ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta jaderná a fyzikálně inženýrská

VÝZKUMNÝ ÚKOL

Praha, 2014

Jakub Klemsa

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta jaderná a fyzikálně inženýrská

Katedra matematiky

VÝZKUMNÝ ÚKOL

Modely sebeskládajících DNA nanostruktur

Models of self-assembling DNA nanostructures

Vypracoval: Jakub Klemsa

Školitel: Ing. Štěpán Starosta, Ph.D.

Akademický rok: 2013/2014

Na toto místo přijde svázat **zadání mého výzkumného úkolu**!

V jednom z výtisků musí být **originál** zadání, v ostatních kopie.

Čestné prohlášení

Prohlašuji na tomto místě, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškerou použitou literaturu.

V Praze dne January 30, 2014

. . . . . .
Jakub Klemsa

Poděkování

Děkuji Ing. Štěpánu Starostovi, Ph.D., za vedení mého výzkumného úkolu a za podnětné návrhy, které ho obohatily.

Jakub Klemsa

Název práce:	**Modely sebeskládajících DNA nanostruktur**
Autor:	Jakub Klemsa
Obor:	Inženýrská informatika
Zaměření:	Matematická informatika
Druh práce:	Výzkumný úkol
Vedoucí práce:	Ing. Štěpán Starosta, Ph.D.,
Konzultant:	—
Abstrakt:	Abstrakt CZ.
Klíčová slova:	Klíčová.

Title:	**Models of self-assembling DNA nanostructures**
Author:	Jakub Klemsa
Abstract:	Abstrakt EN.
Key words:	Keywords.

Contents

Prolog

1959: Feynman's visionary talk, [5];

The ground-breaking work was carried out by Adleman, [2], who showed that DNA computation is practically feasible. In his experiment, Adleman used special DNA sequences for solving Hamiltonian Path Problem, one of the most typical NP-complete problems.

... Extreme parallelism! But also possibility of errors.

Work overview

Chapter 1: Intro.
Chapter 2: First of all I will describe models which exploit specific DNA structure.
Chapter 3: Abstract Tile Assembly Model, temperature, 2D vs. 3D.
Positive integers $\mathbb{N}$.

Chapter 1

Introduction to DNA computation

1.1 Basic DNA principles

Backbone: deoxyribose + phosphate; $5' \to 3'$ ends (due to deoxyribose atoms numbering); bases: adenine, thymine, cytosine, guanine; Watson-Crick complementarity; Polymerase chain reaction; Gel electrophoresis; biostep; hairpin;

1.2 Complexity, languages

P, NP, co-NP, PP, #P, PSpace. Enough? Maybe also polynomial hierarchy: $\Sigma_k P$ and $\Pi_k P$ languages (alternating Turing machine with bounded alternation, [7]).

Regular languages, context-free languages, recursively enumerable languages.

HPP: Adleman uses $O(n)$ biosteps, Winfree one. SAT: Lipton's contribution using m biosteps ($m = \#clauses$), [9], Lipton's set of speedup problems, [8]. Lipton 95 describes basic DNA operations. Energy efficiency (Adleman). NP definition?

Chapter 2

Strand models

Quick overview of considered structures. Winfree's overview (pg 29 – considered molecules, pg 36 – sizes of DAE and a better picture, pg 37 – comparison of DAO/DAE in a lattice, explanation pg 43).

Seeman, Fu and their DAO/DAE in [6], is the picture of DAO strange?

2.1 Adleman's experiment

Adleman showed in his ground-breaking work, [2], that DNA molecules are really capable of computation. He exploited that huge parallelism possible in DNA computation for one of the most fundamental NP-complete problems – the Hamiltonian Path Problem (HPP) in directed graph with designated vertices v_{begin} and v_{end}.

Let us remind this type of HPP. Given a directed graph G_n with n vertices and two designated vertices v_{begin} and v_{end}, the problem is to answer whether there exists an oriented path from v_{begin} to v_{end} through the graph such that the path visits every vertex. Note that *path* cannot visit any vertex more than once from definition.

