depending on how many bins are used in an assembly, the number of stages are affected by the base of the log of *n*. The Jigsaw technique is another method which produces the same results of O(*logn*). Our n x n assembly utilizes a similar way of building a counter as shown in [27] by using geometry to build a binary counter. While previous methods involving a counter use a temperature 2 assembly [1, 2, 10, 22, 23, 27], the counter discussed in this paper uses purely a temperature 1 model to construct the square.

previous n x n square	Tiles	Bins	Stages
staged assembly [28]	O(1)	O(1)	O(logn)
Jigsaw technique [28]	O(1)	O(1)	O(logn)
my n x n square			
without counter (ch2)	O(B)	B	$O(\log_B n)$
with counter (ch3)	O(logn)	1	O(1)

Figure 1: Results of Temperature 1 Models

Our assembly using the counter allows for the use of constant stages and moves the complexities to the tiles instead.

Organization of this Document

The second chapter discussed will be the assembly which does not use a binary counter. Knowing how the bins operate is the first step. Each bin will only accept certain tiles into them. Knowing why there are specific tiles which go in specific bins is an important step in understanding how the construction works. Once proper understanding is gained about tiles and how they are used in bins, knowing how to make a 1 x *n* line will be the next step. As stated before, lines are a basic building block of a square, which is what the final step of our construction will be doing.

The third chapter of this paper is broken into stages, and focuses on construction of lines using a binary counter, which is more complex than a regular line construction. The first stage will describe how to build the basic building blocks of the “bit” tiles. Understanding how these tiles operate as a bit counter is an important part of the construction. The next stage involves knowing how to build bit strings, which will combine together to form lines. These strings, referred to in this paper as “towers”, will have a unique way of combining together in an incrementing manner to form a line, which the next section of the paper explains how to link together. It will involve building multiple lines simultaneously, as each line will be of a power of two. Being able to combine multiple length lines to form the desired length line is thoroughly discussed in the third stage of this chapter. The final stage explains how to form a square with lines having a width factor to take into account.

CHAPTER II

STAGED SELF-ASSEMBLY WITHOUT THE USE OF A BINARY COUNTER

For this construction, the number of bins determines the growth of the supertiles. Not all of the bins need to be used during each stage. Also, without the use of extension tiles, not all sizes of *n* can be achieved. The example constructions will demonstrate the use of these two points to achieve our desired length. The max number of extension tiles required does not exceed 3 for any construction of a line or square, and the inclusion of any extension tiles would modify some of the glues used on the final stage tiles labeled MB.

The Basic Construction

To be able to reference each of the bins and tiles used throughout the explanation, the bins will have labels starting from B1 to B4 for an assembly using only 4 bins. More bins can be used, but, for simplicity, we are limiting the explanation to the use of only 4 bins. A mixing bin, MB, will be used to combine the contents of B1-B4. Each of the primary bins will hold distinct single tiles which will be labeled B1-B4, depending where that tile should be placed. The MB bin also holds unique tiles reserved to its bin, but this only occurs once you are in the final stage of the construction. The letters or numbers associated with a specific side of a tile represents the glue used on that side. An A on the left side of one tile will bond with an A on the right side of another tile. The same rule is used with numbered glues. There are also special tiles, labeled S, which are the most used tiles in the entire assembly. S tiles will only be used during stage 1, but will be placed in every B bin along with the specific tiles for each bin. Keep in mind there are extension tiles needed to construct every size line, and the extension tiles do not exceed three for any construction of a line or square.

Assembly of a 1 x *n* Line

All tiles required to make a 1 x *n* line are displayed in figure 2. To fully assemble the line, we must first resolve the amount of stages and bins required to satisfy its length *n*. As stated before, our example will include four primary bins, and one mixing bin. We will let *n* = 37, and we will describe each stage of this process.

