AQUEOUS COMPUTING: Writing on Molecules

Tom Head and Masayuki Yamamura
Department of Mathematical Sciences
Binghamton University
Binghamton, New York 13902-6000
tom@math.binghamton.edu and my@dis.titech.ac.jp

Susannah Gal
Department of Biological Sciences
Binghamton University
Binghamton, New York 13902-6000
sgal@binghamton.edu

Abstract- Molecular computing is viewed here as a process of writing on molecules while they are dissolved in water. When DNA molecules are employed, they are used only in double stranded form and only as data registers. All computations are initialized with the same single molecular variety. Current progress toward laboratory prototyping of computations is reported.

1 Introduction

A vision for the future of molecular computing is sketched here. It may be regarded as a more conventional approach to molecular computing than other approaches currently in progress: There is an identifiable computer. The computer, in its initial state, consists of a volume of water in which a vast number of identical molecules are dissolved. The molecular type selected for this initialization is chosen to contain specific distinguishable locations that can function as bistable devices, thus representing bits 0 or 1. We will call these locations *stations*. Every computation begins with each of these molecules having all of its stations viewed as representing the same bit. As a computation progresses selected bit settings are altered. Conceptually, each of these molecules is a data register that displays the same fixed number of bits. The collection of all these data register molecules constitutes the memory of the computer. The role of the water is fundamental: (1) it separates the memory molecules from one another to allow access to each one; (2) it randomizes the location of the molecules by diffusion; (3) it allows partition of the memory into any chosen number of (approximately indistinguishable) parts so that different actions can be taken in each part; and (4) it allows the rejoining of the parts. Note that aqueous computing realizes content addressable parallel processing as described in [F76,Chap.1].

There are many reasonable ways that the bits of the data register molecules might be altered. If the memory molecules are circular double stranded DNA then they can be altered at restriction enzyme sites by cut & paste operations using restriction enzymes and a ligase. If the memory molecules are proteins then they can have antibodies, perhaps with phosphorescent labels, attached. With complexes of nucleic acids & polypeptides, electromagnetic radiations of various frequencies might be used to alter the conformations of the polypeptides. Here we report our pursuit of solutions to a collection of NP-complete algorithmic problems using a cut & paste technique as our method of writing on DNA. If aqueous computing is to become practical we expect it be implemented not in test tubes, but perhaps in flowing capillary systems as exemplified by the PCR system of [KMM98].

The proposals made here arose in the current of DNA computing studies generated by the electrifying work of L.Adleman [A94]. Excellent introductory papers on DNA computing are [A98], [Ka97] & [Am99]. The book [PRS98] is highly recommended. The collection of papers [P98] will be of interest in any extensive study of biomolecular computing. Researchers have found the paper [L95] by R.Lipton to be highly stimulating and provocative. The note by J.Hartmanis [Ha95] sounds an important warning. The laboratory work that is reported here was stimulated by [OKLL97] with the details stemming from [H87] & [HPP97]. Ideas presented appear in [H98] in a more relaxed form, but with less generality and unification. For help with the biochemistry see the first chapter of [PRS98], the initial sections of [Ka97], or [HPP97]. See [M95] as a thorough backup.

2 A Single Descriptive Format for Several Classical NP-Complete Problems

Many NP-complete algorithmic problem families are readily assimilated to a common pattern which allows a solution by a single aqueous algorithm. Only three will be treated carefully here. Some others are mentioned in [He98].

(1) Find the largest cardinal number for which the undirected graph $G = (V, A)$ has an independent subset of that cardinal [GJ79,p.194]. In this notation V is the finite set of vertices of G and A is the family of unordered pairs of vertices that constitute the arcs (= unordered edges) of G.

(2) Find the smallest cardinal number for which the undirected graph $G = (V, A)$ has a vertex cover of that cardinal [GJ79,p.190].

(3) For a given finite set P of Boolean variables and a finite set U of clauses over P, does a truth setting for the variables in P exist for which each of the clauses in U is satisfied? [GJ79,p.38].

All these problems can be viewed as special cases of the following algorithmic problem:

The Common Algorithmic Problem (CAP). Let S be a finite set. Let F be a family of subsets of S. Find the largest cardinal number for which there is a subset T of S which fails to contain any of the sets belonging to F. We say that the sets

in the family F are *forbidden* sets.

