

Checkpointing of Rectilinear Growth in DNA Self-assembly

S. Frechette+, Y. B. Kim++ and F. Lombardi++
+Riverside Research Institute
Lexington, MA USA
++Northeastern University,
ECE Department
Boston, MA 02115

Abstract

Error detection/correction techniques have been advocated for algorithmic self-assembly. Under rectilinear growth, it requires only two additional tiles, generally referred to as Isolation tiles. This process can be effectively utilized for checkpointing and is analyzed in this paper self-assembly. Initially, the physical framework (and related features) for the removal of the erroneous sections of an assembly is outlined. A novel Markov based model is presented to establish the optimal rate of checkpointing and assess its performance versus other error tolerant techniques that utilize redundancy. Simulation results are provided.

Index Terms: checkpointing, error tolerance, DNA self-assembly, tiling.

1 Introduction

Self-assembly has been widely advocated as an efficient manufacturing environment for nanotechnologies; self-assembly can be used to avoid expensive nano-lithography in building structures (such as scaffolds), to which other molecular-based electronic devices can be attached. This process uses complexes commonly referred to as *tiles*; for example, in rectangular structures a tile has four types of DNA strands. For self-assembling into specified patterns, a large number of tiles (made of DNA strands) are programmed through their bond types. Previous works have reported that the assembly of incorrect tiles occurs with error rates between 1 to 10 percent [2]; as millions of molecules are usually involved, the presence of these errors represents a serious challenge for efficient manufacturing in the nano ranges. Therefore *detection and correction techniques* are needed.

Several techniques have been proposed. Error correction techniques that require additional matching ends, have been proposed in the literature; in [15], pads between tiles decrease the error rate by requiring more bonds to match for each assembled tile. The use of an additional layer for error rate reduction requires two layers of tiles to match perfectly (as opposed to a single layer) [16]. Two proofreading methods for error correction include a tile set proposed in [2] (in which a so-called block consists of four proofreading tiles) and the snake tile set [10] (that requires a block of sixteen proofreading tiles). The use of checkpoints for DNA-self assembly has been investigated in previous works; [14] has proposed, simulated, and evaluated a checkpoint scheme based on *temperature pulsing* during crystalline array growth. Periodic pulses remove defective, and adversely affect some of the correct (error-free) sections of the crystalline arrays. In this process, it is assumed that a temperature pulse (i.e., a temporary increase in temperature) removes all defective regions of the structure. Although a portion of the removed tiles are correct, incorrect tiles are removed at a higher rate. Additionally in [14], the number of temperature pulses required for an error-free assembly, and the optimal temperature pulse checkpointing interval, have been established. This paper presents a detailed analysis to extend the process of [20] to tile checkpointing for rectilinear growth in DNA self-assembly. This analysis utilizes the Error Isolation Tiles of [20] while introducing a physical

framework by which erroneous sections of the aggregate can be removed. Checkpointing is then analyzed under a novel Markov model to establish the optimal rate. This technique is compared to existing redundancy based techniques. It is shown to exhibit superior performance by solving the proposed Markov model and simulation; a substantial reduction of error rate is also achieved due to the periodic execution of the checkpointing process.

2 Review of Error-Tolerance for DNA

In algorithmic self-assembly, the growth of a DNA crystal is used for information processing. A set of DNA tiles is used to execute an algorithm. The *abstract Tile Assembly Model (aTAM)* [18] provides the basis for analysis of algorithmic self-assembly in an ideal case. A *tile set* consists of a finite set of unique *tiles* that is used to self-assemble into a DNA *crystal*. A tile is assumed to be square and tiles can not rotate by assumption. Each of the four sides of a tile has a *bond type*. The bond types of a tile determines the uniqueness of the tile. Each bond type has an associated *bond strength*. Bond types can be null (strength of 0), single (strength of 1), or double (strength of 2). Two bonds of the same type can glue (i.e. bond) together, with a corresponding strength. It is assumed that the strength between different bond types is always 0. In an ideal process, self-assembly always begins with a *seed tile*. A tile can be added to the existing crystal when its total bond strength to the crystal is greater than or equal to 2. The crystal generated from the seed tile via a series of legal (correct) tile additions is referred to as the *produced assembly*. The *integer* in the tile denotes the *bond type*. In practice, a *mismatch* may occur during this process, such that a tile with a total bound less than 2 is attached. Additionally, a tile can also fall off from a crystal.