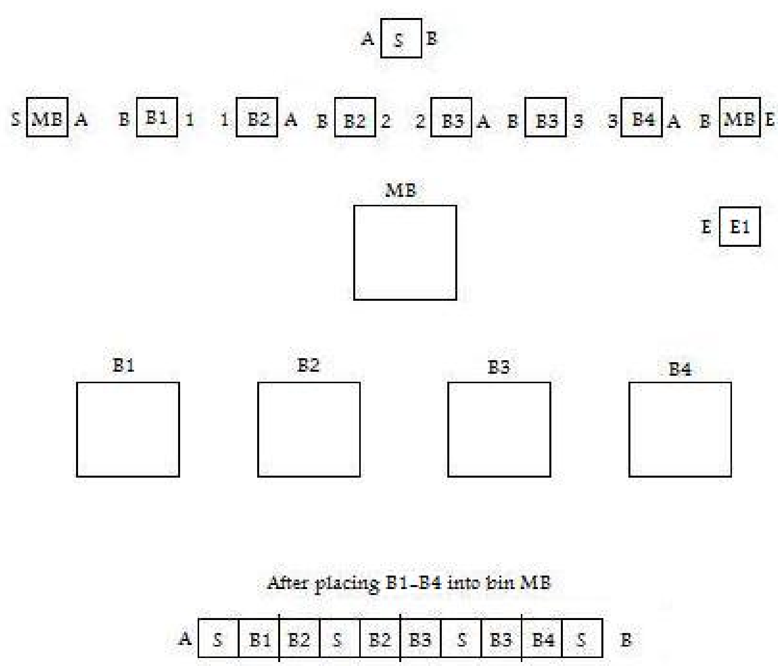

Figure 2: Line Assembly Setup

The first stage of the process is to place the S tiles into the four bins, along with the tiles labeled B1-B4 into their respective bin. This means there will be three different tile types applied to bins two and three, two different tile types in bins one and four. In effect, the supertiles created in each bin have left and right glues which will attach to the supertiles of the bins to their left and right, respectively. The first and last bin only have one tile connecting to an S tile to allow an open A glue on the left side and an open B glue on the right side. This is all done on purpose and will be explained in the second stage.

For the second stage of assembly, we move the fully assembled supertiles from the four primary bins into the MB. There, the four supertiles will combine to form a larger supertile of size 10, since two bins had supertiles of size three and two bins had supertiles of size two. This larger supertile has an A glue on its left and a B glue on its right, resembling a larger form of the S tile. At this point, we would normally distribute the fully constructed supertiles from MB evenly among bins B1-B4 and repeat the same process as stage one. If we were to use all four bins during this third stage, and combined them on the fourth stage, then there would be a supertile of size 48. This number is larger than the desired size we want, so we will take a different route for stage 3.

In order to reach the desired size, we will use only three bins instead of all four. Some adjustments need to be made during the construction when using less than the maximum bins available. Our basic construction attempts to leave an A glue on the left and a B glue on the right, so we will treat the third bin as though it was the fourth bin by placing only one of the two B3 tiles into the B3 bin, specifically the one which has the glue that connects to the B2 supertile. At this point, we have supertiles of size 11 in the first and last bins, and supertiles of size 12 in the middle bin.

During the final stage of the assembly, once we combine the fully assembled supertiles of bins B1-B3, the result is a larger supertile of size 34. We then add an MB tiles to each end, increasing the line's size to 36, and attach one extension tile to end of

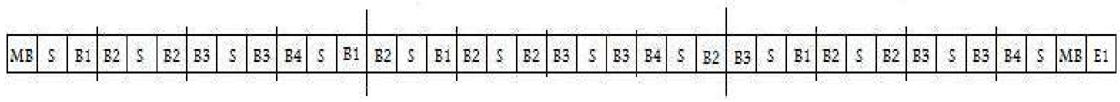

Figure 3: The Fully Assembled Line

the line to complete the intended 1 x *n* supertile.

Figure 3 shows the fully assembled line of length 37. The extended vertical lines which split the diagram in thirds represent the locations in which MB bin combined each bin together. A keen observation to note from this figure is the repeated use of the same type of tile. It will help to fully understand a completely assembled square which will be explained next.

Fully Assembling an *n* x *n* Square

How to make a line has already been explained in detail. All the steps used to construct a 1 x *n* line will also be used for a square, only with a few length modifications. There will be new tiles added to the assembly similar to the ones used in the construction of the 1 x *n* line, except we will be making both a vertical and a horizontal line simultaneously.