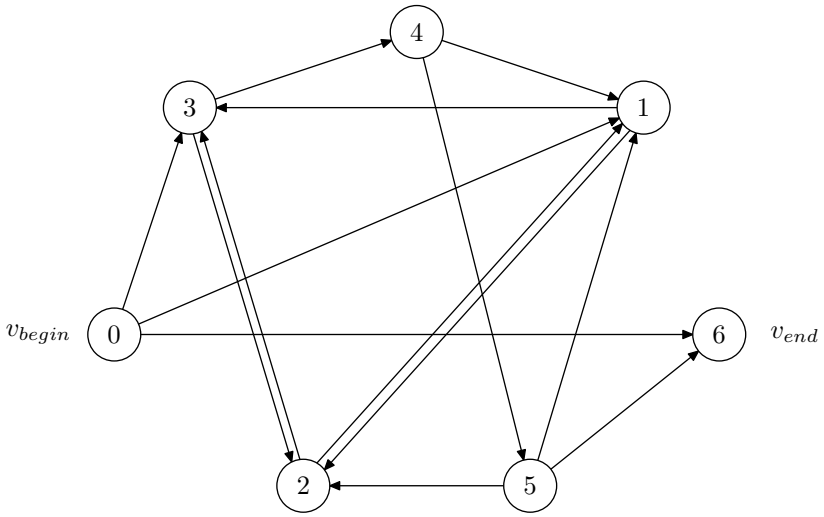


Figure 2.1: Adleman's original graph.

Adleman originally used a graph with seven vertices shown in figure 2.1. It can be seen that the path $0 \to 1 \to 2 \to 3 \to 4 \to 5 \to 6$ is Hamiltonian[1].

Adleman first presents this non-deterministic five-step algorithm, whose steps are then described in terms of DNA manipulations:

Step 1 Generate random paths through the graph.

Step 2 Keep only those paths that begin with v_{begin} and end with v_{end}.

Step 3 If the graph has n vertices, then keep only those paths that enter exactly n vertices.

Step 4 Keep only those paths that enter all of the vertices of the graph at least once.

Step 5 If any paths remain, say "Yes"; otherwise, say "No."[2]

To see how DNA can compute, let us describe this example more precisely. The computation itself (I mean the inception of the final solution) is hidden in Step 1. Each vertex i is associated with a random[3] 20-mer sequence of DNA, let us denote its $5' \to 3'$ orientation by O_i, its 10-mer prefix by p_i and its 10-mer suffix by q_i. Each edge $i \to j$ is then associated with $\overline{q_i p_j}$ sequence with reverse backbone orientation ($3' \to 5'$) where $\overline{q_i}$ stands for Watson-Crick complementary word. There is an exception for $i = begin$ and $j = end$: instead of $\overline{q_{begin} p_j}$ there is $\overline{O_{begin} p_j}$ and in a similar way for $j = end$.

$$
\begin{array}{lc}
 & O_{i\to j} \\
3' \leftarrow & \ldots|\texttt{CTTTCGAATA AAGATCAGTC}|\ldots \quad \leftarrow 5' \\
5' \to & \ldots|\texttt{CCTCTCGCGA GAAAGCTTAT}|\texttt{TTCTAGTCAG CACTCTTTGT}|\ldots \quad \to 3' \\
 & O_i \qquad\qquad\qquad O_j
\end{array}
$$

Figure 2.2: Example of assigned sequences.

It can be easily seen that all correctly bonded double-strands correspond with a valid path in G_n. Moreover, all complete double-strands represent a valid path from v_{begin} to v_{end} through G_n.

All the other steps are fully described in [2]. The most important thing is that the most time-demanding step is Step 4. In this step one has to purify the product of Step 3 with a biotin-avidin magnetic beads system. This process extracts consequently for every vertex i only those DNA strands which contain a substring representing vertex i. Thus its biostep complexity is $O(n)$. If we assume that one biostep takes at least tens of minutes and it should be performed repeatedly to avoid errors, we can conclude that $O(n)$ is just too much[4].

2.2 Single-stranded molecules

SAT in $O(1)$ biosteps etc.

[1] Note that it can be re-labelled such a nice way without loss of generality.

[2] This is the original version, I would rectify the fifth step: If any paths remain, say "Yes"; otherwise, say "*I do not know.*" That is because NP problem gives answer "Yes" iff there *exists* supporting solution. To say "No" one needs to show that *all* solutions do not satisfy. That is exactly the difference between NP and co-NP.

[3] We will expect those sequences to be different enough.

[4] Winfree, [11], gives a positive solution.

2.3 Double-stranded molecules

2.3.1 Linear strands

Equivalent to regular languages.

2.3.2 Dendrimer structures

Equivalent to context-free languages.

2.4 Double crossover molecules

Equivalent to recursively enumerable languages (Turing universal). Important notes in 3.2.5 Winfree – single side hybridization – how to avoid. Tricky solution of Hamiltonian Path Problem.

Chapter 3

Wang tile models

3.1 Definition

More abstract model where one handles only with "glues" on edges of Wang tiles. Define *temperature*.