The *kinetic Tile Assembly Model (kTAM)* [18] provides a framework to analyze and simulate the non-ideal self-assembly. The kTAM model includes rates for both *association and dis-association* of tiles from the crystal. In this model, it is assumed that the on-rate (association) r_{on} is determined only by the tile concentration, that is determined by the parameter G_{mc} . The off-rate r_{off} (dis-association) is determined by the total bound strength b that holds the tile to the crystal and the parameter G_{se} . These rates are given as follows: $r_{on} = k \times e^{-G_{mc}}$ and $r_{off,b} = k \times e^{-bG_{se}}$, where k is a constant, G_{mc} is the physical parameter measuring the tile concentration, while G_{se} is the physical parameter measuring the unit bond strength.

This paper specifically addresses checkpointing for detection and correction of *growth errors*, which are defined as weakly-bonded tile attachments at a location where another tile could and should attach. Growth errors occur if there exist at least two incorrect (erroneous) attachments upon the same four-sided tile. The error correction technique presented in [20] is commonly referred to as the *Error Isolation Tile method*. *Border tiles* are defined as any tile on an edge of an assembly, in which growth has ceased. In the model [20], border tiles assemble at the same rate as the entire assembly, thus a tile mismatch is allowed to *propagate to an edge*, and causes a *disjointed line* to occur in the border tiles, i.e., the initial error results in a disjointed line in the border tiles, that should otherwise be straight. At the break in the border line using the proposed method, a so-called *Error Isolation Tile* attaches and thus, it effectively *tags* that section of the assembly for removal. A growth error is illustrated in Figure 1; the disjoint line (as effect of this error) is evident.

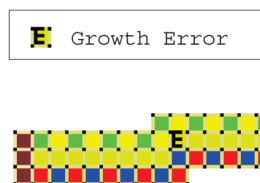


Figure 1. Initial Growth Error.

Although a set of Error Isolation Tiles that attach to every possible defective structure of two

adjacent tiles could be constructed, the number of tiles in such an Error Isolation Tile set would be prohibitively large, thus very inefficient in terms of overhead in the final assembly. In addition to a tile mismatch, it is possible that a growth error is caused by weak bonds. The bond strength must not allow for shifting a subsequent column up or down by one row; if this occurs, then it is considered a growth error. The approach of [20] requires only two additional tiles, which are referred to as the Error Isolation Tile set. The rectilinear growth of the assembly results in the need for two tiles in the tile set, as opposed to just one tile. (1) The first Error Isolation Tile attaches to the two disjoint border tiles from the southeast. (2) The second Error Isolation Tile attaches from the southwest. Errors are detected when an Error Isolation Tile attaches to the assembly at the location of two disjoint and adjacent border tiles, which can only be adjacent in error. Any growth error results in a disjoint line at the southern border tiles, so only two Error Isolation Tiles are required to detect a large number of possible growth errors [20]. Border tiles are employed in most tile sets for algorithmic self-assembly [9]. In most tile sets these tiles assemble prior to the interior tiles. Thus, propagation of an error within the interior tiles will be blocked by the border tiles and prevented from reaching an edge of the assembly. In the proposed model, border tiles in the tile set may only assemble if an interior tile next to the border is assembled. The growth error in the interior will cause a break in the straight line of the border tiles. The border tiles make up the disjoint bottom row. The set employed to demonstrate the Error Isolation Tile correction technique of [20] is a modified version of the linear tile set presented in [1]. The method proposed in [20] can be applied to a modified version of any tile set that exhibits algorithmic self-assembly, such as the Sierpinski triangle or the binary counter tile sets of [9].

