

# DNA Computing and Molecular Self-Assembly

## **Area Exam**

Serafim Batzoglou

# 1. Overview of DNA Computing

Adleman (1994) demonstrated how to solve an instance of the Directed Hamiltonian Path (DHP) problem by encoding it in DNA and subsequently using a biological protocol that can create and extract the solution in a small number of steps. The main attraction of this method of performing computation lies in the potential of massive parallelism resulting in a greater number of computations per second than the fastest supercomputers could perform (Adleman, 1994, Gifford, 1994). Other motivations include the information-encoding density of DNA that exceeds that of conventional computers, and the energy efficiency of enzymes, that could result in very low energy computations (Adleman, 1994, Gifford, 1994). The idea of using DNA to perform computations is earlier than this work, as it appeared in the splicing systems of Head (1987, 1992) that propose the use of one-dimensional self-assembly of double-stranded DNA together with enzymes capable of cutting DNA at sites that contain specific patterns.

Below we discuss the main ideas in the method of Adleman (1994), and in some of the related work that emerged. We will try to briefly cover the main issues involved in using this work to perform practical computations. We assume that the reader is familiar with the basic structure of single- and double-stranded DNA molecules, their basic properties such as annealing of complementary strands, and basic biological assays such as Polymerase Chain Reaction (PCR), magnetic affinity purification, and gel electrophoresis.

The DHP problem asks, given a directed graph, if there is a path visiting every node exactly once (called *hamiltonian*). The problem is NP-complete, therefore no known polynomial time algorithm exists (Papadimitriou, 1994). One possible algorithmic solution that takes exponential time to serially execute, involves generating all possible paths and then examining each (in a linear number of steps) if it is hamiltonian.<sup>1</sup> Adleman's method essentially implements the above algorithm using DNA molecules. An encoding of vertices and edges of the graph in single-stranded oligonucleotides enables the generation of a large number of random paths encoded in double-stranded DNA, which probabilistically contains a solution if there exists one. The large number of random paths is screened by an array of biological procedures: PCR, magnetic bead separation, graduated PCR, and gel electrophoresis. These procedures remove paths that are not hamiltonian, until any solutions are detected. Adleman estimated that the number of above procedures needed grows linearly with the number of vertices in the graph. Adleman solved with the above method a 6-vertex DHP instance containing a hamiltonian path.

It was immediately pointed out that Adleman's method in principle could solve DHP and maybe similar problems for instances that are much larger than those handled by conventional computers (Adleman, 1994, Gifford, 1994). Adleman gave the plausible estimate of  $10^{20}$  operations performed within reasonable laboratory time. This would far exceed current supercomputers and thus a special-purpose DNA computation might be able to handle a larger instance of DHP or other important problems.

At the same time, some of the limitations of this first method of DNA computation became immediately apparent. Adleman pointed out three such important limitations. One is the scalability of the volume of DNA required to solve larger instances of exponential-search problems. Whereas the number of different oligonucleotides encoding a problem instance would

---

<sup>1</sup> It should be noted that much more efficient algorithms exist for this important problem, often only applicable to special instances of some interest. See for example (Chvatal, 1985, Gould, 1991, Vandegriend, 1998). This is in general an important point when evaluating the potential of DNA algorithms to beat more conventional algorithms. This is also pointed out by Bach et al. (1996).

grow linearly with the size of the instance, the quantity of DNA used should be enough to guarantee probabilistically that all possible solutions will be generated. That would require at least exponential quantity of DNA. Another problem is the possibility of errors in all the biological screens, as the size of the problem increases. Such possible errors include incorrect ligations, as well as false positives and false negatives during screening steps. A third limiting factor pointed out by Adleman is the lack of flexibility of his method to perform general computations. Finally, it was pointed out that the extraction of solutions was too labor intensive (Gifford, 1994).

Subsequent work focused on various aspects of DNA computing. Some of the topics that researchers have investigated are: demonstration of solutions to problems other than DHP, comments on Adleman's method and proposed improvements, demonstration of more versatile, and even universal, protocols of computing with DNA, various theoretical studies, and different methods for performing DNA computing, that exhibit theoretical or practical advantages. Below we give a brief mention of some of this work.

Lipton (1995) presented a method for solving the 3-SAT problem in  $O(n)$  lab steps using DNA of length  $O(n \log n)$  base pairs. Adleman (1995, 1996) presented a molecular algorithm for solving the 3-COLORING problem. These algorithms used search in solution spaces of size  $O(2^n)$  and  $O(3^n)$  respectively. This led Bach et al. (1996) to propose DNA computation algorithms for 3-COLORING and INDEPENDENT SET that have an exponent basis lower than 2, and thus could be proven more practical in terms of volume of DNA needed. Adleman et al. (1996) described a plausible attack to the US Data Encryption Standard using DNA for computation. Baum and Boneh (1996) demonstrated how to use DNA computers to run dynamic programming. Ouyang et al. (1997) solved the maximal clique problem on a molecular experiment, on a graph with 6 vertices and 11 edges.

There have been several studies on the limitations of Adleman's method, and suggestions for improvements. In particular, a topic of interest has been the error tolerance of DNA algorithms. Boneh and Lipton (1995, 1996) show how Lipton's SAT algorithm (1995) could be made error resistant, and how the technique could be extended to other algorithms. Baum (1996) considers the problem of incorrect ligations, and suggests some constraints on oligonucleotides that are used in encoding problem instances, so as to avoid that error. Reif (1995) presents some techniques for reducing the error probabilities. That however involves a slowdown in the computation. Deaton et al. (1998) study the errors associated with imperfect hybridization (false positive and false negative ligations). They point out that these errors depend largely on the reaction conditions of the hybridization, and especially on temperature. They give an estimate of the size of the largest problem instances that can be encoded with nucleotides of a certain length, in experiments similar to Adleman's (1994).

A question raised by Adleman (1994) was whether DNA could be used for universal computation. This has been shown in various ways by Adleman (1995), Boneh et al. (1996), and Wilhelm and Rothermund (1996). Winfree (1995) claims that the operations of "annealing" and "ligation" are capable of universal computation. In that paper a 2-dimensional self-assembly model of DNA computation is introduced, and proved universal. We will discuss this model more in detail in section 3. No method has been developed however, that has been experimentally verified to demonstrate the versatility of a Turing machine even for very small programs.

Various enhancements and alternative models to Adleman's basic method have been proposed. Lipton (1994) described three basic operations on test tubes, capturing Adleman's method: *extract, detect, and amplify*. Those correspond to Adleman's magnetic bead separation, gel

electrophoresis, and PCR, respectively. Roweis et al (1996) introduce the “sticker model” for DNA computation, exhibiting a random access memory. Amos et al. (1996a,b) describe a model using restriction enzymes instead of successive cycles of separation by DNA hybridization, which thus they claim should have improved error-tolerance. The splicing systems introduced early by Head (1987, 1992) represent a significant body of literature. See for instance (Culik & Harju, 1989, Ferreti & Kobayashi, 1996, Bohn & Paun, 1997, Freund et al., 1997). In the following sections we will discuss in more detail some approaches to DNA computing that are considerably different from Adleman’s initial method.

We would like to mention that there has been so far little progress towards demonstrating a DNA computation that cannot be trivially performed in conventional machines. It is a very important open question to decide whether DNA computing will at some point reach the level of being useful from a practical point of view.