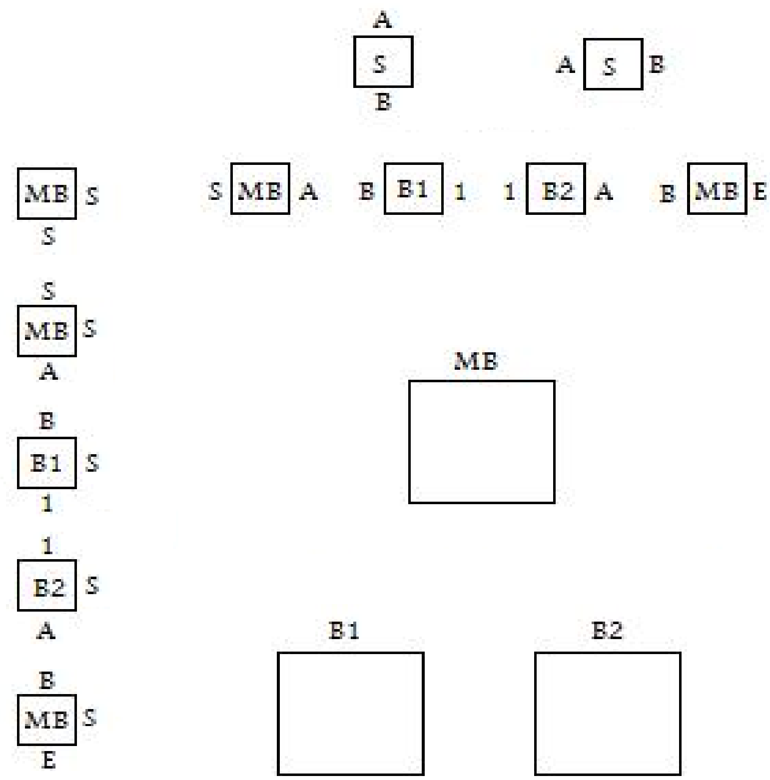

Figure 4: Square Assembly Setup

The figure is observed from top to bottom, left to right. As shown in Figure 4, the vertical tiles have glues identical to the horizontal tiles, with an extra glue on their right side labeled S. This serves as an anchor for the horizontal lines to attach to the vertical lines during the last stage of the assembly. The final construction of the square would be size n in height but size n+1 length. To solve this issue, we will construct lines of length n-1 instead, and add an extra vertical tile to make an n x n square. In each bin, there will be two types of supertiles being constructed; one being a vertical growth and the other being a horizontal growth. These two supertiles will not interact with each other until the appropriate glue which would allow them to bind is added.

To make the construction easier to observe, we will be making a square of size 13, meaning we need to construct lines of length 12. To build a square of this size efficiently, we require two bins during stage one and two during stage three, which is why figure 4 has fewer bins than figure 2. This construction requires four stages as well, and is similar to the 1 x *n* line. The same stage layout is used, but we will include more tiles into each bin.

For stage one and two, all tiles labeled B1 and B2 will go into their respective bins, along with both S type tiles being placed into both bins. Once the bins have fully constructed the supertiles, they will be moved to the MB bin. Supertiles of size 4 will be created both vertically and horizontally. Repeating the same steps for stage three and four produces lines of length 10. Since we are in the last stage, the MB tiles will also be added, producing lines of length 12, giving us our desired square size of 13. Figure 5 demonstrates an assembled size 13 square. Building a square does not require additional stages.

MB	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
MB	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
S	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
B1	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
B2	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
S	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
B1	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
B2	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
S	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
B1	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
B2	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
S	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB
MB	MB	S	B1	B2	S	B1	B2	S	B1	B2	S	MB

Figure 5: The Fully Assembled Square

CHAPTER III

STAGED SELF-ASSEMBLY THROUGH THE USE OF A BINARY COUNTER

With the help of a binary counter, we are able to use a constant number of bins and stages while still achieving the *logn* tile complexity and using only a temperature one construction, yet are limited to building lines which are a power of 2 in length. In order to achieve a length *n* line, we must first form lines of different powers of 2, then combine them. Also, the lines will have a height represented by the number of bits in the binary sequence. These towers connect to each other using geometry, which will be explained in detail during stage 3 of the assembly.

Stage 1 – Forming the Basic Tiles

We begin by constructing binary string supertiles, referred to from now on as "towers". First, a group of tiles must form together to create a binary digit. Next, several digits must combine to form a bit sequence. Each bit sequence will only adhere to its subsequent sequence, which goes next in the counter. All towers are created using the tiles shown in figure 4.

