An opinionated, but reasonably short, summary of the Mini DIMACS Workshop on DNA based computers, held at Princeton University on April 4 1995

Warren D. Smith*
wds@research.nj.nec.com

April 5, 1995

Len Adleman began by reviewing his article [Science 266 (Nov 1994) 1021-1024] which started it all, and expounding on his views of the field. The fact that his invention (and Lipton's) are not universal computers does not bother him so much because

1. computing is about particular problems, and
2. the problems he and Lipton selected are very nicely adapted to DNA computation.

Both Adleman's and Lipton's schemes in some sense will solve any problem in NP, and it isn't clear that the capability of universal computers to solve harder problems than those in NP, is actually very accessible in real life (although I myself run exponential algorithms all the time...). Lipton hammered on the same theme arguing that "take home message: constant factors matter." More on that later.

Generally Adleman and Lipton want to keep the tone optimistic and compare critics who carp about the presently limited capabilities of DNA computing schemes to somebody who would have criticized early electronic computer plans ("ha! vacuum tubes are too unreliable and power hungry, you'll never get anywhere").

Richard Lipton described how to solve SAT problems (i.e. any problem in NP...) via a very nice scheme described in his web-available manuscript (`/ftp/pub/people/rjl/bio.ps` on `ftp.cs.princeton.edu`). Lipton introduced a few abstract operations to model what was accomplished with certain DNA processes, and allowing a clean mathematical formulation of questions such as "what can we accomplish using N of these abstract operations?"

Eric Baum pointed out that 1000 liters (= 1m^3... a California style hottub) of soup could contain about 10^{20} DNA molecules, each encoding a memory of reasonable length. That is a lot more memory than all computer memories ever made, combined, also a lot larger than the memory capacity (between 10^6 and 10^{15} memories) of the human brain. In principle the memories that most closely match antisense beads could be extracted and sequenced. Similarly to human memories, retrieval would be "associative" and "fuzzy." Also the whole memory might be clonable. Baum has an NECI tech report describing this and a patent pending. Actually I think Baum's proposal wouldn't work well since the retrieval times would be very large, how would you synthesize that many memories in the first place (would it take 10^{20} hours? Age of universe is order 10^{17} seconds...) et cetera.

Dan Boneh, a student of Lipton's, described how to break the DES cryptosystem (find a key that would generate a particular plaintext ciphertext pair) in 907 "bio-steps." He estimated that these steps could be carried out in about 4 months. Meanwhile a conventional computer using brute force and doing 10^5 DES transformations per second would take 10^4 years. (It's true that special purpose hyperparallel DES-breaking machines have been conceived which would break DES in about a year and would cost about \$$10^6$, and algorithms a little better than brute force have also been conceived, but this doesn't alter the picture a great deal.) But there is skepticism about some of Boneh's tricks that reduce the step count by doing, e.g. several ORs or several ANDs of various things simultaneously. I think they would foul up. Different DNA melting temperatures would be required and can't be supplied, mispairing would happen, etc. Still, I don't think such criticisms really affect the validity of Boneh's ideas, they would just force him to abandon some of his "accounting tricks" (reminiscent of congressmen calculating the national debt in creative ways) and increase his step count to, say, 3000 bio-steps. Then it would take a year to break DES, not 4 months.

Along the way, Boneh described some cute operations with DNA, e.g. "tagging" and "table lookup" primitives, which ought to have other uses.

*NECI, Princeton NJ

Warren Smith and **Allan Schweitzer** (talk by Smith) showed how to create a nondeterministic Turing machine with DNA and standard lab techniques, and then presented a loose inspirational collection of biological phenomena that look computational, attempting to show that C.S. thinking applied to biological problems can yield dividends in biology. Their paper is an NECI technical report (NEC Research Institute, 4 Independence Way Princeton NJ 08540 USA) and was handed out at the workshop.

Although I'd characterize all speakers as more or less optimistic, Smith was the exception... definitely the least optimistic and most pessimistic of all the speakers about the near future of DNA based computing.

Also, Smith was the only speaker to actually present NUMBERS about the severity of various undesirable side processes and also about the times various "bio-steps" would take. Undesirable processes include:

- DNA hydrolyzes in water, and the situation is especially severe and unrepairable with single strand DNA. All schemes (including Smith's) presented at the workshop depended on the use of single strand DNA, with the exception of Rothemund's.
- Restriction enzymes occasionally cut at noncognate sites... the schemes by Smith, Rothemund, and some Liptonian schemes depend on them.
- Various kinds of DNA damage (depurinations and oxidative hits)
- Gel electrophoresis is basically not understood.
- Replication errors occur.
- Mismatches in DNA annealing occur... annealing occurs slowly at low concentrations... Smith's paper has data and formulas on this. Example: it seems to me that in Lipton's SAT scheme, it could happen at some point that the next logical operation you want to perform will be satisfied by only, saym 1 in 10^{13} of the molecules in the soup. Due to the low concentrations, it will take a very long time for the annealing (antisense beads) he needs to run to completion at this step. It will not suffice to not run it to completion since then DNA types that might be needed in future logical steps, would be missed.
- Fluid shear destroys DNA, especially DNA longer than 15000 base pairs.
- Biological and nuclease contamination may not be totally avoidable...