Problem (1) above is obtained by letting $S = V$ and $F = A$. The cardinal number desired in (1) is the cardinal number that is the solution of the CAP.

Problem (2) above is obtained by letting $S = V$ and letting F consist of the neighborhoods of the vertices of G, where the neighborhood of the vertex v in V is $\{u \text{ in } V : u = v \text{ or } \{u, v\} \text{ in } A\}$. The cardinal number desired in (2) is produced by subtracting the solution of the CAP from the cardinal number of V.

Problem (3) above is obtained as follows: For each variable p in P, let p' be a new symbol. Let $P' = \{p' : p \text{ in } P\}$. Let S be the union of P and P'. Construct a subset C' of S from each clause C in U as follows: If a symbol p appears in C, place p' in C'; and if the negation of p appears in C, place p in C'. Construct F as follows: For each p in P, place $\{p, p'\}$ in F; and for each C in U, place C' in F. Now let n be the cardinal number of P. Notice that the solution to the CAP is at most n. From this it follows that the solution to Problem (3) is YES if the solution of the CAP is n and NO otherwise.

3 An Aqueous Algorithm for Solving the Common Algorithmic Problem

Let S be a finite set and let F be a family of forbidden subsets of S. In order to discuss the proposed algorithm with minimal dependence on the choice of molecular implementations, the biochemical details of the laboratory work now in progress will be deferred to Section 4. The algorithm is given immediately for inspection - with explanation following.

```
Algorithm.     Initialize;
  For   each {s1, s2, ... ,sk} in F   Do
       Pour (k)
         1: SetToZero( s1 )
         2: SetToZero( s2 )
            ...
         k: SetToZero( sk )
       Unite
  EndFor;
  MaxCountOfOnes.
```

The validity of the algorithm follows immediately once the actions Initialize, Pour–Unite, and MaxCountOfOnes have been explained.

Let n be the cardinal number of S. The molecule to be used as the data register molecule must have at least n stations. The elements of S are then placed in one-one correspondence with a subset of the set of stations of the data register molecule. We Initialize to a tube of water containing many identical data register molecule. We choose to regard each station of each molecule as representing the bit 1. For- -Do- -EndFor has its usual meaning. Pour (k) requires that the contents of a single tube be poured into k, $(k < n)$, separate tubes with equal amounts in each of the k tubes. In parallel (or in any order): for each i $(1 < i < k)$, for every data register molecule in tube i, set the bit at the station corresponding to the element s_i of S to 0. Unite the k tubes by pouring the contents of each into a single common tube. On arrival at MaxCountOfOnes we have a single tube. This command requires the determination of the largest cardinal number that appears as the number of 1's that remain at the n stations (i.e., those n that were set in one-one correspondence with S) of any of the data register molecules.

This algorithm is a variation of the algorithm used in [OKLL97]. If our algorithm is applied to the problem treated in [OKLL97] then the forbidden sets are $\{2, 0\}$ $\{0, 5\}$ $\{5, 1\}$ & $\{1, 3\}$. The test tube is initialized (as always) to contain only one molecular variety. The numbers of distinct molecular varieties in the test tube at the completion of each pass through the For loop are: 2, 4, 7, 12. One of these 12 provides the solution. The procedure of [OKLL97] may be formulated using a similar For loop. There the test tube is initialized to contain 64 molecular varieties. The numbers of distinct molecular varieties in the test tube at the completion of each pass through that For loop are: 48, 40, 32, 26. One of these 26 provides the solution.

4 A Proposed Laboratory Procedure Using the Cut & Paste Technology

In order to provide a convenient proof of concept we use as our data register molecule one of the standard double stranded DNA cloning plasmids commercially available. This plasmid is a circular molecule of approximately three kilobases. It contains a subsegment, MCS (multiple cloning site), of approximately 175 base pairs that can be removed using a pair of restriction enzyme sites that flank the segment. The MCS contains eight pairwise disjoint sites at which restriction enzymes act such that each produces a 5' overhang of four bases. These eight sites serve as the stations of our data register molecules. If we obtain laboratory evidence that suggests that problems requiring a larger number of stations might be solved using DNA plasmids then an appropriate plasmid having many more stations can be constructed.