## 2. Whiplash PCR for DNA Computing

Recently Hagiya et al. (1999) proposed a new method of DNA computing that involves a self-acting DNA molecule containing both the input, program, and working memory. As such, this method is unrelated to previous models of DNA computation (Hagiya et al., 1999). Its main motivation is that self-acting molecules can compute in parallel in a single-tube reaction allowing for a “multiple program, multiple input” architecture (MPMD). Hagiya et al. showed how to theoretically learn  $\mu$ -formulas using this model. Subsequently, Adleman called this method “whirlash PCR” (Winfree, 1998b). Winfree (1998b)<sup>2</sup> showed how to solve several NP-complete problems in “O(1) biosteps”.

### 2.1. Overview of the method

In this method, a single-stranded DNA molecule consists of an input segment on the 5’-end, followed by a formula (program) segment, followed by a spacer, and finally with a “head” on the 3’-end that moves and performs the computation. The head anneals to complementary subsequences in the input and formula alternatively. Each time, a symbol is copied to the 3’-end effectively shifting the head. Subsequently the 3’-end is denatured to single-stranded again and moves to the next step. The computation ends when special symbol “out=1” or “out=0” is copied to the head. The success of the method depends on a technique introduced and experimentally verified in (Hagiya et al., 1999), called “polymerization stop”.

The formula portion consists of a  $\mu$ -formula, which is a boolean formula with each variable occurring once. In that way there is no ambiguity as to where the head will anneal. At each step it can anneal only to one place.<sup>3</sup> Hagiya et al. (1999) propose a procedure for learning  $\mu$ -formulas with this method: let all  $\mu$ -formulas be generated randomly, and attach to them positive and negative examples, selecting the formulas that satisfy the desired conditions. Pit and Valiant

---

<sup>2</sup> Winfree’s work (1998b) follows Hagiya et al.’s (1999) work which was presented on the 3<sup>rd</sup> DIMACS meeting on DNA Based Computers, June 1997.

<sup>3</sup> Winfree (1998b) proposes the possibility of more parallelism introduced in this method by allowing for non-determinism in the head annealing.

(1988) showed that  $\mu$ -formulas are not learnable from examples in polynomial time. Hagiya et al. also show that a single molecule is capable of simulating a finite state automaton.<sup>4</sup>

On the experimental front, Hagiya et al. demonstrated how to perform the polymerization stop reaction, and were able to observe two successive head transitions of a very simple version of their self-acting molecule.

## 2.2. Discussion

### 2.2.1. Expressive power of the model

Hagiya et al. (1999) showed how one molecule can evaluate a  $\mu$ -formula and how many molecules in a tube can learn  $\mu$ -formulas from examples in polynomial time. Winfree (1998) showed how to solve several NP-complete problems within the model. We claim that one molecule has exactly the computational power of a finite state automaton. We assume that the head never aligns to a place to the 3' of the spacer. This is an implicit assumption of Hagiya et al. We do not need the assumption that the segment 5' from the spacer contains an input and an  $\mu$ -formula. It suffices to say that it contains triplets (*STOP*,  $x$ ,  $y$ ) whereby the head can anneal to  $y$ , and copy  $x$  to be the new head. Notice that we allow non-determinism in the head annealing.

The simulation by a non-deterministic FSA is trivial: for each triplet (*STOP*,  $x$ ,  $y$ ), construct a state  $x$ , a state  $y$  (if not existing already) and an empty transition from  $y$  to  $x$ . Since deterministic and non-deterministic FSA are equally powerful, one molecule computes exactly the regular languages.

The power of an exponential number of such molecules is unclear. It would depend on certain assumptions. For instance, whether the number of different molecules that end the computation with an *accept* can be counted by some assay, can be approximated, or the only inferable information is whether *some* molecule accepted. If the number of different molecules accepting could be counted, then #P problems could be solved such as number of 3SAT satisfying assignments. We believe it is premature to discuss such theoretical questions in detail since little is known about the capability of whiplash PCR to perform computations in practice.

### 2.2.2. Whiplash PCR as a practical means to perform computations

Little progress has been made on the experimental side to demonstrate that whiplash PCR is capable of performing useful computation. An obvious question is how large a single-stranded molecule can be, and how much longer it can grow during computation, without breaking. This question would pose limits on the size of problems that the method can handle. It should be possible to address it experimentally, with gel electrophoresis. The experiments performed by Hagiya et al. (1999) involve very short molecules whereby one would not expect to observe strand breaking.

Another problem that is mentioned by Hagiya et al. (1999) and should be addressed is the possibility of different molecules interacting. They propose to keep the concentration sufficiently low. It should be possible to address this problem experimentally, also possibly by gel electrophoresis. In particular, determine (1) at what concentration molecules start to interact with each other at a significant frequency (forming aggregates observable in the gel), and (2) how

---

<sup>4</sup> Deterministic, or non-deterministic; those two are equivalent.

these interactions scale with the size and number of molecules used. Allowable levels of concentration in order to minimize interactions could very likely be dependent on the size of molecules. They should be investigated so that the crucial volume requirements of the method are determined.

Also mentioned in Hagiya et al. (1999) is the problem of slowdown caused by reannealing of the head to the previous step of the computation. This is probably a very significant slowdown since there is a larger matching portion of the head at that location, than at the “correct” next location. An even more severe problem would be formations of loops in different parts of the sequence. Yet another potential problem would be occasional wrong annealing of the head, and subsequent extension of it. This would cause errors in the computation. In order to minimize it, the encoding would need to be highly redundant, possibly using the results of Baum (1996).

One issue not mentioned in Hagiya et al. (1999) is how reliably the procedure for evaluating  $\mu$ -formulas they describe, could be realized. It involves generating at random a large pool of formulas, which then have to be attached reliably to input, spacer, and head ssDNA, and subsequently go through several cycles of selection, detachment from input, and attachment to new input. All those steps could be error prone.

Experiments could be designed to assess all the above issues, and determine what are the error rates and the physical limits of the method. In order to perform reliable computations, the method would probably need to rely on error-tolerant algorithms, and/or be enhanced with chemical techniques that decrease the error rates.

Turning to the question of whether the method is promising for practical computations, we observe that in their experiments Hagiya et al. (1999) managed to demonstrate two successive steps of computation, in very short molecules. Those steps involved several steps programmed into Perkin Elmer Thermal Cycler technology. The total number of steps took around 10 minutes. Although it is possible that all those steps could be sped up somewhat, there is no mention in the paper of Hagiya et al. (1999) of ideas towards this direction. Given issues addressed above including potential errors and slowdowns, one would expect possibly slower cycle time for much larger, complicated molecules that would be needed to perform a realistic computation. In that case it is difficult to see how this method would outperform conventional computers. Experimentally exploring the limits of this method would enable to make assessments of the size and nature of problems that could realistically be solvable.

## 2.3. Conclusions

Hagiya et al. (1999) have shown in theory and experiment, that a single ssDNA molecule can act as a FSA under controlled conditions. This is an important discovery and could potentially be useful, as part of a larger experiment for instance. From the point of view of DNA Computing, it adds another model of computation with the advantage of single-tube computation of many programs in parallel. This is in contrast to Adleman’s method (1994). On the other hand, this method faces serious problems and limitations that need to be addressed experimentally. In any case, computation with this method is probably bound to be slower, and have higher error rates than competing methods. A fundamental difficulty is the use of single-stranded DNA that is expected to cause lower stability and more misalignments, aggregations, and strand breakages.