Smith basically concludes that his scheme, and Adleman's, and Lipton's, and Baum's, would *not* be practical, and while they will work (or ought to) for small toy problems such as the one in Adleman's Science paper, consisting of 5-100 biosteps, they will succumb to these bad effects when one tries to scale up to large problems.

I'm also very skeptical of bandied schemes that want to use order 1 Kg of DNA. Either the assumption that a human being contains 10^{13} cells, or the assumption that DNA is half the weight of cell nuclei and using the well known estimate that nuclei are 1/1000 of eukaryote cell volumes, leads to the conclusion that a human contains about 30 grams of DNA. Biologists use less than 30 micrograms of DNA in essentially every standard lab procedure. Many of the schemes proposed involve surface effects (antisense beads) rather than volume effects. I finally point out that DNA enzymes such as T4 ligase presently cost approximately 10^5 times as much as gold. You do *not* want to cheerily wave your hands about kilogram quantities and bathtubs, OK?

Here is a nonquantitative argument about why all in vitro DNA schemes are doomed to failure: All chemical reactions suffer from parasitic side reactions, sub-100% yields, etc. These things will accumulate exponentially. All this adds up to a very big problem to try to overcome. The only reason we are still alive is that lifeforms are continually fighting to repair the damage. Every human cell every day repairs tens of thousands of DNA damage events. Perfectly. (Well, almost. Order 1 mutation per year per human cell leaks through the system.) It is amazing – but despite all that, we still die in the end. The whole purpose of lifeforms is simply to repair and protect DNA. In vitro, our current DNA error rates, yields, protection from contamination and fluid shear, packaging, structural assembly, and general chemical capabilities (e.g. making enzymes to do what we want, is something we cannot do at present) are far worse than those of lifeforms. They will remain that way in the forseeable future. So I claim that the only way you are going to make a DNA based computer that keeps on ticking for the ≥ 1000 biosteps required to do interesting calculations (e.g. Boneh's) that lie beyond the capacities of electronic computers, is if you somehow duplicate all the protective, supportive, and repair capabilities of life itself. And at that point, what you would have created in vitro, would in fact *be* life of comparable capability to real life.

Adleman suggested that perhaps artificial polymers such as PNA [see Science (Dec 1991) 1497-1500] might be less susceptible to damage, mispairing, etc than DNA. Unfortunately, there are no enzymes for manipulating PNA... such as ligase (which Adleman needs)... and while PNA is certainly superior than DNA in some ways for some purposes (e.g. stronger binding) I'm dubious anybody is going to come up with anything better than DNA in sum total.

Smith also pointed out the previously conveniently ignored facts that conventional computers rigorously solve 2392 city real life TSP problems and 300-bit "hard" 3-SAT problems. Meanwhile Adleman and Lipton's DNA schemes can't come close. Moral: good branch and bound algorithms can buy you a lot more than some constant factor of 10^{12} from DNA. Smith and Schweitzer show that their DNA Turing machine ought to be able to duplicate the asymptotic exponential speedups from good branch and bound algorithms, but they suffer by a large constant factor since they have to use a 1-tape Turing machine, a horribly innefficient computer. So, despite all the hype, I remain unconvinced that there exists *any* practically important problem that DNA computers can solve that conventional computers cannot. That is an open problem.

Adleman counter-argued that he and Richard Karp suspected that the hypothetical *worst* 70-bit SAT problem *would* require 2^{70} steps and would *not* succumb to the usual branch and bound algorithms, and hence would be untouchable by conventional computers but perhaps reachable by Lipton's scheme. My response was that (A) what matters in arguments about practical impact is not hypothetical worst problems but real world ones, and (B) if Adleman and Karp think they are so smart, then all they have to do is write a problem generator that generates hard n-bit SAT problems, crank it up with $n = 70$, run the DIMACS SAT challenge SAT programs on them, and see what happens. I don't know if they are right on this – maybe you can't force it to take more than order $2^{n/3}$ steps, say, but it would certainly be of interest both for DNA computors, and also for other people (DIMACS, for example) to find out.

On the other hand, there are rays of hope. Life does most things better in vivo than we can do in vitro. So the logical approach is to Steal from the Masters, e.g. use life systems.

I think the most interesting idea I've seen, from the practical point of view, is Smith and Schweitzer's observation that RNA editing, e.g. in trypanosomes such as *T.Brucei*, may allow construction of a Turing Universal rewrite system. Since it was "designed by God," rather than by us, it is likely to be a much superior computer to the ones we are devising. For instance it probably runs at speeds of order 1 op per second, versus $\lesssim 1$ per hour of all the schemes proposed at this workshop. It would also be a "1 pot, self running, system," unlike everything else proposed. The logical path to pursue is therefore to investigate RNA editing further in the lab to really nail down the guide RNA code, the speeds you get, the chemical details... and try to clone up general purpose in vitro editing systems, and once that is done, computer scientists would need to work out the computational details of how best to embed a computer inside this.