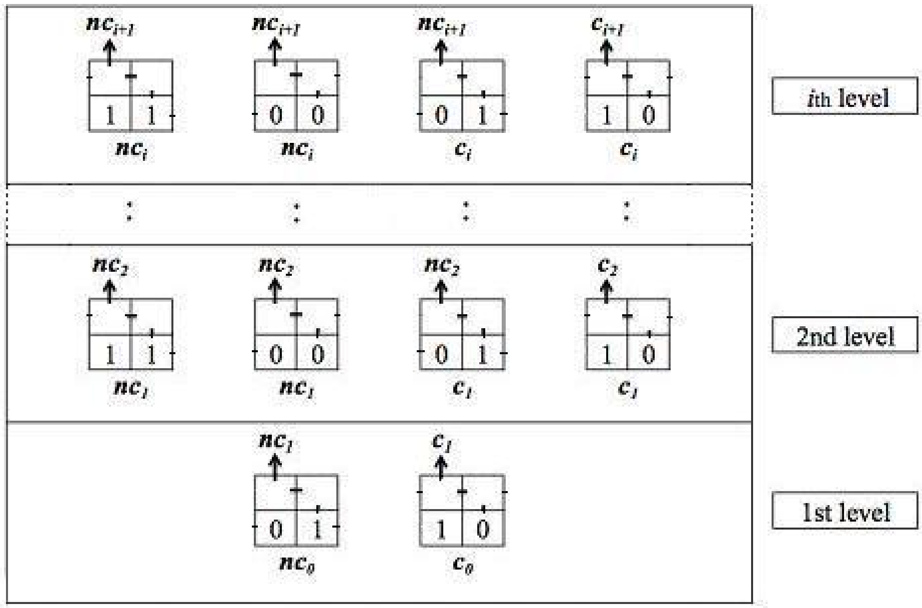

Figure 6: The Bit Tiles

In the first level, the two tiles represent the least significant digit. Squares at higher levels are digits of greater significance. The number on the left side of any given square is the bit we are currently on, while the number on the right side is the bit which is next. For level 1, the binary number in the least significant bit will always alternate between zero and one, which is a zero is always followed by a one, and vice versa. When a bit goes from zero to one, the binary counter increases by one. This results in a glue being added to the top of the square, which will only attach to squares which have no "carry glues". When a bit goes from one to zero, a change in the counter takes place at a higher level, so the top glue of this square will be a "carry glue" and will only attach to squares which accept this type of glue. For the second level, squares which had no carry glues will not forward a carry glue to the next level. Squares which accept carry glues will either increase the counter by one and not pass a carry glue, or will not increase the counter and pass on the carry glue to the next level of our tower construction. Every square will also be accompanied by tiles on its left and right sides, observed on the squares where the small lines are visible. The tile placement depends on the binary digits of the square, and will act as "teeth" for that square. For the left side of the square, if the bit is a zero, the tooth will attach to that tile directly. If the bit is not a zero, it will attach to the tile directly above it. The right side of the square uses the same mechanics, with emphasis on the one digit. If the digit is a one, it will have a tooth attached to it. If it is not a one, it will attach to the tile directly above it. These teeth greatly assist in the construction of the line and will be more closely observed in the connectivity of towers.

Stage 2 – Construction of Supertiles

To better understand how towers interconnect, we will observe how geometry plays its role in increasing the counter. In figure 7, the first three towers are displayed along with proper tooth placement, which are shaded in gray. As stated before, the left side of the tower represents the binary number you are currently on, while the right side is the binary number which is next in the sequence. As observed, the right side of each tower has a tooth connected to each one digit, and the left side of each tower has a tooth connected to each zero digit. Also, zeros on the right side of each tower have a tooth connected to the digit above it, while digits on the left side of each tower have a tooth connected to the digit above it. Tooth placement in this manner provides a secure and foolproof connection method for every binary tower.

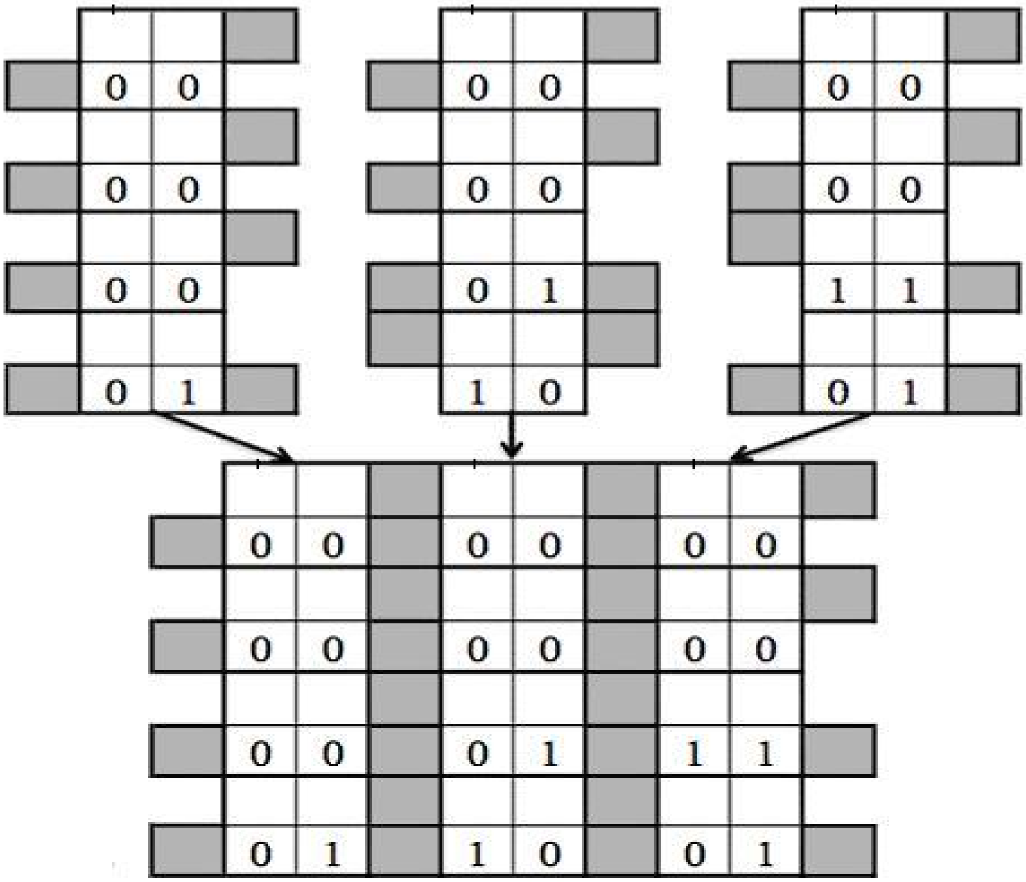

Figure 7: Geometry of Tiles

Once we reach the final binary sequence, we run into an issue. The next number in the sequence will be all zeroes, which would mean the right side of one supertile can connect to the left side of another supertile, resulting in an infinite length line. To resolve this situation, we will modify one of the four bit squares at the last level of our binary bits in order to prevent an infinite loop. If two teeth were to be added to the right side of our final sequence, it would create a geometrical obstruction, preventing the supertile from connecting to other supertiles of the same type. Since we are using a binary counter, there will be different length supertiles in the mix which must connect to each other uniquely to produce our final length *n* line. In order for them to connect, we must first give them the same height. For smaller supertiles, we must extend the height of the towers without increasing their length.