### 3. DNA computation by Self-Assembly

Winfree (1995) introduced a 2-dimensional self-assembly model for DNA computation. In this model, DNA double-crossover (DX) molecules<sup>5</sup> with four sticky single-stranded ends behave as tiles in a two-dimensional assembly. These sticky ends can assume different types, and tend to attach to other sticky ends of the same type. This results in spontaneous self-assemblies of the tiles into conformations that can in principle perform any deterministic or non-deterministic computation. The attraction of this method over more conventional methods such as Adleman's (1994) is that the time-consuming and error-prone laboratory steps are all replaced by one experiment in a single test tube.

Following this work, Winfree et al. (1998) made an important experimental demonstration of the feasibility of spontaneous two-dimensional self-assembly of double-crossover molecules. Winfree (1998a) developed a thermodynamics and kinetics model for the self-assembly process and studied *in silico* the error rates of the system under different physical conditions. LaBean et al. (1999) introduced new types of molecular tiles that are more versatile. Other research in self-assembly models of computation includes (Eng, 1997) studying a self-assembly model that generates the context-free grammars and (Yokomori, 1999) that introduces YAC (Yet Another Computation) self-assembly model that is Turing-universal.<sup>6</sup> There is a substantial body of work on DNA self-assembly in relation to nanotechnology, rather than computation. We will discuss such work in section 4.

#### 3.1. Description of main results

Winfree (1995) proposed several different self-assembly models. One model involves linear self-assembly of DNA molecules. Another model involves two-dimensional self-assembly. Yet another involves three-dimensional self-assembly. Later work has focused on the two-dimensional self-assembly model that was introduced. Below we give a brief description of the model.

Computation in this model is performed by the formation of *aggregates* by placing single new tiles (*monomers*) next to other ones of matching labels. A "temperature"  $T$  controls this reaction so that the combined strength of matching labels for monomer addition has to exceed  $T$  (*stable additions*). At any stage the aggregate can be seen as representing a computation history. Allowing for an arbitrary number of different types of labels and *rule* tiles makes the model Turing-universal.<sup>7</sup> Nondeterminism is introduced in the model by the possibility of many different rule tiles to be stably added and extend the aggregate in the same location. Therefore the model allows for parallelism in two important ways: Firstly, an aggregate can be extended in many different locations simultaneously across its perimeter. Secondly, different rule tiles may fit at the same location in the perimeter, which allows for the possibility of simultaneous formation in a tube of several aggregates each following non-deterministically a different computation history.

Winfree (1998a) proves that the tiling assembly model is Turing universal. This is achieved by representing with tiles the entire computation history of a special kind of cellular automaton.

---

<sup>5</sup> Special DNA molecules consisting of two double strands with two crossovers, with structural properties that make them appropriate for forming 2-dimensional lattices, refer to Fu & Seeman (1993).

<sup>6</sup> The splicing systems of Head (1987) use linear self-assembly of DNA, together with restriction enzymes that contribute to their computational power. Therefore that is not a purely self-assembly method.

<sup>7</sup> In the actual proof label strengths of 1 and 2, and a temperature  $T = 2$  are used.

Winfree (1998a) claims that double-crossover DNA molecules can be used as tiles, by using the four sticky ends as labels. Label strengths could be implemented by designing the sticky ends to have different lengths, and hence different hybridization strengths. Winfree (1998a) points out that this abstract ideal model cannot be perfectly realized with double-crossover DNA molecules. A more realistic thermodynamic/kinetic model is presented and run in computer simulations to provide insight into the plausible real behavior of the system. A simple deterministic structure consisting of 7 rule tiles is formed in those simulations.

The thermodynamic/kinetic model simulations rely on certain assumptions. Most notably, the assumption that monomer concentrations are held constant, and no interaction between different aggregates occurs. The simulations demonstrate a tradeoff between error rate<sup>8</sup> and rate of crystal growth. Error rates can decrease by setting the temperature close to the “melting” temperature where crystal growth is marginal, by increasing binding strengths, and by decreasing molecular concentrations. Any target error rate is achievable by sufficiently reducing growth rate. Unfortunately growth rate is dependent on the square of the error rate.

Winfree et al. (1998) experimentally demonstrated the formation of 2-dimensional sheets of double crossover molecules. They were able to observe crystals consisting of up to 500,000 units. Growth took in the order of 36 hours. Unfortunately the possible error of wrong tile placements was not measured in these experiments.

## 3.2. Discussion

### 3.2.1. Algorithms for some NP-complete problems in the tile assembly model

Tile assembly is Turing-universal. However, from a practical perspective it is important to know exactly what it involves to perform difficult computations in this model.<sup>9</sup> In order to attempt solve NP-complete problems it would hardly be sufficient to rely on a simulation of the computation history of a cellular automaton. Rather, as Winfree (1998) suggests, one would like to exploit the opportunity for parallelism coming from the simultaneous formation of many aggregates. Given an instance of size  $N$  of a certain problem then, it would be interesting to know how large such aggregates would need to grow.

#### 3SAT:

We provide a solution of 3SAT in the tile model, with the use of random generation in the spirit of Adleman (1994). Our construction is described below. Figure 1 shows all the kinds of rule tiles and a schematic of how a specific aggregate would be formed. Briefly, an assignment is first chosen that satisfies all the clauses. This possibly contains contradictions whereby a variable is true in a certain clause, and false in another. The assignment is “propagated” then to check for inconsistencies. If an inconsistency is found, a special tile “FALSE” is inserted. This tile could be used for screening. Alternatively, we could alter slightly the construction to include reporter tiles

---

<sup>8</sup> Proportion of wrong tiles “fixed” in an aggregate.

<sup>9</sup> It should also be noted that if we limit the sticky-end lengths to some realistic maximum, the method is not Turing-universal. Of course in conventional computers the amount of memory is not infinite, thus strictly speaking they are not universal either. But a realistic constraint on sticky-end lengths, and a realistic requirement of maximum allowable alignment between different sticky ends, would probably be much more limiting to the model, than the maximum memory and time allowed in realistic conventional computers.

for “TRUE”, i.e. for satisfying assignments. A combination of “TRUE” and “FALSE” tiles would also be possible. These details are not shown.

Below we will be denoting tiles with labels a, b, c, d clockwise, by (a, b)-(d, c). Capital letters denote labels of strength 2. 0 denotes no label (label of strength 0). Otherwise labels have strength 1. We assume that the growth temperature is 2.