3 Physical Framework

In the method of [20], all Error Isolation Tiles hold a metal, i.e. *a conductive cargo*. Thus, the assemblies with an attached Error Isolation Tile can be attracted to a charged metal and a section of the assembly (within a given radius of the Error Isolation Tile) can be raised in temperature, thus causing bonds to break and a partition of the assembly to occur. This partition divides the assembly into error-free and erroneous (defective) sections. This is accomplished as follows. (1) The device that removes the defective section of the assembly is referred to as the *Partitioner Cell*, and is shown in Figure 2a. An array of partitioner cells is illustrated in Figure 2b. The Error Isolation Tiles are attached to the charged metal strip, known as the Error Isolation Tile *Attractor*, this is shown as (a) in Figure 2a. The portion of the assembly to be removed contains the initial error and all tiles to the right of the error, so the assembly attaches to the Partitioner Cell with the Error Isolation Tile on the south side. (2) To ensure a proper orientation of the assembly, a 3-D *Orientation Bar* (shown as (b) in Figure 2a) is employed to prevent the assembly from attaching to the Partitioner Cell at an incorrect orientation. (3) Lastly, the *Partitioner Strip* (shown as (c) in Figure 2a) is a 2-D strip of metal that heats the local portion of the assembly directly above it, thus causing the bonds (also directly above it) to break. Figure 3a illustrates an occupied Partitioner Cell. After an assembly attaches to a Partitioner Cell and the portion of the assembly (that is located above the Partitioner Strip) breaks, the assembly is partitioned into error-free and erroneous sections. This is illustrated in Figure 3b. The erroneous section of the assembly remains connected to the charged strip, and the error-free sections of the assembly return to the solution. The array of Partitioner Cells continues to fill with defective portions of the assembly, so there must be sufficient arrays of Partitioner Cells for all assemblies with Error Isolation Tiles attached. Arrays of Partitioner Cells are always immersed in the solution; they attract only erroneous assemblies when the self-assembly is in a checkpointing state. The checkpoint process is therefore complete, and it may repeat at the next checkpoint for a set interval.

The section of the assembly located above the Partitioner Strip ((c) in Figure 2a), is removed, as well as the subsequent tile growth that could have been affected by the initial error. The Orientation Bar (given by (b) in Figure 2a) ensures that the erroneous assembly attaches at the proper orientation with respect to the Partitioner Cell. A proper orientation is required, such that only tiles near

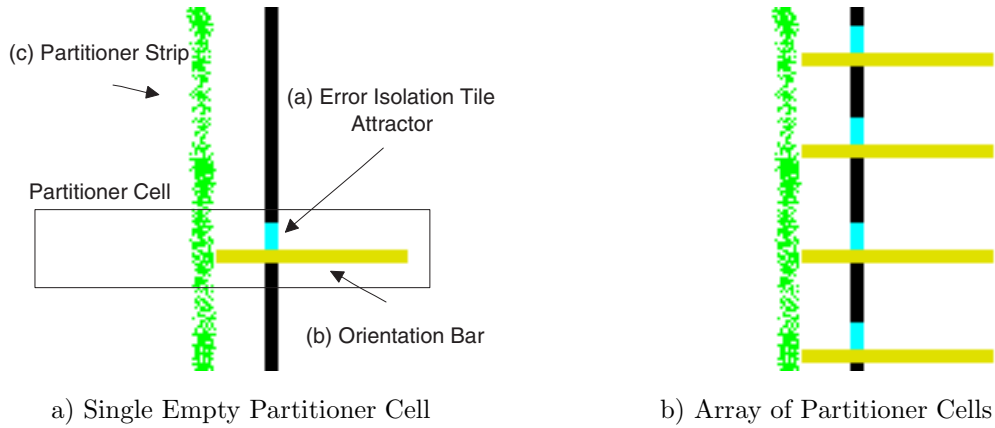


Figure 2. Partition Cells.