In order to achieve the same height on all supertiles, we must add additional tiles.

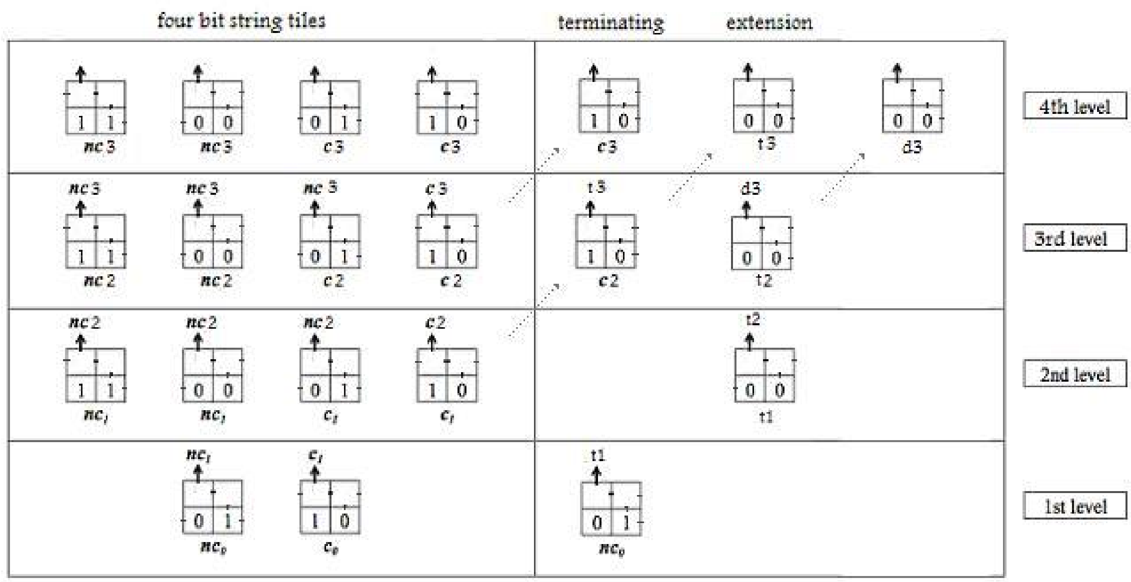

Figure 8: Creation of Varying Length Lines

Figure 8 shows the tiles needed to build a line which requires 3 different size supertiles. The largest supertile has a bit string of four bits, followed with a tower with three bits, and ending with a single bit. At each one of these levels, we allow our constructions to grow regularly, receive a termination tile, or receive extension tiles. If a termination tile attaches next in the construction, it will indicate the tower only has the amount of bits in the bit string as the level the termination tile was encountered. For example, if the termination tile was selected at random during the third level, then the middle sized supertile in our construction has completed. After a termination tile is selected, if it is not at the highest level of our construction, level 4 in our case, then extension tiles will be placed above to ensure each supertile has the same height. The bottom section of figure 9 demonstrates a fully assembled supertile based on the description above along with the top tiles on the top of the towers responsible for connecting the towers together.

Stage 3 – Supertile Connectivity

Combining each pre-constructed supertile will once again involve the use of geometry. At this point, every tower is floating around freely, with none connected to each other due to the absence of glues to connect them. Figure 9 shows the tiles which will stack above our towers, and the means of which all towers will be connected