Given formula  $\varphi = (u_1 \vee v_1 \vee w_1) \wedge (u_2 \vee v_2 \vee w_2) \wedge \dots \wedge (u_m \vee v_m \vee w_m)$  in 3CNF of  $m$  clauses, where all literals  $u_i, v_i, w_i$  are variables  $\{x_1, \dots, x_k\}$  or their negations, and without loss of generality  $m$  is even, construct the following tiles:

1. *First layer* tiles:  $(0, 0)-(A_1, A_1), (0, 0)-(A_2, A_2), \dots, (0, 0)-(A_l, A_l)$  where  $l = \frac{3}{2}m - 1$ .
2. *First choice* tiles:  $(0, A_1)-(0, a_1), (A_1, A_2)-(b_1, b_1), (A_1, A_2)-(c_1, c_1), (A_2, A_3)-(a_2, a_2), (A_2, A_3)-(b_2, b_2), (A_3, A_4)-(c_2, a_3), (A_3, A_4)-(b_3, b_3), (A_3, A_4)-(c_3, c_3), \dots, (A_{m-1}, A_m)-(a_m, a_m), (A_{m-1}, A_m)-(b_m, b_m), (A_m, 0)-(c_m, 0)$ .
3. *Second choice* tiles:  $(a_1, b_1)-(0, u_1), (a_1, c_1)-(0, u_1), (a_1, b_1)-(0, v_1), (a_1, c_1)-(0, w_1), (b_1, a_2)-(g, g), (c_1, a_2)-(g, g), (b_1, b_2)-(g, g), (c_1, b_2)-(g, g), (a_2, c_2)-(u_2, u_2), (a_2, c_2)-(w_2, w_2), (b_2, c_2)-(v_2, v_2), (b_2, c_2)-(w_2, w_2), \dots, (b_{m-1}, a_m)-(g, g), (c_{m-1}, a_m)-(g, g), (b_{m-1}, b_m)-(g, g), (c_{m-1}, b_m)-(g, g), (a_m, c_m)-(u_m, g), (a_m, c_m)-(w_m, g), (b_m, c_m)-(v_m, g), (b_m, c_m)-(w_m, g)$ .
4. *Fourth layer* tiles:  $(x_1, g)-(x_1, x_1), (\neg x_1, g)-(\neg x_1, \neg x_1), (g, x_1)-(x_1, x_1), (g, \neg x_1)-(\neg x_1, \neg x_1), \dots, (x_k, g)-(x_k, x_k), (\neg x_k, g)-(\neg x_k, \neg x_k), (g, x_k)-(x_k, x_k), (g, \neg x_k)-(\neg x_k, \neg x_k)$ .
5. *Propagation* tiles: for all  $u_i, u_j$  in  $\{x_1, \dots, x_k, \neg x_1, \dots, \neg x_k\}$ , with  $u_i \neq \neg u_j$ , the tile  $(u_i, u_j)-(u_j, u_i)$ .
6. *FALSE* tiles:  $(x_i, \neg x_i)-(FALSE)$ .

The *FALSE* tiles above can be implemented in some way that we will not specify. It would depend on the easiest way to screen out aggregates containing those tiles. The above construction assembles as a triangular aggregate, and completes only if the formula is satisfied with the chosen assignment.<sup>10</sup> If for practical reasons aggregates that contain satisfying assignments need to contain a special tile, a modified construction that involves keeping some more tiles in the boundary of the above triangular construction, would enable to insert a *TRUE* tile at the tip of a successful assembly. We do not show the details. The construction is slightly more complicated and less intuitive than another, which would involve on the other hand larger aggregates. We give the above construction because the size of the aggregate is an important consideration.

It is tedious but easy to give a full proof that the above construction works. We only sketch a proof here. The first and second layers interact with strength-2 labels that enable their formation. The “bottom” of the second layer consists of labels representing the literals at the order in which they appear in  $\varphi$ , except that if two literals from the same clause coincide in the same tile, then one of the two is *chosen* and the corresponding label appears twice. The third layer consists of two types of tiles (not including the boundary tiles): ones that have at the “top” labels from two different clauses, and make no choice (simply propagate the neutral label  $g$ ); and ones that have at the “top” labels from the same clause in  $\varphi$  and “choose” one of them. At the bottom of the third layer, there are faces either containing the neutral label  $g$ , or containing a chosen literal for some clause. Each clause has exactly one chosen literal, from now on assumed true. The fourth layer tiles consist of tiles that get rid of  $g$ , and of tiles of types (5), (6). All further layers only contain tiles of types (5), (6). Those tiles simply propagate the information of the literals that each clause

<sup>10</sup> The last “tip” tile of a complete triangular aggregate can be a *FALSE* tile, in which case the formula is still not satisfied.

fixed to true, to the left and to the right. If at some point in the assembly it is detected that a variable  $x$  has been set to *true* and to *false*, a special *FALSE* tile is inserted.

The above construction involves:

1.  $4k^2 + 4k + 10m$  different rule tiles. Calculation not shown.
2. Aggregates that grow in almost triangular shapes, of dimensions  $(1.5m \ 1.5m)$ , with number of tiles in an aggregate  $\leq \frac{9}{8}m^2 + \frac{9}{4}m - 1$ . Calculation not shown.

Therefore, a 40-variable SAT problem with say 40 clauses, would involve 6,960 rule tiles, and aggregates of size at most 1,889 tiles. A SAT problem with the same number of variables and 100 clauses would involve 7,560 rule tiles, and aggregates of size  $\leq 11,474$ .<sup>11</sup> However, as Winfree (1998) points out, error rates could be a problem: if the satisfying assignments are exponentially fewer than the non-satisfying ones, the acceptable error rate would need to be exponentially small in order to prune out enough the wrong assignments.

### MAX INDEPENDENT SET:

A tile set that attempts to solve MAX INDEPENDENT SET is even simpler. Given undirected graph  $G = (V, E)$ , find maximum number of vertices among  $V = \{v_1, \dots, v_m\}$  not sharing an edge. We provide the tile set below:

1. “Left top” tile  $(0, 0)-(0, A)$ ; “Middle top” tile  $(0,0)-(A, A)$ ; “Right top” tile  $(0, 0)-(A, 0)$ ; whereby probably need “middle top” tile at higher concentration than the other two.
2. Tiles  $(A, A)-(v_1, v_1), \dots, (A, A)-(v_m, v_m)$ .
3. For each  $v_i \neq v_j$  such that  $\{v_j, v_j\} \notin E$ , tile  $(v_i, v_j)-(v_j, v_i)$ .
4. For all rest pairs  $v_i, v_j$ , including pairs  $v, v$ , the tile  $(v_i, v_j)-(FALSE)$ .

As before, we can choose if we want to use alternative reporting strategies, such as reporting success, or both success and failure. Details are not shown. It should be clear how the above construction works, to choose initially a random multiset of nodes, and later check that it is a set (and not a multiset) of nodes, and that it contains no pair of nodes with a common edge. Presumably, we could screen all unsuccessful candidate aggregates, and then somehow pick the largest successful aggregate. This may not be trivial of course, and in a non-ideal error prone system may be impossible.

The above construction uses  $m^2 + m + 3$  different tiles, and arbitrarily large aggregates that need not be larger than  $\frac{1}{2}(m+1)(m+2)$ , or just  $\frac{1}{2}(k+1)(k+2)$  to solve a k-INDEPENDENT SET.

### MAXIMAL CLIQUE:

The same above construction can be used to solve the MAXIMAL CLIQUE problem if tiles of types (3,4) above, instead of detecting edges connecting nodes, detect absence of edges connecting nodes. The same calculation of the number of tiles and size of aggregates carries to the MAXIMAL CLIQUE problem.