Don Beaver also showed how to make a universal Turing machine with DNA and standard lab techniques. It is described in a manuscript available electronically `http://www.cse.psu.edu/ ~beaver/publications/tm.ps.Z`. In fact, there now appear to be a very large number of people who have worked out how to make Turing machines, in varying degrees of detail...

Beaver's Turing machine has some advantages in terms of conceptual simplicity over Smith and Schweitzer's, but it also has some disadvantages. I sent Beaver a more detailed criticism, but here I'll only mention the following

1. Beaver's scheme keeps needing to duplicate the entire tape using polymerase – slow, expensive, error prone, while Smith and Schweitzer only need to use polymerase on small portions of their DNA.

2. The inexact pairing component central to Beaver's scheme is capable of *chaining* several of his Turing machines together, creating a monstrosity. That is a serious bug, in my opinion. Beaver suggested the monstrosities could be removed by length segregation, but since Turing machines worthy of the name have unpredictable lengths, and anyway this would lead to exponentially accumulating losses, I'm unhappy with that fix. The right fix is probably to switch to solid support as in the Smith-Schweitzer scheme, but of course that has its problems too...

3. Beaver uses S1 nuclease to digest single strand DNA, but it can nick, cut, and eventually digest double strand DNA too, and has to be used at pH's at which DNA tends to depurinate. This effect will be a limitation for Beaver, but doesn't affect the validity of his scheme for small problems.

More interestingly, Beaver showed via a different scheme that the set of computational problems soluble by DNA techniques in a polynomial number of bio-steps, was in fact not merely NP (as shown by Adleman, Lipton, Smith and Schweitzer, etc.) but actually PSPACE.

This is under the same assumption (or rather pretense) that there is an exponential amount of DNA around, that A, L, S & S, etc. must make to get NP.

But, as Smith pointed out, Beaver is *also* relying on the additional assumption, which Smith and Adleman dislike, that some of the single biosteps are allowed to be DNA annealings which, according to the usual formula, will take exponentially long time.

So, it now is unclear what *the class of problems soluble by DNA methods in polynomial real time* ("CPSDNAPRT"? ugh.) is. We know it lies somewhere between NP and PSPACE but it isn't clear where. As Smith points out in his paper, NDTMs *with* interprocessor communication mechanisms, may be able to solve a bigger class than NP. This is an open problem.

Erik Winfree in his talk made inroads into it by showing that the fundamental set of Liptonian abstract DNA operations allowed the solution exactly of the class of problems soluble in polylog space by Turing machines, and if you supplement the Liptonian set by allowing "amplify" steps, you get exactly the class of problems soluble in polylog space by nondeterministic Turing machines.

Finally, **Paul Rothemund** talked about Yet Another Scheme For Making a Turing Machine with DNA, devised by him. It works entirely with type IIS restriction enzymes (a considerable number of different types are needed) and ligases, and Rothemund claims he has worked out the *full details* of the DNA sequences required, et cetera showing how to embed Minsky's 4×7 universal Turing machine (the smallest known one). This embedding is actually a very tight fit; Rothemund doesn't think he could do any larger UTM. This is entirely unlike Beaver and Smith-Schweitzer's Turing machines, which are quite flexible – if you want to put in a larger transition matrix, no problem.

There was no way Rothemund was going to show us the full details and get us to comprehend it all in a 25 minute talk, and he doesn't have a distributable paper yet, but overall I'm very impressed. This guy has only got an undergraduate education, but he's struck out on his own doing research, he claims he figured out the bare bones of his scheme actually over a year before Adleman did but never published... and he seems to have dug up a lot more biochemical details than and has worked harder than most or all the other speakers at the workshop. (E.g. he's the only one claiming to have laid out all the Turing machine details, which conventional computer scientists like me prefer to ignore.)

Rothemund's scheme is the only one which works entirely with double strand DNA, which could be a big advantage for the purpose of avoiding hydrolysis damage – which as Smith and Schweitzer showed was a big limitation. Unfortunately, because Rothemund has to rely on lots of kinds of restriction enzymes, probably some of them are going to have to be rather poor performing ones. The imperfect specificities of restriction enzymes is probably just as severe an ultimate limitation as DNA hydrolysis is, so I'm still sticking to my Pessimism Platform.

Finally, to point out the obvious, there was a severe deficit of actual in-the-lab biochem types speaking at this workshop. Everything except for Adleman's experiment is totally on paper or in the mind at present – unsupported by experiments – and biochem people looking at the gel photos in the Adleman Science paper seem to think that already with $n = 7$ cities it looks like he was starting to encounter some problems, although of course, the bottom line is that it worked.

Lipton and Baum did a fine job organizing this workshop, and it was fun. Adleman is starting a DNA computation Email group (write him at `adleman@cs.usc.edu`) to encourage more flaming on this topic, and Lipton has incited vague threatening rumors that a book may arise out of all this.