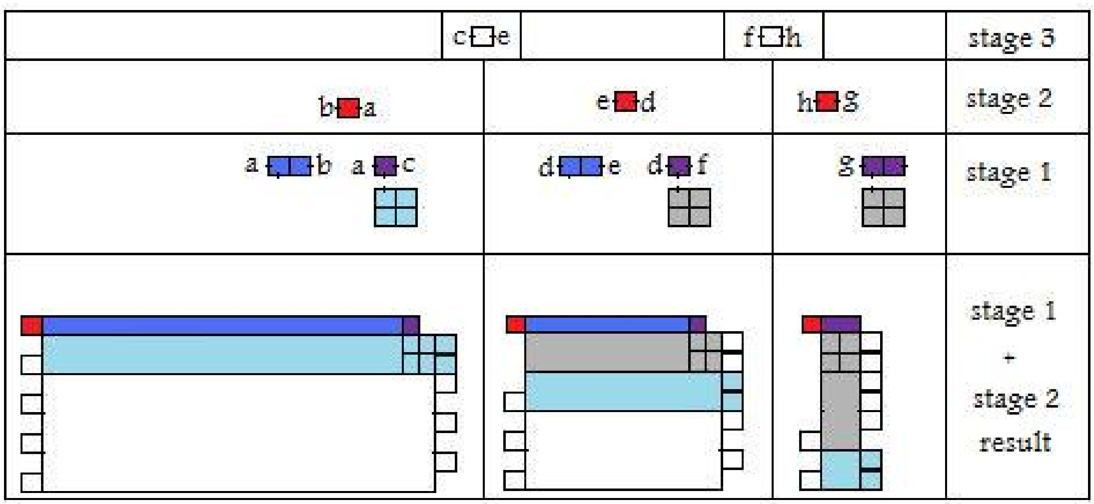

Figure 9: Connection of the Supertiles Stages 1-3

together. At the bottom of Figure 9, the supertiles are shown as they should appear after stages 1 and 2 of the assembly. The next section, labeled as stage 1, shows squares which represent the top right square of each supertile. Above each square is the unique, single tile glue piece which will attach to it. All the other top tiles which are not the top right square will receive the dual-tiled structure. The last supertile will be different, as it will no longer have a supertile to attach to on its right side. Instead of having a single tile attachment on its top, two tiles will be placed, marking the end of the supertile construction. At the end of stage 2, the towers will connect to each other forming supertiles. At the end of stage 3, all supertiles should be connected together forming our desired length *n* line. Note that teeth are still attached to the left and right side of the line. Solving this requires only 2*logn* more tiles, which still keeps the tile complexity at O(*logn*) and will be explained soon.

Stage 4 – The Final Construction

The final step in our construction is finishing our square. To do this, a way to connect vertical and horizontal line supertiles must be created. Our horizontal line is of size *k* x *n*, where *k* represents the height of our largest bit string. The vertical line must be

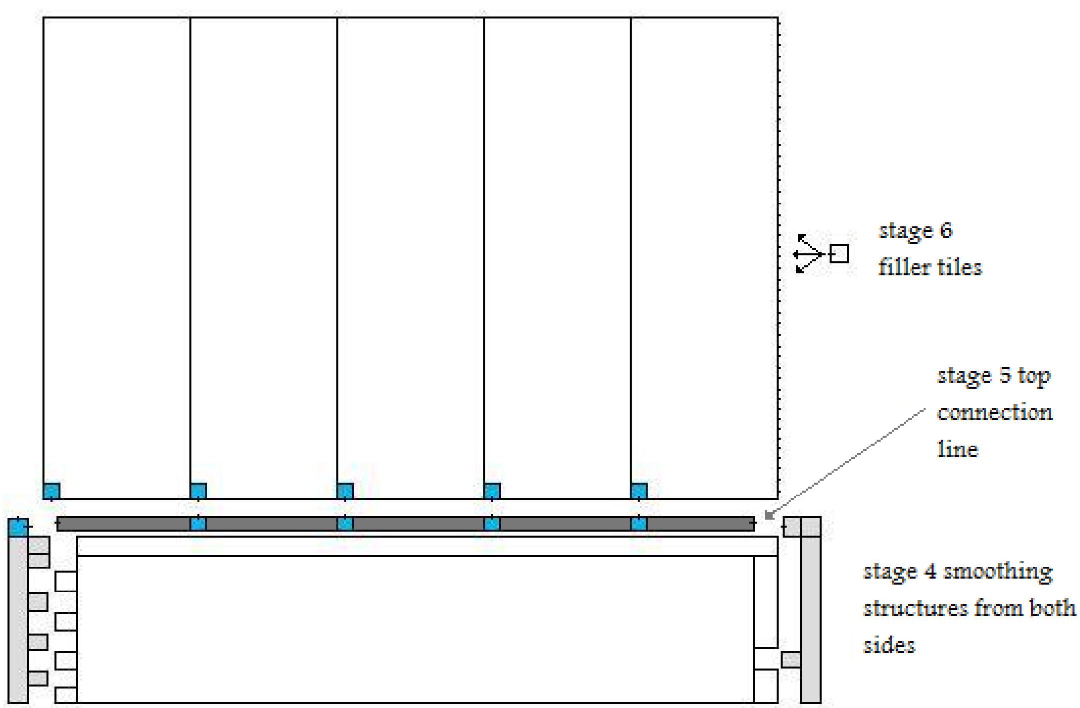

Figure 10: Creating the Full Square

of length *n*-*k*, and will be constructed similar method used for horizontal lines. For stage 4 of the assembly, we must even out the teeth on both sides of the lines, and the next stage will create a line adding glues to the attachment points between our vertical line and horizontal lines. Figure 10 demonstrates the smoothing procedure, along with a way to add top-side glues for other vertical lines to attach. Although smoothing will be done to both the horizontal and vertical lines, only the horizontal line will have the extra step of adding connection glues, which will be placed directly underneath connecting vertical lines. Placement in this manner results in having vertical lines slightly smaller in length to compensate. When placed side by side, the vertical lines may not be of sufficient length to create a full square. In those cases, filler tiles will be added to the right side during stage 6 to complete a perfect square. The tiles required to achieve this will not exceed *logn* tiles since they are there to fill in the space where a line would not be possible. Shown in figure 10, only one filler tile type was needed to fill out the entire right side of the figure.