---

<sup>11</sup> Regarding Winfree’s comment (Winfree 1999, p. 24) on solving a 40-variable 3SAT with  $10^{13}$  aggregates of size 300, we do not believe that interesting instances of 3SAT could be solved with such small aggregates. However, his calculation probably falls within the correct order of magnitude or two.

In order to solve a  $k$ -INDEPENDENT SET, or  $k$ -MAXIMAL CLIQUE problem, where  $m = 100$  and  $k = 50$ , say, one would need 10103 different rule tiles and aggregates of size at most 1326. Again, we are ignoring practical problems that would very likely be prohibitive.

### **k-COLORING:**

We only provide a sketch of a tile assembly that solves  $k$ -COLORING: Like in 3SAT above, the first and second layers of an aggregate are constructed so as to provide a scaffold where each vertex of the graph is represented exactly once. The tile representing a vertex of the graph contains labels in the bottom left and right, encoding the vertex id, and the chosen vertex color. The same propagation technique used in all above problems can transmit to the left and right the information of each vertex color, stopping the transmission locally and inserting FALSE when two vertices of the same color connected with an edge are detected.

We do not believe that most NP-complete problems have similarly simple descriptions in the tile assembly model. For instance, it is not clear to us how to check with equal simplicity the Hamiltonian property of a path.

### **3.2.2. Self-Assembly as a practical means to perform computations**

Clearly on the theoretical front the self-assembly approach of Winfree is very attractive, where simple one-tube experiments with little requirement for laborious interventions, provide a versatile system for expressing an array of difficult computational problems in a very elegant mathematical way. Moreover, it is at least imaginable, that the requirements imposed by the algorithms described above in terms of number of tiles, and aggregate sizes, could be met to attempt problem instances of interesting size.

However little has been achieved in the experimental front. Winfree's simulations provide a good starting point to ask the right experimental questions. We would propose experimental verification of many of the simulation results, as well as experiments assessing the degree in which the assumptions of the kinetic/thermodynamic model and simulations hold in real conditions.

One important assumption of the tiling model that has not been experimentally demonstrated is that labels can have different strengths. Winfree claims that sticky end lengths of the DX molecules can determine the strengths of edge labels. In practice however, this may result for instance in non-planar assemblies. It seems that the ability to have labels of strengths either 1 or 2 helps in expressing problems in the tile assembly model, so this would be a property of high priority for experimental verification. Many experiments could be devised to test this. We propose for instance the formation of a simple half-square assembly with a single corner tile  $(0, 0)$ - $(A, A)$ , edge tiles of forms  $(A, 0)$ - $(a, A)$  and  $(0, A)$ - $(A, a)$ , and interior tiles  $(a, a)$ - $(a, a)$ . It should be relatively easy to initially check the fraction of aggregates containing  $(a, a)$ - $(a, a)$  that do not contain the corner residue, and compare it with theoretical/simulation predictions. Subsequently, more elaborate tests should explore this assumption in more detail.

Alternatively, Winfree (1998a) mentions the possibility of creating labels with negative weights. Sticky ends that do not agree maybe in practice will prove to constitute negative weight, or at least this may be possible to ensure. Negative weight labels, plus labels of weight 1, may be just as good as labels of weights 1 and 2 that we have been using in the previous section to express several NP-complete problems. They could help by allowing reducing the temperature of the assembly while preventing wrong monomer incorporations.

Assumptions of the thermodynamic/kinetic model need experimental verification. Most notably, the assumption that aggregates don't interact. Aggregates, especially small ones, could interact and cause errors. The degree of this would probably depend on the particular rule tile sets. For example tiles  $x$ ,  $y$ ,  $z$ , and  $w$  where  $(x, y)$  and  $(z, w)$  can assemble with strength-2 bonds and  $(x, z)$  and  $(y, w)$  interact with strength-1 labels. At temperature  $T=2$  these should not aggregate very frequently to a 4 tile configuration, if the assumption of sequential addition of monomers holds to a good approximation. A direct experiment could measure the discrepancy from assumptions. Turberfield et al. (1999) point out the problem of interaction between aggregates in Winfree's model, and propose a method of "protected self-assembly" where hybridization is controlled with processes of *protection*, and *catalysis*. This is an interesting approach that we will discuss more in the following section. We should mention here, however, that the application of it has not been experimentally demonstrated, and that the approach has some clear drawbacks such as a significant slowdown of aggregate growth.

Winfree (1998a) concludes that the proportion of aggregates of a certain size produced by the kinetic assembly model, and not by the tile assembly model, is exponentially small. This guarantee is not necessarily enough however, to solve problems where the solution space is exponentially larger than the correct solutions. Instances of intractable problems tend to be like that.

Winfree (1998a) examines kinetic traps, as a source of errors in the simulated assemblies. Specifically, if growth is faster than the time it takes to reach local reaction equilibrium of right/wrong monomer addition, wrong tiles can become "frozen" in the interior of the aggregate, as other tiles attach around them and trap them. We would like to point out the possibility that in practice this source of error may be less severe than in the thermodynamic/kinetic model of Winfree. The reason is that Winfree assumes the forward reaction rate of monomer addition is independent of sticky ends. That is, the addition rate is independent of whether the monomer matches the addition location. Only the drop rate depends on sticky end agreement. This simplifying assumption could be proven inaccurate. In case the forward rate is enhanced by matches, or reduced by mismatches, the kinetic traps could generate fewer errors than the simulations predict. Negative label strengths could also reduce kinetic traps, and are a promising idea to be further investigated.

Winfree (1998a) suggests that the first layer of an aggregate should serve as the input, and the last layer as the output. He furthermore suggests that SAT could be solved by finding the correct solution with ligation and PCR. It would be interesting to see an actual experiment that solves an instance of 3SAT, or some other one of the problems in the previous subsection. One additional idea would be to try to prune out, and maybe even recycle, aggregates that contain the tile "FALSE". That could reduce volume requirements.

### 3.3. Conclusions

Winfree's self-assembly approach to DNA Computing offers many advantages: easy and elegant expression of several NP-complete problems in the tile assembly model, single-tube computation that avoids labor and sources of errors in approaches such as Adleman's (1994), and a potential to improve with techniques such as in Turberfield et al. (1999). Moreover, the formation of stable aggregates that are large enough for practical computations has been demonstrated (Winfree et al. 1998). However actual computations have not been demonstrated with this method, nor has any error analysis been performed on aggregates of interesting sizes. In order to further explore this

method of DNA Computing, we would suggest two approaches that could be performed concurrently. One is to make experiments as suggested above to test the physical limits of the method and the validity of certain important assumptions. Another approach would be to pick the most promising NP-complete problem that could be handled with the method, and see for what instance sizes the first experimental difficulties with the method are encountered.

On the other hand, Winfree's method has potential in the area of fabrication of objects consisting of nanometer-sized components. This area could even result in tremendous speedup of conventional computing.<sup>12</sup> We will discuss more about this in the next section.

## 4. Control of Self-Assembly and nanofabrication

Besides DNA Computing, DNA self-assembly has increasingly been recognized as a means to nanotechnology (Whitesides et al., 1991, Winfree et al., 1998). In this section we will shift our focus to DNA self-assembly used in nanofabrication.

