$5' \longrightarrow 3'$ Watson-Crick Automata with Several Runs

Peter Leupold

Department of Mathematics, Faculty of Science
Kyoto Sangyo University, Japan

Joint work with Benedek Nagy (Debrecen)
Presentation at NCMA 2009

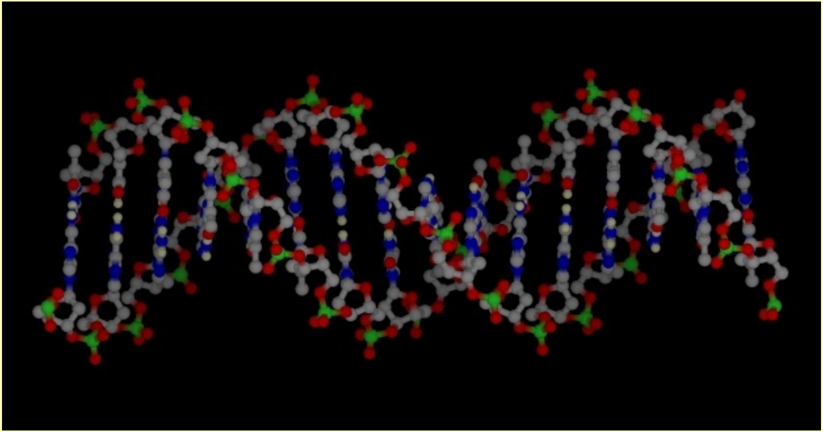
Motivation for Watson-Crick Automata

What if finite automata worked on DNA strands instead of abstract strings of symbols?

Motivation for Watson-Crick Automata

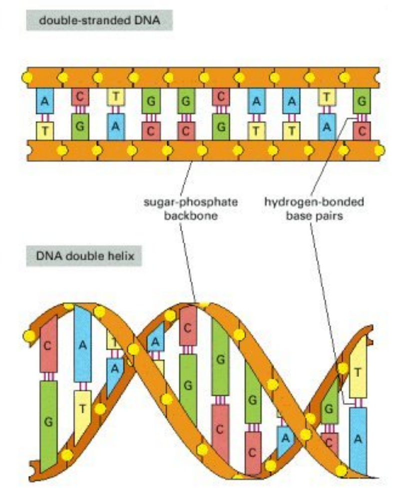
What if finite automata worked on DNA strands instead of abstract strings of symbols?

A DNA strand is not just a simple string, but normally is a double strand with a three-dimensional helix structure.



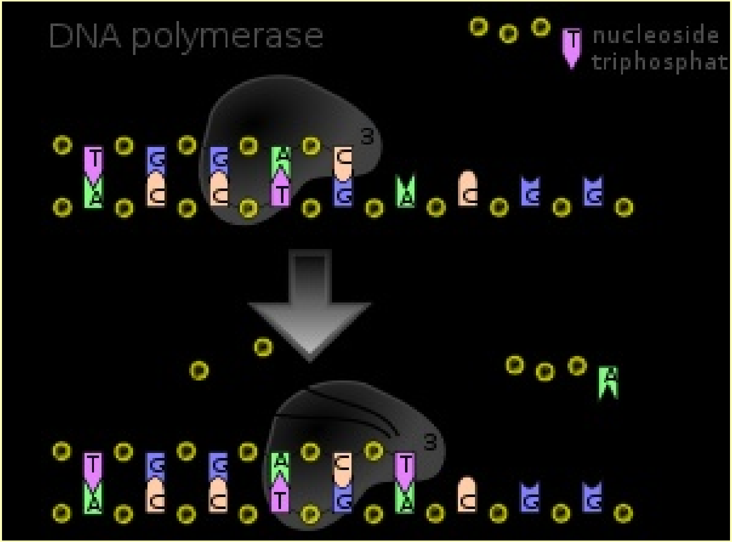
Abstracting the Structure

We view a DNA strand as a linear sequence of pairs of complementary symbols..



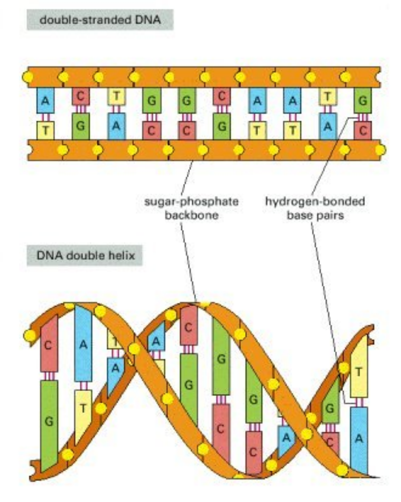
The Head

What could an automaton's reading head look like?



Two Heads

To read both parts of the double strand, two heads are necessary.



The Role of Complementarity

Intuitively, a strand and its complement are equivalent from an information theoretic point of view.

The Role of Complementarity

Intuitively, a strand and its complement are equivalent from an information theoretic point of view.

Kuske and Weigel (at DLT 2004) observed, that all language classes of WK-automata are the same, even when the complementarity relation is simply the identity relation.