CHAPTER IV

CONCLUSION AND FUTURE WORK

Our constructions have shown the strength of temperature 1 assemblies to create squares efficiently, and present an alternative to current methods. Both methods presented in this paper are efficient for relatively large squares, especially the method involving the binary counter. Building a small square using our methods would require more tiles than the $2n$-1 tile types presented in [28] by using only one stage and one bin.

A great future project which has sparked interest is the possibility of combining both assemblies explained in this paper to allow enough bins to create a bit sequence of size k. Each bin would contain the bit squares to be created. Another bin, labeled MB in our first construction, would allow the bit squares to merge into lines. These lines would then be taken back into their original bins, where they would be modified to a further degree, allowing them to act in the same manner as the bit squares from stage 1, and again combine in MB to increase the size of our construction once again by 2^k. It would be something very interesting to observe.

REFERENCES

[1] Leonard Adleman, Qi Cheng, Ashish Goel, and Ming-Deh Huang, *Running time and program size for self-assembled squares*, STOC '01: Proceedings of the thirty-third annual ACM Symposium on Theory of Computing (New York, NY, USA), ACM, 2001, pp. 740-748.

[2] Gagan Aggarwal, Michael H. Goldwasser, Ming-Yang Kao, and Robert T. Schweller, *Complexities for generalized models of self-assembly*, Proceedings of ACM-SIAM Symposium on Discrete Algorithms, 2004.

[3] Robert D. Barish, Rebecca Schulman, Paul W. Rothemund, and Erik Winfree, *An information-bearing seed for nucleating algorithmic self-assembly*, Proceedings of the National Academy of Sciences 106 (2009), no. 15, 6054-6059.

[4] Florent Becker, Ivan Rapaport, and Eric Remila, *Self-Assembling classes of shapes with a minimum number of tiles, and in optimal time*, Foundations of Software Technology and Theoretical Computer Science (FSTTCS), 2006, pp. 45-56.

[5] Ho-Lin Chen, Qi Cheng, Ashish Goel, Ming-Deh Huang, and Pablo Moisset de Espanes, *Invadable self-assembly: Combining robustness with efficiency*, SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms), 2004.

[6] Ho-Lin Chen and Ashish Goel, *Error free self-assembly with error prone tiles*, Proceedings of the 10^{th} International Meeting on DNA Based Computers, 2004.

[7] Ho-Lin Chen, Ashish Goel, and Chris Luhrs, *Dimension augmentation and combinatorial criteria for efficient error-resistant DNA self-assembly*, Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008 (Shang-Hua Teng, ed.), SIAM, 2008, pp. 409-418.

[8] Ho-Lin Chen, Ashish Goel, and Chris Luhrs, *Dimension augmentation and combinatorial criteria for efficient error-resistant dna self-assembly*, SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms (Philadelphia, PA, USA), Society for Industrial and Applied Mathematics, 2008, pp. 409-418.

[9] Ho-Lin Chen, Rebecca Schulman, Ashish Goel, and Erik Winfree, *Reducing facet nucleation during algorithmic self-assembly*, Nano Letters 7 (2007), no. 9, 2913-2919.

[10] Qi Cheng, and Pablo Moisset Espanes. *Resolving two open problems in the self-assembly of squares*. Technical Report 03-793, University of Southern California, June 2003.

[11] Erik D. Demaine, Martin L. Demaine, Sandor P. Fekete, Mashhood Ishaque, Eynat Rafalin, Robert T. Schweller, and Diane L. Souvaine, *Staged self-assembly: nanomanufacture of arbitrary shapes with O(1) glues*, Natural Computing 7 (2008), no. 3, 347-370.

[12] David Doty, *Randomized self-assembly for exact shapes*, Proceedings of the Fiftieth IEEE Conference on Foundations of Computer Science (FOCS), 2009.

[13] Kenichi Fujibayashi, David Yu Zhang, Erik Winfree, and Satoshi Murata, *Error suppression mechanisms for dna tile self-assembly and their simulation*, Natural Computing: an international journal 8 (2009), no. 3, 589-612.

[14] Ming-Yang Kao and Robert T. Schweller, *Reducing tile complexity for self-assembly through temperature programming*, Proceedings of the 17^{th} Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2006), Miami, Florida, Jan. 2006, pp. 571-580, 2007.