We will discuss some recent results in the control of DNA hybridization for the purpose of self-assembly with lower error rates (Turberfield et al., 1999), and in the construction of nanomachines (Kelly et al., 1999, Koumura et al. 1999, Mao et al., 1999, Turberfield et al., 1999). First we will mention broadly some recent work related to self-assembly in nanofabrication, and molecular machines. Then we will describe Turberfield et al.'s approach (1999), as it relates to self-assembly of DNA, and DNA Computing.

### 4.1. Recent experimental work on molecular self-assembly and nanofabrication

A rapidly expanding branch of material science uses molecular systems in order to control properties of materials, such as size, shape, and assembly into higher structures (Mann, 1993). Recent research on nanometer-scale electronics for instance, focuses in their integration into useful circuits (Braun et al., 1998). However there are difficulties in achieving inter-element wiring. DNA molecules can prove useful in exactly that, because of their well-understood structure and properties. It is generally understood that the construction of functional nanoscale electronics, will come from a "bottom up" approach (Alivasatos et al., 1996) as opposed from the traditional engineering top down approach,<sup>13</sup> and is likely to require self-assembly processes (Braun et al., 1998).

DNA has poor electrical characteristics, and for that reason many scientists study hybrid systems of DNA and other materials. In such constructions, they use the specificity of DNA interactions to direct the interactions between different kinds of nanoparticles. Braun et al. (1998) stretched DNA between two gold electrodes, and subsequently used it as a template for the growth of a 12 $\mu$ m-long, 100nm-wide, silver wire. Alivasatos et al. (1996) organized discrete numbers of gold nanocrystals into specific three-dimensional structures, based on base pairing of DNA. Coffey et al. (1996) demonstrated the self-assembly of semiconductor nanostructures, using DNA as a template. In this work, circular plasmid DNA was used as a template for the formation of rings of

---

<sup>12</sup> Making it even harder for DNA Computing to compete !

<sup>13</sup> For example lithography. See also (Davis, 1999).

cadmium sulfate nanoparticles. In this way the size and shape of semiconductor structures of several micrometers was controlled with a DNA template. Mirkin et al. (1996) attached DNA non-complementary oligonucleotides capped with thiol groups, to gold particles. Subsequently, they were able to assemble the gold particles in a controlled and reversible manner into larger aggregates. Such work is motivated by the useful optical, optoelectronic, and structural properties of DNA/nanoparticles assemblies. Such constructions could have several applications as chemical sensors, spectroscopic enhancers, as well as in nanostructure fabrication, and microimaging methods (Mirkin et al., 1996).

On the front of self-assembly of DNA and fabrication of DNA structures with no added materials, Seeman and his group provided several pioneering constructions, including branched junctions, knots, Borromean rings (Mao et al., 1997),<sup>14</sup> a cube, and a truncated octahedron (as mentioned in LaBean et al. 1999, reviewed in Seeman, 1998). Winfree et al. (1998) used DNA double crossover molecules, and the method of Winfree (1995) to form specific periodic nanometer-sized sheets.

A very recent third class of objects studied and constructed on the nanoscale level, is nanoscale machines. A “nanomechanical device” based on the B-Z (right-handed to left-handed) transition of DNA was recently constructed (Mao et al., 1999) from double crossover DNA molecules. Such a device could be very useful for molecular-scale information processing but is not capable of continuous movement of a true molecular motor (Davis, 1999). Recent attempts to create motors include Kelly et al.’s (1999) use of chemical energy to create unidirectional motion of a nanoscale “molecular ratchet”, and Koumura et al.’s (1999) structure that exhibits light-driven rotation between four different states.

Turberfield et al. (1999) demonstrated a novel nanomachine, which they call “molecular tweezers” after the function it imitates on the molecular level. In this machine, the hybridization of DNA provides the chemical energy to produce motion. The machine consists of two 44-base long arms of DNA, called B and C, each with a 20-base long end hybridized to the same 40-base long strand A, resembling a pair of tweezers. A fourth strand F subsequently hybridizes to the unhybridized portions of B and C to close the tweezers. Finally, another strand is used to remove F and reopen the tweezers. The force exerted by the closing of these tweezers was calculated to be around 15pN (Turberfield et al. 1999).

## 4.2. Protected Self-Assembly

Controlled self-assembly of DNA for nanofabrication or for computations, usually relies on the addition of monomers to a larger aggregate. An important problem that Winfree (1998a) does not adequately address, and that Turberfield et al. (1999) point out is the wrong incorporation of small aggregates (often two or three monomers stuck together) into a larger aggregate, producing errors and discontinuities in the assembly. One method to decrease the frequency of errors is slow growth at temperatures near the melting temperature, as implied by simulations (Winfree, 1998a) and experiments (Winfree et al. 1998). Turberfield propose a new technique for reducing errors caused by aggregate interactions, where a monomer has its sticky ends “protected” from interacting with other monomers, by the formation of a metastable secondary structure. They then

---

<sup>14</sup> A family of topological objects that consist of interlocking rings, with the property that removing any ring unlinks the remaining rings (Mao et al., 1997).

demonstrate a catalytic technique, which can be used to unfold the secondary structure and allow the monomer to be incorporated to the lattice.

It should be noted that the protection/catalysis procedure of Turberfield et al. has the important disadvantage of slowing down the monomer/aggregate interaction by a factor of 150. This could be counterbalanced with higher concentrations, given the potential benefits of the technique in terms of lower error rates caused by interactions between the aggregates. However, as Winfree (1998) describes higher concentrations would increase the problem of kinetic traps. It would still be interesting to experiment with higher concentrations of monomers, in order to see how the protection/catalysis technique performs in such concentrations. Another problem with applying this technique is that there is no clear way of implementing it in Winfree's self-assembly system. Turberfield et al. (1999) have an idea of how to pursue research on this question, but no results have been presented. We believe that such results would be very worthwhile to pursue.

The protection/catalysis technique could be extremely useful for applications where speed of assembly is not so much an issue as fidelity of the process, and the formation of large structures with minimal amount of errors and misassemblies.<sup>15</sup> Unfortunately Turberfield et al. (1999) do not provide with references to previous work, where errors in the self-assembly were caused by interactions between small and larger aggregates, which they claim to be a motivation for their work. On the other hand such interactions should be expectable, and certainly there is no existing work that adequately studies this potential problem. We believe that it would be worthwhile to demonstrate a working example of self-assembly under this protection/catalysis technique, and possibly a control experiment of the same self-assembly without it. Research on this is in progress (Turberfield et al., 1999) and it would be very interesting to see experimental results.

## 5. Discussion

The models of DNA Computing we examined, have in various degrees, serious problems to overcome before they can perform non-trivial computations. Most likely these problems will in practice lead to constraints on the models. Such constraints would define the parameter space where computations under each method would be possible. For instance the whiplash PCR method may be constrained to single stranded molecules of a certain maximum length, in a maximum concentration, and where the different variables are encoded with a minimum base pairing disagreement for each pair of variables. Similarly the self-assembly method would probably face constraints on the maximum concentration, maximum aggregate size, maximum size of sticky ends, minimum disagreement of sticky ends, and many more. Algorithms devised for these methods would have to be of a highly parallel nature, and probably would be required to guarantee a certain amount of fault-tolerance. In general, we believe that experimentation would be the only way to further understand these issues, and theory would be most useful when needed in order to meet certain needs arising in practice and suggested by extensive experiments. Comparing the two main techniques we examined, namely whiplash PCR and self-assembly, we would like first of all to note that it is too early to decide whether either of them can deliver useful computation. Whiplash PCR would have to face problems that are very likely to be insurmountable, arising from the lower stability of single stranded DNA, and the many opportunities for unwanted ligations. Moreover the technique seems inherently much less parallel, and considerably slower than competing methods. Self-assembly has been demonstrated

---

<sup>15</sup> We believe that large-scale circuitry developed by a "bottom up" approach of molecular self-assembly might be such a case.

at least for the non-computing structures of Winfree et al. (1998). We believe that this method could benefit from a realistic test on a small instance of an NP-complete problem, in the near future.