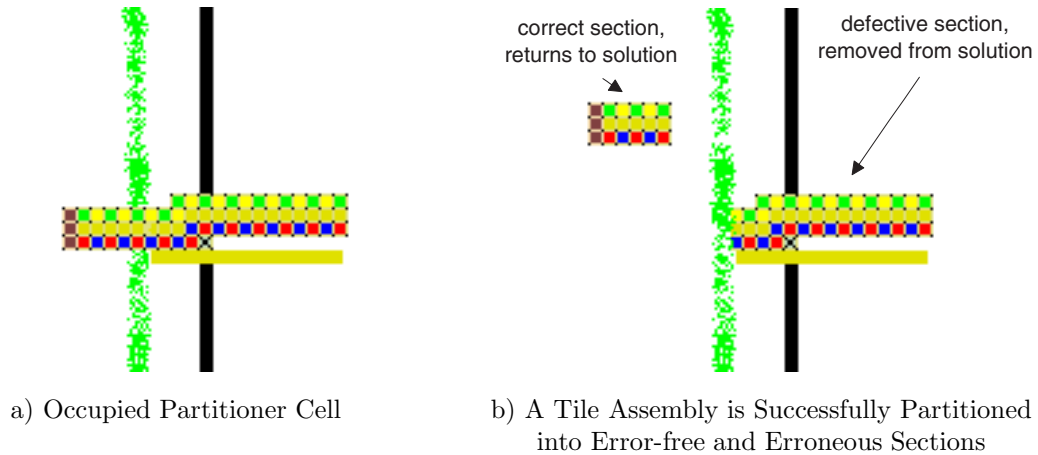


Figure 3. Example of an Occupied Partitioner Cell Before and After Partitioning an Erroneous Tile Assembly.

and to the east of the error are removed. The checkpointing process occurs over a period of time; in the current state of self-assembly, ten seconds is an estimate of the length of the time required for this process. Since the items are in a solution, it is estimated that ten seconds are required for all assemblies in a solution to pass close enough to the charged metal strip and to become attached.

4 Optimal Checkpointing

The optimal checkpoint interval is determined in terms of the error rate per tile (denoted by λ). λ is given by $2 \times N \times e^{-G_{se}}$ in [2], where N is the number of tiles in the set ($N = 10$ for the $3 \times Y$ rectangular tile set of [20]). The rate at which an erroneous assembly attaches to a Partitioner Cell and returns to the solution, is given on a per assembled tile basis, and is represented by μ . The optimal checkpoint interval for the removal of the erroneous (defective) sections of the assembly can be determined. Given an error rate $\lambda = 2 \times N \times e^{-G_{se}}$, and the rate at which an erroneous tile attaches and then partitions using a Partitioner Cell (given by μ) the optimal checkpoint rate for error detection and correction is determined in this subsection.

A Markov model is employed to represent the states that an assembly is classified. A discrete-time Markov model is used; in this model, transitions occur at fixed time intervals of Δt . The

Markov diagram consists of the three possible states of an assembly and is shown in Figure 4. The value of a directed edge in the Markov diagram is the probability of moving to the respective state during the time step Δt , i.e. Δt represents the time required for a tile to attach to the assembly.

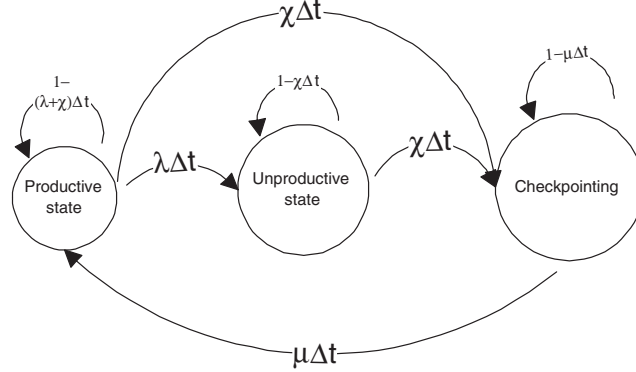


Figure 4. Markov Diagram for the Error Isolation Tile Correction Method.

The Markov diagram of Figure 4 has the three possible states of an assembly. (1) When an assembly is in the *Productive state*, tiles can attach to the assembly, thus an Error Isolation Tile cannot be attached to the assembly. (2) If the assembly is in the *Unproductive state*, then a growth error has occurred. In the model [20], it is estimated that an Error Isolation Tile will always attach to an erroneous assembly prior to the occurrence of checkpointing. Since an Error Isolation Tile will eventually attach near the growth error, at the next checkpoint all subsequently attached tiles will be removed along with the entire erroneous section of the assembly. (3) When a checkpoint occurs, all assemblies are moved into the *Checkpointing state*; in this state, the mechanism that partitions the assembly for containment of errors, is active. Due to the disruption incurred by the array of Partitioner Cells, it is estimated that no tile attaches to any assemblies while in the checkpointing state.