The initial condition of a station (= restriction enzyme site) is chosen to represent the bit one. A zero is written at a station by altering the site using the following three step process: (1) linearize the plasmid by cutting it at the station (=site) with the restriction enzyme associated with the site; (2) using a DNA polymerase, extend the 3' ends of the strands lying under the 5' overhangs to produce a linear molecule having blunt ends; and (3) in dilute solution apply a ligase to recircularize the linear molecule by ligation of the blunt ends. When a station is altered to represent the bit zero the length of the station is increased by four base pairs and the station no longer encodes a site for the originally associated restriction enzyme.

At the initiation of a computation all eight sites are present in each of the data register molecules. Thus initially each

molecule is read as: 11111111. Note that the first time a zero is written at a station the molecule increases in circumference by four base pairs, after which no further alteration of that station can occur during a computation. If we suppose that zeros have been written at the second, fourth, & fifth stations of a molecule then the circumference of the molecule will have been increased by twelve base pairs and the molecule will be read as: 10100111. This completes the biochemical realization of Initialization and SetToZero appearing in the Algorithm of Section 3. MaxCountOfOnes can be realized by applying the following three step process: (1) Using the restriction enzyme having sites at each end of the MCS (multiple cloning site), cut the plasmids into the roughly 175 base pair and the nearly 3 kilobase long linear molecules; (2) separate the short strands of length roughly 175 base pairs on an acrylamide gel in one lane with a calibrating DNA ladder in a second lane; and (3) from the length in base pairs of the molecules in the band on the gel that consists of the molecules that have the least length, calculate the number of restriction enzyme sites that remain. This is the value returned by MaxCountOfOnes. For example, if the length in base pairs of the molecules of least length is 187 then MaxCountOfOnes is $8 - (1/4)(187 - 175) = 5$, which is the number of ones in 10100111.

If desired, the restriction sites that remain in the molecules that provide the maximum number of ones can be obtained in parallel by applying the eight restriction enzymes in eight separate tubes. If the solution is not unique this will need to be done after a cloning process. The status of the stations can also be determined by DNA sequencing.

5 Progress Report from the Laboratory

The critical challenge for the computational procedure sketched in Section 4 is expected to be the three step cut & paste process by which a station is set to zero. More precisely, the difficulty will lie in carrying out each cut & paste operation with such precision that, after several such operations have been carried out in series, the answer can be read definitively at the final gel separation step. We report here the results of carrying out three successive cut & paste operations.

We used a pBluescript plasmid (Strategene, La Jolla, CA) as our data register molecule. The cut operations were made using the three restriction enzymes *Hind*III, *Bam*HI, and *Xba*I (New England Biolabs, Beverly, MA) in the order given. pBluescript has a unique site for each of these three enzymes and these three sites lie in the MCS (multiple cloning site). In all three cases the four base extensions were made with Klenow DNA polymerase (Stratagene, La Jolla, CA) and the blunt end ligations were done with T4 DNA ligase (Stratagene, La Jolla, CA).

In this paragraph, we explain the molecular basis for the plan of our testing procedure and the expected results. pBluescript includes a gene for the protein β-galactosidase encoded in part in the MCS. An Ecoli bacterial clone that contains a pBluescript plasmid is blue in the presence of a substrate for the β-galactosidase, X-Gal (5-bromo-4-chloro-3-indolyl-β-D-galactopyranoside). When the β-galactosidase gene of pBluescript has been disabled, the clones are white. The translation system between DNA and protein occurs in triplets, three bases of DNA yielding one amino acid of the protein. In step (1), the cut, fill-in & paste operation should successfully add four bases in the *Hind*III site of pBluescript with the result that the reading frame will be off by one base pair for the β-galactosidase gene. Thus this plasmid should produce an inactive β-galactosidase and 100% white clones. The plasmid DNA from one of the white colonies can be used for the next step. In step (2), the second cut, fill-in & paste operation in the *Bam*HI site should successfully add four more bases making the reading frame off by two base pairs this time. The plasmid should again produce 100% white clones. The plasmid DNA from one of these white clones can be used for the next step. In step (3), the third cut, fill-in & paste operation in the *Xba*I site should restore the proper reading frame for the β-galactosidase gene of pBluescript. The total extension of twelve base pairs in the gene for the β-galactosidase should result in the insertion of four extra amino acids and some alterations that produce a slightly different, but likely active protein. Thus we expect 100% blue clones from the DNA after this step. If the results in (1), (2) & (3) were each as expected, this would assure us that the sequences of cut, fill-in & paste operations, required in computing according to the Algorithm of Section 3, can be successfully carried out. The final gel separation that concludes the Algorithm is not expected to present an obstacle since it is a standard procedure of molecular biology.