[15] Ming-Yang Kao and Robert T. Schweller, *Randomized self-assembly for approximate shapes*, International Colloquium on Automata, Languages, and Programming (ICALP) (Luca Aceto, Ivan Damgrd, Leslie Ann Goldberg, Magns M. Halldrsson, Anna Inglfsdttir, and Igor Walukiewicz, eds.), Lecture Notes in Computer Science, vol. 5125, Springer, 2008, pp. 370-384.

[16] James I. Lathrop, Jack H. Lutz, and Scott M. Summers, *Strict self-assembly of discrete Sierpinski triangles*, Proceedings of The Third Conference on Computability in Europe (Siena, Italy, June 18-23, 2007), 2007.

[17] Furong Liu, Ruojie Sha, and Nadrian C. Seeman, *Modifying the surface features of two-dimensional DNA crystals*., Journal of the American Chemical Society 121 (1999), no. 5, 917-922.

[18] Urmi Majumder, Thomas H LaBean, and John H Reif, *Activatable tiles for compact error-resilient directional assembly*, 13^{th} International Meeting on DNA Computing (DNA 13), Memphis, Tennessee, June 4-8, 2007., 2007.

[19] Chengde Mao, Thomas H. LaBean, John H. Relf, and Nadrian C. Seeman, *Logical computation using algorithmic self-assembly of DNA triple-crossover molecules*., Nature 407 (2000), no. 6803, 493-6.

[20] Chengde Mao, Weiqiong Sun, , and Nadrian C. Seeman, *Designed two-dimensional DNA holliday junction arrays visualized by atomic force microscopy*., Journal of the American Chemical Society 121 (1999), no. 23, 5437-5443.

[21] John Reif, Sudheer Sahu, and Peng Yin, *Compact error-resilient computational DNA tiling assemblies*, DNA: International Workshop on DNA-Based Computers, LNCS, 2004.

[22] Zachary Abel, Nadia Benbernou, Mirela Damian, Erik Demaine, Martin Demaine, Robin Flatland, Scott Kominers, Robert Schweller, *Shape Replication Through Self-Assembly and RNase Enzymes*, Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), Austin, Texas Jan, 2010.

[23] Paul W. K. Rothemund and Erik Windfree, *The program-size complexity of self-assembled squares (extended abstract)*, STOC '00: Proceedings of the thirty-second annual ACM Symposium on Theory of Computation (New York, NY, USA), ACM, 2000, pp. 459-468.

[24] Paul W. K. Rothemund, Nick Papadakis, and Erik Winfree, *Algorithmic self-assembly of DNA Sierpinski triangles*, PLoS Biology 2 (2004), no. 12, 2041-2053.

[25] Rebecca Schulman and Erik Winfree, *Programmable control of nucleation for algorithmic self-assembly*, DNA: International Workshop on DNA-Based Computers, LNCS, 2004.

[26] Rebecca Schulman and Erik Winfree, *Synthesis of crystals with a programmable kinetic barrier to nucleation*, Proceedings of the National Academy of Sciences 104 (2007), no. 39, 15236-15241.

[27] David Doty, Matthew J. Patitz, Dustin Reishus, Robert T. Schweller, and Scott M. *Summers, Strong Fault-Tolerance for Self-Assembly with Fuzzy Temperature*. TBA Conference Paper.

[28] Paul W. K. Rothemund, *Theory and experiments in algorithmic self-assembly*, Ph.D. thesis, University of South California, December 2001.

[29] Hao Wang, *Proving theorems by pattern recognition* – II, The Bell System Technical Journal XL (1961), no. 1, 1-41.

[30] Hao Wang, *Dominoes and the AEA case of the decision problem*, Proceedings of the Symposium on Mathematical Theory of Automata (New York, 1962), Polytechnic Press of Polytechnic Inst. Of Brooklyn, Brooklyn, N.Y., 1963, pp. 23-55.

[31] Erik Winfree and Renat Bekbolatov, *Proofreading tile sets: Error correction for algorithmic self-assembly.*, DNA (Junghuei Chen and John H. Reif, eds.), Lecture Notes in Computer Science, vol. 2943, Springer, 2003, pp. 126-144.

[32] Erik Winfree, *Simulations of computing by self-assembly*, Tech. Report Caltech CSTR:1998.22, California Institute of Technology, 1998.

[33] Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman, *Design and self-assembly of two-dimensional DNA crystals*., Nature 394 (1998), no. 6693, 539-44.

[34] Erik Winfree. Algorithmic Self-Assembly of DNA. PhD thesis, California Institute of Technology, Pasadena, 1998.

BIOGRAPHICAL SKETCH

Nicolas H. Gutierrez earned a Master of Science in Computer Science from the University of Texas – Pan American in May 2010. He earned a Bachelor of Science in Computer Science with a minor in Electrical Engineering from the University of Texas – Pan American in May 2006. He earned his Associates Degree in Computer Science from South Texas Community College May 2004. He has two years teaching Mathematics at the Rio Grande City High School. His permanent mailing address is 1004 River Bend Street, Rio Grande City, Texas 78582.