As it is estimated that all state changes in the Markov diagram exhibit an exponential distribution, then let $P_{Sn}(t)$ represent the probability of an assembly being in a given state n in the Markov diagram of Figure 4. On the assumption of an error-free initial condition and non-negative continuous random variables, in the Laplace transform domain, the first-order differential equations are solved via algebraic manipulation. Using the Final-value theorem, the steady state availability is approximated as follows:

$$\begin{aligned}
 A_{checkpoint}(steady - state) &= \lim_{s \rightarrow 0} s(P_{S0}(s)) \\
 &= \lim_{s \rightarrow 0} s \frac{\frac{1}{s}((\chi + s)(\mu + s))}{\lambda\mu + \mu\chi + \mu s + \lambda\chi + \lambda s + \chi^2 + 2\chi s + s^2} \\
 &= \frac{\chi\mu}{\lambda\mu + \mu\chi + \lambda\chi + \chi^2} \\
 &= \frac{\mu}{\frac{\lambda\mu}{\chi} + \mu + \lambda + \chi}
 \end{aligned} \tag{1}$$

Thus, the optimal value of χ , which results in a maximum of $A_{checkpoint}(steady - state)$, is given below in terms of λ and μ

$$\text{Optimal checkpoint rate } (\chi) = \sqrt{\lambda\mu} \tag{2}$$

As an example of the convergence of χ towards a maximum $A_{checkpoint}(steady - state)$, a plot of χ versus $A_{checkpoint}(steady - state)$ for $\lambda = \mu = 0.1$ is shown in Figure 5. As the tile set employed

exhibits rectilinear growth (from west to east), then once an error occurs all subsequent tiles that assemble will be removed at the next checkpoint, thus guaranteeing the assembly's correctness.

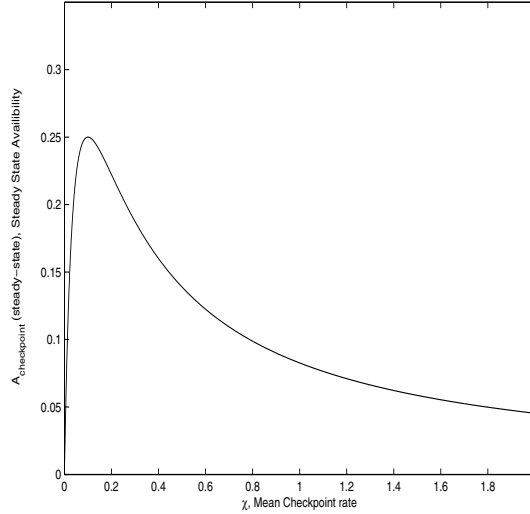


Figure 5. χ versus $A_{checkpoint}(steady - state)$ for $\lambda = \mu = 0.1$

5 Simulation Results and Analysis

Two error correction techniques are compared with the Error Fault Isolation Tile method [20]. Results were obtained using the Xgrow simulator [1]; these three techniques are compared using the probability of error and size of the final assembly as criteria. In the simulation, the following two parameters represent the physical conditions in which the crystalline arrays assemble: Gse is defined as the *temperature* of the solution, and Gmc is the *tile concentration*. The time required for a checkpoint is fixed at ten (virtual)seconds. The optimal checkpointing rate for a given error rate and mean checkpoint length have been determined in (2). For $Gse=9.1$, the asymptotic error rate is given as $O(e^{-Gse})$ [2], and the mean checkpoint time is estimated as 10 (virtual) seconds. Moreover, it is also estimated that the entire checkpoint process takes 10 (virtual) seconds (as observed in the execution of the Xgrow simulation environment). Ten *virtual seconds* is the time spent at a checkpoint in which no tiles can assemble. In the simulations, the $3 \times Y$ rectangle assembles at a rate of approximately 0.2 assembled tile per virtual second for $Gmc = 2Gse$. Once the assembly is in the checkpointing state, the rate at which the erroneous tiles attach to the Partitioner Cell and successfully partition, is $\mu = \frac{1}{0.2 \times 10} = 0.5 \frac{\text{partitioned erroneous tile}}{\text{assembled tile}}$. When in the checkpointing state, the assembly ceases, and only partitioning occurs.