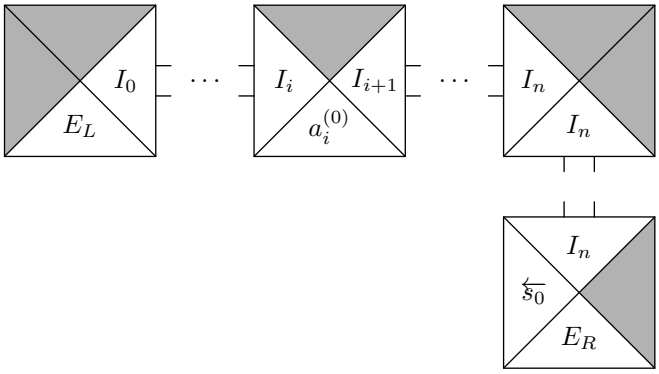
3.2 Computational power

Give table of Turing universality [4].

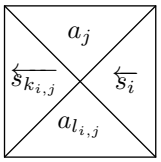
See [11]. Many other results in [4], [3], [10], [1] ...

3.2.1 Turing universality of 2D tiles at $T = 2$

Input tape:



Comes from right, continues left:



Comes from right, continues back to the right:

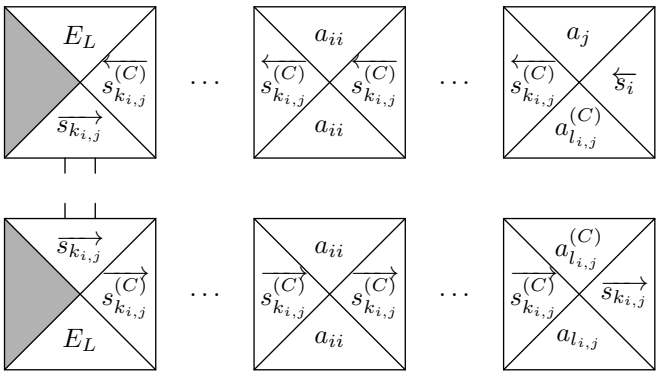
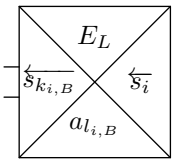
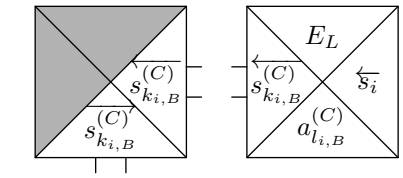


Figure 3.1: Tileset 1/2

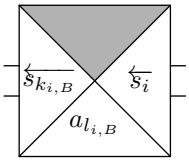
Reads EndLeft, continues left:



Reads EndLeft, continues back to the right:



Comes from right and reads Blank, continues left:



Comes from right and reads Blank, continues back to the right:

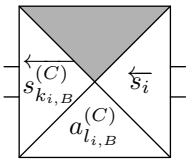


Figure 3.2: Tileset 2/2

error-tolerant rules – Gacs and Reif, 1988

Chapter 4

Results

4.1 Graph 3-coloring

Might be problem with ordering, see original Knuth's algorithm at `http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/networks/oetsen.htm`!

Epilog

The very last section to be done.

Bibliography

[1] Leonard Adleman, Qi Cheng, Ashish Goel, and Ming-Deh Huang. Running time and program size for self-assembled squares. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 740–748. ACM, 2001.

[2] Leonard M Adleman. Molecular computation of solutions to combinatorial problems. *Science - New York then Washington*, pages 1021–1024, 1994.

[3] Bahar Behsaz, Ján Maňuch, and Ladislav Stacho. Turing universality of step-wise and stage assembly at temperature 1. In *DNA Computing and Molecular Programming*, pages 1–11. Springer, 2012.

[4] Matthew Cook, Yunhui Fu, and Robert Schweller. Temperature 1 self-assembly: Deterministic assembly in 3d and probabilistic assembly in 2d. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 570–589. SIAM, 2011.

[5] Richard P Feynman. There's plenty of room at the bottom. *Engineering and Science*, 23(5):22–36, 1960.

[6] Tsu Ju Fu and Nadrian C Seeman. Dna double-crossover molecules. *Biochemistry*, 32(13):3211–3220, 1993.

[7] Dexter Kozen. *Theory of computation*. Springer, 2006.

[8] Richard J Lipton. Speeding up computations via molecular biology. *DNA Based Computers*, 27:67–74, 1996.

[9] Richard J Lipton et al. Dna solution of hard computational problems. *Science*, 268(5210):542–545, 1995.

[10] Paul WK Rothemund and Erik Winfree. The program-size complexity of self-assembled squares. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 459–468. ACM, 2000.

[11] Erik Winfree. *Algorithmic self-assembly of DNA*. PhD thesis, California Institute of Technology, 1998.