DNA Computing in principle is not a solution to the intractability of NP-hard problems. Theoretically, the exponential time that the best conventional algorithms require to solve NP-hard problems is replaced in DNA Computing with exponential volume and exponential number of molecules. In practice however, the hope is that in the future the massive parallelism that DNA molecules offer, will enable solve certain problems, for instances of a specific size range, faster than conventional computers. Adleman (1994) for instance in his seminal paper says that in the future DNA computers might be able to run at a speed 1,000 times faster than supercomputers in 1994. Winfree (1998) gives a similar estimate for the number of operations per second possible in his method. Adleman (1995) gives also a more optimistic estimate, of a speedup of around  $10^6$  with respect to supercomputers in 1995.

That potential speedup might, or might not be enough by itself. The reason is that a molecular computer achieving these speeds is nowhere near in sight, and it is still debatable whether such a computer will ever be possible. On the other hand, conventional computers have been getting faster according to the quite popular Moore's law: *the size of a conventional chip doubles every 18 months, with an associated 2-fold increase in speed and decrease in cost*. This law has been holding since its inception 34 years ago. That would imply that the speed of computing is increased 1000-fold every 15 years, and similarly the cost decreases 1000-fold. This is true for the past 15 years, regarding data storage density, and it is slightly worse regarding processor speed. Therefore starting from 1995 and counting 30 years, according to Moore's law we would have in 2025 computers that are  $10^6$  faster with conventional architectures. It is hard to argue that DNA Computing has sufficient time until then, to deliver faster supercomputers, and use them efficiently for the benefit of science and engineering.

Unfortunately Moore's law cannot hold for much longer with conventional technologies. There are several articles in popular publications<sup>16</sup> that express this view, and point to the increasing search in industry for new competing technologies. Recent work provided experimental proof of the fundamental physical limit to the size of a gate oxide (Muller et al., 1999, Schulz, 1999), which is the smallest element of silicon devices. This work predicts that at the year 2012 this hard limit will have been reached. After that, further speedup of computers cannot come from conventional technological advances in silicon technology. Biological computation together with Quantum computation, are two popular avenues to speedier computers.<sup>17</sup> But another direction, perhaps more intuitive, is to find other technologies that can move the physical miniaturization limits of silicon devices further down (Schulz, 1999, Service, 1999a, b, c, d). Among the possible techniques promising to further miniturize computer components, are molecular self-assembly, or hybrid DNA/nanocrystal assembly techniques. Therefore research on DNA Computing may eventually achieve its ultimate goal of providing much faster computation, by means of helping build faster and smaller conventional computers. Beyond computing, there seem to be a number of applications, mainly in nanofabrication, that will benefit from DNA Computing and self-assembly techniques. For all these reasons, we strongly believe that work on DNA Computing will eventually prove worthwhile and applicable.

---

<sup>16</sup> See for instance The Economist, December 4<sup>th</sup>-10<sup>th</sup>, 1999, p. 79, Financial Times, Survey, December 1, 1999, p.1.

<sup>17</sup> Quantum computing actually offers theoretical advantages over conventional computing, unless factoring can be performed in polynomial time.

## References

- L.M. Adleman. 1994. Molecular Computation of Solutions to Combinatorial Problems. *Science*, 266:1021.
- L.M. Adleman. 1995. On Constructing a Molecular Computer. Manuscript, Department of Computer Science, University of Southern California.
- L.M. Adleman. 1996. On Constructing a Molecular Computer. In *DNA based computers*, p.1-22.
- Amos, M., Gibbons, A., and D. Hodgson. 1996a. Error-resistant implementation of DNA computations. *Proceedings of the 2<sup>nd</sup> DIMACS Workshop on DNA Based Computers*.
- Amos, M., Gibbons, A., and D. Hodgson. 1996b. A new model of DNA computation. *12<sup>th</sup> British Colloquium on Theoretical Computer Science*.
- Adleman, L.M., Rothmund, P.W.K., Roweis, S., and E. Winfree. 1996. On applying molecular computation to the data encryption standard. *Proceedings of the 2<sup>nd</sup> DIMACS Workshop on DNA Based Computers*.
- Alivasatos, A.P., Johnsson, K.P., Peng, X., Wilson, T.E., Loweth, C.J., Bruchez, M.P.Jr, and P.G. Schultz. 1996. Organization of 'nanocrystal molecules' using DNA. *Nature*, 382:609-611.
- Bach, E., Condon, A., Glaser, E., and C. Tanguay. 1996. *DNA Models and Algorithms for NP-complete problems*. IEEE Computer Society Press, p.290-299.
- E.B. Baum. 1996. DNA Sequences Useful for Computation. Manuscript, <http://www.neci.nj.nec.com/homepages/eric/seq.ps>.
- Eric Baum. 1996. Will future computers be made of DNA? *Windows Magazine*, June 1996, <http://www.neci.nj.nec.com/homepages/eric/window.ps>.
- Baum, E. and D. Boneh. 1996. Running dynamic programming algorithms on an DNA computer. *Proceedings of the 2<sup>nd</sup> annual DIMACS conference on DNA computing*.
- Bohm, C. and G. Paun. 1997. Cooperating distributed splicing systems. Hand-out workshop on *Molecular Computing*.
- Braun, E., Eichen, Y., Sivan, U., and G. Ben-Yoseph. 1998. DNA-templated assembly and electrode attachment of a conducting silver wire. *Nature*, 391:775-777.
- Boneh, D., Dunworth, C., Lipton, R., Sgall, J. 1996. On the Computational Power of DNA. *Discrete Applied Mathematics*, 71(1):79-94.
- Boneh, D. and R. Lipton. 1995. Making DNA computers error resistant. Princeton University TR-491-95.
- Boneh, D. and R. Lipton. 1995. Making DNA computers error resistant. *Proceedings of the 2<sup>nd</sup> DIMACS Workshop on DNA Based Computers*.
- V. Chvatal. 1985. Hamiltonian cycles. In Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., Shmoys, D.B., (Eds.), *The traveling salesman problem -- A guided tour of combinatorial optimization*, Wiley & Sons, Chichester, p.403-429.