The optimal checkpoint rate, for $Gse = 9.1$, is given by $\sqrt{2 \times 10 \times e^{-Gse} \times 0.5} = 0.033$ checkpoints per assembled tile. Thus, the optimal checkpoint interval (or period between checkpoints) in this case is given by 161 virtual second/checkpoint. Therefore if the Error Isolation Tile method is employed with a checkpoint interval of $151 + 10 = 161$ virtual seconds, and an assembly is completed in 1,058 seconds, then $\lceil 1058/161 \rceil \times 10 = 70$ virtual seconds are spent in the checkpointing state. The optimal checkpoint rate in term of $\frac{\text{assembled tile}}{\text{checkpoint}}$ is simply $151 \frac{\text{virtual seconds}}{\text{checkpoint}} \times 0.2 \frac{\text{assembled tile}}{\text{virtual seconds}} = 30.2 \frac{\text{assembled tile}}{\text{checkpoint}}$, for $Gse = 9.1$.

This section presents the performance evaluation for three error correction techniques, namely 2×2 Proofreading [2], 4×4 snake proofreading [10], and the Error Isolation Tile method [20] for the assembly of a 3×20 rectangular structure. In the two error correction techniques used for comparison, each of the three tiles marked either southern border or interior, were subdivided into 2×2 proofreading or 4×4 snake proofreading tile sets.

[10] has proved that the 4×4 snake proofreading method results in no error for $Gmc \approx 2Gse$ (optimal growth). To compare the performance of the error correction techniques, simulation must assemble tiles at a cooler temperature, thus resulting in a greater error percentage, i.e. the cooler the temperature, the stronger the bonds and the greater the error rate due to the faster speed of assembly. Hence, the values of Gse and Gmc that were used in the presented simulations are not optimal in terms of a minimum error rate for $Gmc \approx 2Gse$ [2], so to better evaluate the error correction techniques and to increase the error rates of the original $3 \times Y$ tile set, the error correction techniques were compared in cooler and non-optimal temperature conditions.

Winfree has noted that for DNA-based self-assembly, Xgrow [18] accurately models the quantities of all error types observed in physical assemblies [5]. The *optimal checkpoint interval* for the conditions of the simulation can then be determined. The optimal checkpoint interval for $Gse = 8.5$ is 98 virtual (Xgrow time) seconds, and 156 virtuals seconds for $Gse = 9.5$. Checkpointing intervals within this range were used for all simulations.

A *trial* is categorized as an assembly error if the assembly is not error-free, i.e. all columns match previous columns and the border tiles are not disjoint. The *assembly error probability* is defined as the measured proportional probability that an assembly is not error-free. In these experiments, the assembly of a 3×20 rectangular is considered. Figure 6 plots the assembly error probability versus Gse , (Gmc is held constant at 15). The *tile mismatch probability* is the number of mismatched tiles divided by the number of tiles used, i.e. 60, 240, and 960 for the original tile, 2×2 proofreading, and the 4×4 snake proofreading tile sets respectively. Figure 7 plots the tile mismatch probability versus Gse . The large assembly error probability and the small tile mismatch probability of the original $3 \times Y$ rectangular tile set are caused by an incorrect formation due to tiles with weak bonds, such that other tiles attach and strengthen before they fall off.

Figures 6 and 7 show a comparison of the original 3×20 tile set, 2×2 proofreading tile set, 4×4 snake tile set and the Error Isolation Tile Method (averaged over 11 trials. The Error Isolation Tile correction method [20] exhibits less errors than the 2×2 proofreading and 4×4 snake proofreading methods. The technique of [20] requires significantly less tiles than the other error correction techniques listed in Figure 6 (that require an increase in the number of tiles of the final assembly by a factor of four or sixteen fold due to the massive redundancy required), i.e. if an assembly requires either four or sixteen times as many tiles, then the final assembly is four or sixteen times as large.