In this paragraph we report the actual test results we have obtained at the time of this writing, March 26, 1999, at each of the three steps described in the previous paragraph: At step (1) we obtained 87% white clones, as computed from 40 white and 6 blue. The plasmid DNA from three of the 40 white colonies was isolated and tested for the presence of a *Hind*III site; as expected, none of the plasmids contained the site. At step (2) we obtained 96.4% white clones, as computed from 80 white and 3 blue. Plasmid DNA from four of the 80 white colonies was prepared and tested for the continued presence of a *Hind*III site or a *Bam*HI site; as expected, none of the DNAs contained either of these sites. At step (3) we obtained 85% blue clones, as computed from 97 blue and 17 white. DNA plasmids from four of the 97 blue colonies was prepared and tested for the presence of a *Hind*III site, a *Bam*HI site, or an *Xba*I; as expected, none of the DNAs contained any of these three sites. To confirm the correct filling in reactions at each site, we sequenced one of the plasmid DNAs obtained from white clones at steps (1) and (2) and two of the plasmid DNAs obtained from blue clones at step (3). These sequences are found in Figure 1 and confirm the completion of the cut, fill-in & paste operations. These sequences also confirm the restoration of the β-galactosidase reading frame

in the DNA from the two blue colonies from step (3).

See Figure 1

In none of the three steps were the results 100% as expected. How serious are these divergences? At step (1) the 87%, with near certainty, presents no problem. The pBluescript molecules, but not the lengthened molecules, are expected to be supercoiled. This allows the (comparatively few?) remaining unmodified molecules to enter bacteria with much higher probability than the modified molecules. Consequently the slightest trace of remaining unmodified molecules (which would produce blue colonies) might have resulted in this observed deviation from 100% white colonies. At step (2) the 3.6% blue clones gives us more concern since it indicates an unexpected restoration of the reading frame of β-galactosidase. Plasmid DNA from two of the three blue colonies was prepared and tested for the presence of either a *Hind*III site or a *Bam*HI site; neither restriction enzyme site was present. This supplementary result is comforting, but these plasmids should be examined further to determine how they have produced blue clones after only the *Hind*III and *Bam*HI sites were destroyed. At step (3) the 85% figure gives us serious concern since it suggests inefficiency at either cut or fill-in steps.

What we have reported here are our first test results. As such, we consider them encouraging. With more laboratory experience in performing our three step cut & paste operation, it seems reasonable that we may improve the efficiency. We think this will be possible, especially once the mysterious appearance of blue colonies at step (2) and the continued appearance of so many white colonies at step (3) is understood. It may be necessary to replace *Xba*I with a different restriction enzyme at step (3). We hope that, by the time of the oral presentation of this paper, we will report laboratory solutions for instances of computational problems of the sort described in Section 2 using the Algorithm of Section 3.

Acknowledgments

Partial support for this research through NSF CCR-9509831 and DARPA/NSF CCR-9725021 is gratefully acknowledged. Support for the second author through the Japan Society for the Promotion of Science under the Research for the Future Program (JSPS-RFTF 96100101) is acknowledged with thanks. The first author thanks the Leiden Center for Natural Computing (LCNC) at Leiden University, the Netherlands, for support and for providing stimulating contacts during the summer of 1998 when some of the ideas presented here were developed. Closely related developments are also in progress at the LCNC and the first author is very grateful to Professors Grzegorz Rozenberg and Herman Spaink of Leiden University for their collaboration, encouragement, and advice. All three authors thank Xia Chen for excellent technical support.

M.Yamamura visited Binghamton University during the academic year 1998-1999. His permanent address is: Professor M. Yamamura, Department of Computational Intelligence and Systems Science, Tokyo Institute of Technology, 4259 Nagatsuta, Yokohama, 226 JAPAN.