- Coffer, J.L., Bigam, S.R., and X. Li. 1996. Dictation of the shape of mesoscale semiconductor nanoparticle assemblies by plasmid DNA. *Applied Physics Letters*, 69(25):3851-3853.
- Collier, C.P., Wong, E.W., Belohradsky, M., Raymo, F.M., Stoddart, J.F., Kuekes, P.J., Williams, R.S., and J.R. Heath. 1999. Electronically Configurable Molecular-Based Logic Gates. *Science*, 285:391-394.
- Culik, K. and T. Harju. 1989. Dominoes and the regularity of DNA splicing languages. *Proceedings of the 16th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science*, 372:222—233.
- Davis, A.P. 1999. Synthetic molecular motors. *Nature*, 401:120-121.
- Deaton, R., Garzon, M., Murphy, R.C., Rose, J.A., Franceschetti, D.R., and S.E. Stevens, Jr. 1998. Reliability and Efficiency of a DNA-Based Computation. *Physical Review Letters*, 80(2):417-420.
- T.L. Eng. 1997. Linear DNA self-assembly with hairpins generates the equivalent of linear context-free grammars. *Proceedings of the 3rd DIMACS Workshop on DNA Based Computers*, p.296--301.
- Ferretti, C. and S. Kobayashi. 1996. DNA splicing systems and Post systems. Hunter, L. and T. Klein, editors. *Biocomputing: Proceedings of the 1996 Pacific Symposium*. World Scientific Publishing Co.
- Freund, R., Csuhaj-Varjú, E., and F. Wachtler. 1997. Test tube systems with cutting/recombination operations. *Pacific Symposium on Biocomputing*.
- Freund, R., Kari, L., and G. Paun. 1999. DNA computation based on splicing: The existence of universal computers. *Theory of Computing Systems*, 32:69—112.
- Fu, T.J. and N.C. Seeman. 1993. DNA double-crossover molecules. *Biochemistry*, 32:3211-3220.
- D.K. Gifford. 1994. On the Path to Computation with DNA. *Science*, 266:993-994.
- R.J. Gould, 1991. Updating the Hamiltonian problem. *Graph Theory*, 15:121-157.
- Hagiya, M., Arita, M., Kiga, D., Sakamoto, K., and S. Yokoyama. 1999. Towards Parallel Evaluation and Learning of Boolean  $\mu$ -formulas with Molecules. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 48:57-72.
- T. Head. 1987. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. *Bulletin of Mathematical Biology*, 49(6):737-759.
- T. Head. 1992. Splicing systems and DNA. *Handbook of Formal Languages*. Springer Verlag, Berlin, Heidelberg, New York, p.371-383.
- Head, T., Yamamura, M., and S. Gal. 1999. Aqueous computing: writing on molecules. *Proceedings of the Congress on Evolutionary Computing (CEC'99)*.
- Kelly, T.R., De Silva, H., and R.A. Silva. 1999. Unidirectional rotary motion in a molecular system. *Nature*, 401:150-152.
- Koumura, N., Zijlstra, R.W.J., van Delden, R.A., Harada, N., and B.L. Feringa. 1999. Light-driven monodirectional molecular rotor. *Nature*, 401:152-155.
- LaBean, T.H., Winfree, E., and J.H. Reif. 1999. Experimental Progress in Computation by Self-Assembly of DNA Tilings. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 48:121-138.
- R.J. Lipton. 1994. Speeding up computations via molecular biology. Unpublished manuscript.

- R.J. Lipton. 1995. DNA solutions of hard computational problems. *Science*, 268:542-545.
- S. Mann. 1993. *Nature*, 365:499.
- Mao, C., Sun, W., and N.C. Seeman. 1997. Assembly of Borromean rings from DNA. *Nature*, 386:137-138.
- Mao, C., Sun, W., Shen, Z., and N.C. Seeman. 1999. A nanomechanical device based on the B-Z transition of DNA. *Nature*, 397:144-146.
- Mirkin, C.A., Letsinger, R.L., Micic, R., and J.J. Storhoff. 1996. A DNA-based method for rationally assembling nanoparticles into macroscopic materials. *Nature*, 382:607-608.
- Muller, D.A., Sorsch, T., Moccio, S., Baumann, F.H., Evans-Lutterodt, K., and G. Timp. 1999. The electronic structure at the atomic scale of ultrathin gate oxides. *Nature*, 399:758-761.
- Ouyang, Q., Kaplan, P.D., Liu, S., and A. Libchaber. 1997. DNA Solution of the Maximal Clique Problem. *Science*, 278:446-449.
- C. Papadimitriou. 1994. *Computational Complexity*. Addison Wesley.
- Pitt, L., and G.L. Valiant. 1988. Computational limitations on learning from examples. *J. Assoc. Comput. Mach.*, 35(4):965-984.
- Reif, J.J. 1995. Parallel molecular computation: Models and simulations. *Proceedings of the Seventh Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA95)*, p.213-223.
- Reif, J. 1998. Paradigms for biomolecular computation. *First International Conference on Unconventional Models of Computation*. Published in *Unconventional Models of Computation*, C.S. Calude, J. Casti, and M.J. Dinneen, Springer Publishers, p.72-93.
- Reif, J.J. 1998. Parallel molecular computation: Models and simulations. *Algorithmica*, (Special issue on Computational Biology).
- Roweis, S., Winfree, E., Burgoyne, R., Chelyapov, N.V., Goodman, M.F., Rothmund, P.W.K., and L.M. Adleman. A Sticker Based Model of DNA Computation. 1996. *Proceedings of the 2<sup>nd</sup> DIMACS Workshop on DNA Based Computers*, p.1-27.
- Schulz, M. 1999. The end of the road to silicon? *Nature*, 399:729-730.
- Service, R.F. 1999b. Borrowing From Biology to Power the Petite. *Science*, 283:27-28.
- Service, R.F. 1999a. Finding Speed on the Smallest Scales. *Science*, 283:165-167.
- Service, R.F. 1999c. New Memory Cell Could Boost Computer Speeds. *Science*, 284:1444.
- Service, R.F. 1999d. Organic Molecule Rewires Chip Design. *Science*, 285:313-315.
- Turberfield, A.J., Yurke, B., and A.P. Mills, Jr. 1999. DNA Hybridization Catalysts and Molecular Tweezers. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 48:169-181.
- B. Vandegriend. 1998. Finding Hamiltonian cycles: algorithms, graphs and performance, MSc Thesis, University of Alberta, Canada.

- Whitesides, G.M., Mathias, J.P., and C.T. Seto. 1991. Molecular self-assembly and nanochemistry: a chemical strategy for the synthesis of nanostructures. *Science*, 254:1312-1319.
- E. Winfree. 1995. On the Computational Power of DNA Annealing and Ligation. In Baum and Lipton, *DNA Based Computers*, volume 27 of DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.
- E. Winfree. 1998a. Simulations of Computing by Self-Assembly. Fourth International Meeting on DNA Based Computers, p. 213-239.
- E. Winfree. 1998b. Whiplash PCR for  $o(1)$  computing. In Preliminary Proceedings 4th DIMACS Workshop on DNA Based Computers, p.175-188.
- Winfree, E., Liu, F.I., Wenzler, L.A., and N.C. Seeman. 1998. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394:539-544.
- Winfree, E., Yang, X., and N. C. Seeman. 1996. Universal computation via self-assembly of DNA: Some theory and experiments. Proceedings of the 2<sup>nd</sup> DIMACS Workshop on DNA Based Computers
- Wilhelm, P., and K. Rothmund. 1996. In Baum & Lipton, *DNA Based Computers*, volume 27 of DIMACS: Series in Discrete Mathematics and Theoretical Computer Science.
- T. Yokomori. YAC: Yet another computation model of self-assembly. In *DNA Based Computers V*, 1999