6 Conclusion

A novel method for error correction in nano-scale DNA based self-assembly of crystalline arrays was presented in [20], it relies on so-called Error Isolation Tiles. In this paper, a detailed treatment of error Isolation Tiles has been pursued. Initially, a physical framework has been proposed for implementing tile removal using the technique of [20]. Through the use of the border tiles in conjunction with Isolation Tiles, it has been shown that this physical framework with checkpointing can be used to remove defective (erroneous) sections. A novel Markov model has also been presented to assess the optimal checkpointing rate of self-assembly; it has been shown that [20] with checkpointing results in a smaller final assembly than all previous methods known to the authors because no redundancy in the tiles is utilized.

References

- [1] E. Winfree et al.. The xgrow simulator. <http://www.dna.caltech.edu/Xgrow/>
- [2] E. Winfree and R. Bekbolatov. Proofreading tile sets: Error correction for algorithmic self-assembly. *Proc. 9th Int. Meeting on DNA Based Computers*, Madison, Wisconsin, June 2003.
- [3] C. Mao, T. H. LaBean, J. H. Reif, and N. C. Seeman. "Logical computation using algorithmic self-assembly of DNA triple-crossover molecules," *Nature*, vol. 407(6803), pp. 493-496, 2000.

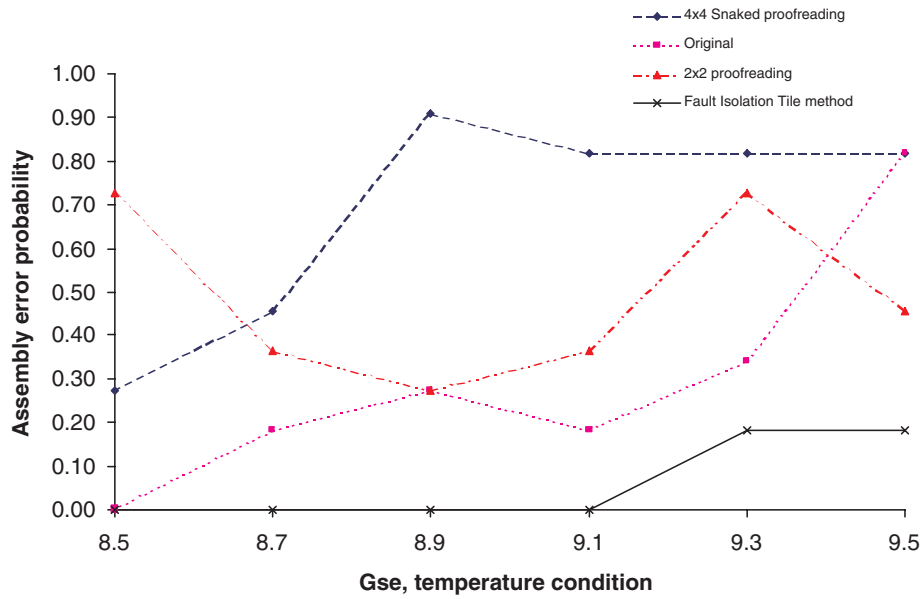


Figure 6. Assembly error probability vs. G_{se} , $G_{mc}=15$ for a 3×20 rectangular assembly.

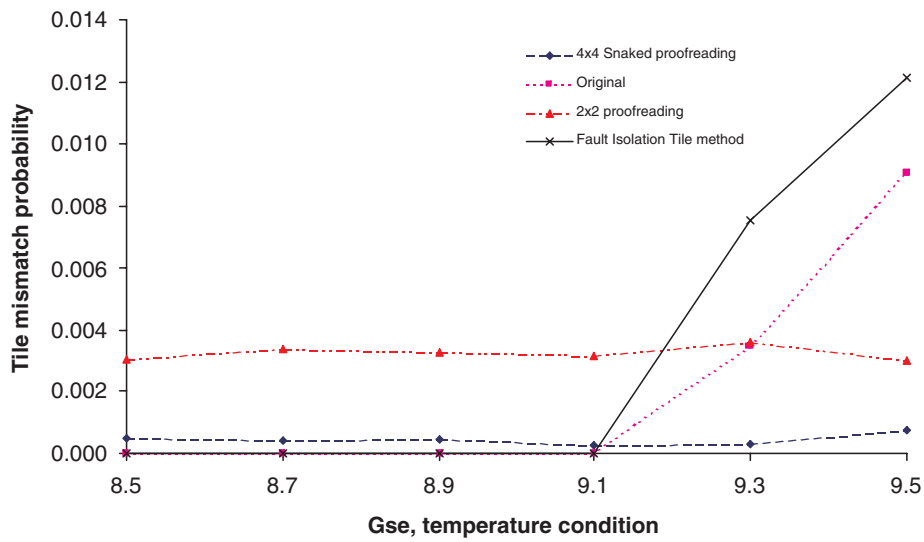


Figure 7. Tile mismatch probability vs. G_{se} , $G_{mc}=15$ for a 3×20 rectangular assembly.