Bibliography

[A98] L.Adleman, “Computing with DNA,” Scientific American, 297 No.2(August, 1998)54-61.

[Am99] M.Amos, “Theoretical and experimental DNA computation,” Bull. European Assoc. for Theor. Computer Sci., 67(1999)125-138.

[F76] C.C.Foster, “Content Addressable Parallel Processors,” Van Nostrand Reinhold, New York, (1976).

[GJ79] M.R.Garey & D.S.Johnson, “Computers and Intractability - A Guide to the Theory of NP-Completeness,” W.H.Freeman, San Francisco, CA, (1979).

[Ha95] J.Hartmanis, “On the weight of a computation,” Bull. European Assoc. for Theor. Computer Sci., 55(1995)136-138.

[He87] T.Head, “Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors,” Bulletin of Mathematical Biology, 49(1987)737-759.

[He98] T.Head, “Circular suggestions for DNA computing,” (submitted).

[HePP97] T.Head, Gh.Paun & D.Pixton, “Language theory and molecular genetics: generative mechanisms suggested by DNA recombination,” in: Handbook of Formal Languages, G.Rozenberg & A.Salomaa, EDs., Springer Verlag, (1997).

[Ka97] L.Kari, “DNA computing: arrival of biological mathematics,” The Mathematical Intelligencer, 19 No.2(Spring 1997)9-22.

[KMM98] M.U.Kopp, A.J.de Mello, & A.Manz, “Chemical amplification: continuous-flow PCR on a chip,” Science, 280(1998)1046-1047.

[L95] R.Lipton, “DNA solution of hard computational problems,” Science, 268(1995)542-545.

[M95] R.A.Meyers, Ed., “Molecular Biology and Biotechnology - A Comprehensive Desk Reference,” VCH Publishers Inc., New York (1995).

[OKLL97] Q.Ouyang, P.D.Kaplan, S.Liu & A.Libchaber, “DNA solution of the maximal clique problem,” Science, 278(1997)446- 449.

[P98] Gh.Paun, Ed., “Biomolecular Computing. Theory and Experiment,” Springer Verlag, (1998).

[PRS98] Gh.Paun, G.Rozenberg & A.Salomaa, “DNA Computing. New Computing Paradigms,” Springer Verlag, (1998).

```
                        XbaI       BamHI                          Hind3
 pBS   DNA  GTGGCGGCCGCTCTAGAACTAGTGGATCCCCCGGGCTGCAGGAATTCGATATCAAGCTTATCGATACCGTCGACCTCGAGG
[H]    DNA  GTGGCGGCCGCTCTAGAACTAGTGGATCCCCCGGGCTGCAGGAATTCGATATCAAGCTagctTATCGATACCGTCGACCTC
[HB]   DNA  GTGGCGGCCGCTCTAGAACTAGTGGATCgatcCCCCGGGCTGCAGGAATTCGATATCAAGCTagctTATCGATACCGTCGA
[HBX]  DNA  GTGGCGGCCGCTCTAGctagAACTAGTGGATCgatcCCCCGGGCTGCAGGAATTCGATATCAAGCTagctTATCGATACCG

 pBS   prot  V  A  A  A  L  E  L  V  D  P  P  G  C  R  N  S  I  S  S  L  S  I  P  S  T  S  R
[H]    prot  V  A  A  A  L  E  L  V  D  P  P  G  C  R  N  S  I  S  S stop
[HB]   prot  V  A  A  A  L  E  L  V  D  R  S  P  G  L  O  E  F  D  I  K  L  A  Y  R  Y  R  R
[HBX]  prot  V  A  A  A  L  A  R  T  S  G  S  I  P  R  A  A  G  I  R  Y  O  A  S  L  S  I  P
```

Figure 1. The DNA and protein sequences for the pertinent segments of the MCS region of pBluescript and the altered plasmids produced from the computation

The pBluescript (pBS), filled in HindIII ([H]), filled in BamHI and HindIII ([HB]), and filled in XbaI, BamHI and HindIII ([HBX]) plasmid sequences (DNA) and the protein sequences deduced from the DNA sequences (prot) are shown. The new nucleotides that were introduced in [H] [HB] and [HBX] are printed in lower case.