- [4] P. Rothemund, N. Papadakis, and E. Winfree “Algorithmic Self-Assembly of DNA Sierpinski Triangles,” *PLoS Biology*, 2 (12). pp. 2041-2053. ISSN 1544-9173 2004.
- [5] E. Winfree. Interviewed in: Technology Research News. *Programmed DNA Forms Fractal*, by Kimberly Patch, Technology Research News, May 4, 2006
- [6] E. Winfree, *Self-Healing Tile Sets*, Foundations of Nanoscience: Self-Assembled Architectures and Devices, 2005, pp. 21-22.
- [7] H. Yan, S. H. Park, G. Finkelstein, J. H. Reif, and T. H. LaBean. “DNA-templated self-assembly of protein arrays and highly conductive nanowires,” *Science*, 301:1882:1884, 2003.
- [8] B.H. Robinson and N.C. Seeman, “The Design of a Biochip: A Self-Assembling Molecular-scale Memory Device,” *Protein Engineering*, vol. 4, pp. 295-300, 1987.
- [9] M. Cook, PWK Rothemund, and E. Winfree, ”Self-Assembled Circuit Patterns,” *DNA Based Computers 9*, 2004, pp. 91107.
- [10] H.L. Chen and A. Goel, “Error Free Self-Assembly using Error Prone Tiles,” *Proc. of the 10th Int’l Meeting on DNA Computing (DNA10)*, Univ. of Milano-Bicocca, June 2004.
- [11] “DNA damage checkpoint,” Technical Document, European Bioinformatics Institute, Wellcome Trust Genome Campus, Cambridge, UK, March. 2006. [Online]. Available: <http://www.ebi.ac.uk/ego/DisplayGoTerm?id=GO:0000077&selected=GO:0031571>
- [12] A. Carbone and N.C. Seeman, Circuits and Programmable Self-Assembling DNA Structures, *Proc. of National Academy of Science, USA* 99:12577-12582, Sept. 13, 2002. [Online]. Available: <http://www.pnas.org/cgi/content/full/99/20/12577>
- [13] H. Yan, X. Zhang, Z. Shen and N.C. Seeman, “A Robust DNA Mechanical Device Controlled by Hybridization Topology,” *Nature*, vol. 415, pp. 62-65, 2002.
- [14] Y. Baryshnikov, E.G. Coffman, N. Seeman and B. Yimwadsana, “Self-Correcting Self-Assembly: Growth Models and the Hammersley Process,” *Proc. of the 11th Int’l Meeting on DNA Computing*, London, Ontario, 2005.
- [15] J.H Reif, S. Sahu, and P. Yin, “Compact error-resilient computational dna tiling assemblies,” *10th Int’l Meeting on DNA Based Computers*, Lecture Notes in Computer Science, Springer-Verlag: New York, 2004.
- [16] K. Fujibayashi and S. Murata, “A Method of Error Suppression for Self-assembling DNA Tiles,” *Proc. 10th Int’l Meeting on DNA Based Computers*, Lecture Notes in Computer Science, Springer Verlag: New York pp. 284-293, 2004.
- [17] Kishor Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, John Wiley and Sons, Inc. USA 2002, pp. 173, 673.
- [18] E. Winfree, “Simulations of Computing by Self-assembly,” Tech. Report CS-TR:1998.22, Caltech, 1998.
- [19] K.F. Wong and M. Franklin, “Checkpointing in Distributed Computing Systems,” *J. of Parallel and Distributed Computing*, Vol. 35, pp. 67-75, 1996.
- [20] S Frechette and F. Lombardi, “Error Detection/Correction in DNA Algorithmic Self-assembly” *Proc. IEEE DATE08*, Munich, April 2008.