The Role of Complementarity

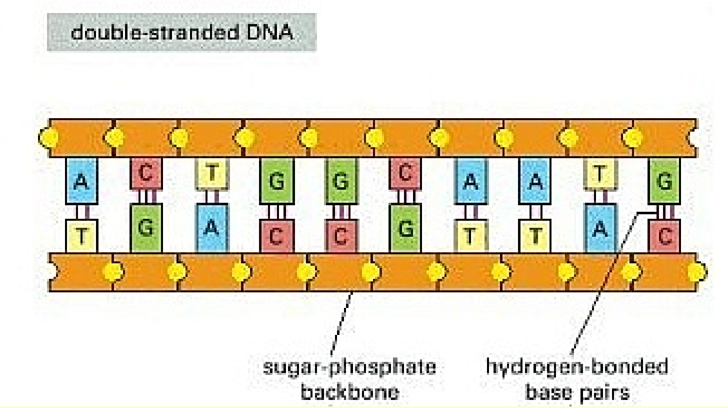
Intuitively, a strand and its complement are equivalent from an information theoretic point of view.

Kuske and Weigel (at DLT 2004) observed, that all language classes of WK-automata are the same, even when the complementarity relation is simply the identity relation.

Therefore we will let the automata's two heads work on the same string for simplicity of notation.

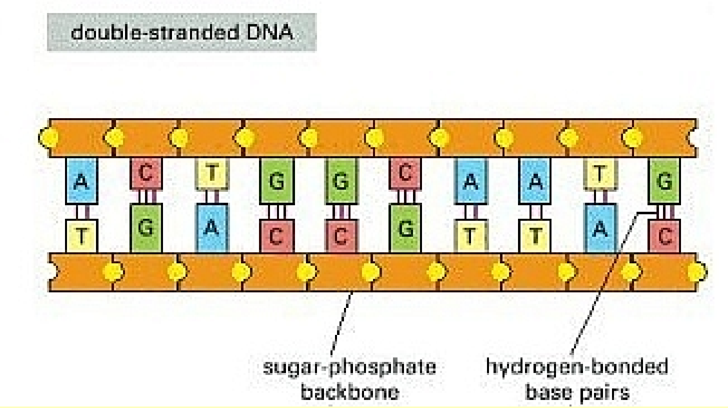
Moving the Heads

The two strands have a direction...



Moving the Heads

The two strands have a direction...



...and their directions are opposite.

$5' \longrightarrow 3'$ Watson-Crick Automata modify standard Watson-Crick Automata with the preceding two observations made in mind.

Thus there are two main differences:

$5' \longrightarrow 3'$ Watson-Crick Automata

$5' \longrightarrow 3'$ Watson-Crick Automata modify standard Watson-Crick Automata with the preceding two observations made in mind.

Thus there are two main differences:

- The tape is a standard one, no complementarity.

$5' \longrightarrow 3'$ Watson-Crick Automata

$5' \longrightarrow 3'$ Watson-Crick Automata modify standard Watson-Crick Automata with the preceding two observations made in mind.

Thus there are two main differences:

- The tape is a standard one, no complementarity.
- The two heads start on opposite ends of the input and run in opposite directions.

$5' \longrightarrow 3'$ Watson-Crick Automata

$5' \longrightarrow 3'$ Watson-Crick Automata modify standard Watson-Crick Automata with the preceding two observations made in mind.

Thus there are two main differences:

- The tape is a standard one, no complementarity.
- The two heads start on opposite ends of the input and run in opposite directions.

So in contrast to conventional Watson-Crick automata we can expect ease in recognizing palindromic structures and problems in recognizing copy-type structures.

Runs and Acceptance

A <u>run</u> is a complete reading of the input string by both heads in their respective direction.

Runs and Acceptance

A run is a complete reading of the input string by both heads in their respective direction.

- There are start and end markers on the input string.

Runs and Acceptance

A run is a complete reading of the input string by both heads in their respective direction.

- There are start and end markers on the input string.
- After a run the heads turn around and can read the respectively other side of the strand – which is the same string in our case.

Runs and Acceptance

A run is a complete reading of the input string by both heads in their respective direction.

- There are start and end markers on the input string.
- After a run the heads turn around and can read the respectively other side of the strand – which is the same string in our case.

An input word is accepted in k runs, iff the automaton halts in an accepting state after k runs.

Variants

Definition

The class of languages accepted by $5' \longrightarrow 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\underset{m}{\rightleftharpoons}\mathbf{WK}$. Such automata are called

Variants

Definition

The class of languages accepted by $5' \longrightarrow 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\overset{\rightleftharpoons}{m}\mathbf{WK}$. Such automata are called

N: stateless if they have only one state;

Variants

Definition

The class of languages accepted by $5' \rightarrow 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\overset{\rightleftharpoons}{{}_m}\mathbf{WK}$. Such automata are called

- **N**: stateless if they have only one state;
- **F**: all-final if they have only final states;

Variants

Definition

The class of languages accepted by $5' \rightarrow 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\underset{m}{\rightleftharpoons}\mathbf{WK}$. Such automata are called

- **N**: stateless if they have only one state;
- **F**: all-final if they have only final states;
- **S**: simple if at most one head is moving in every step;

Variants

Definition

The class of languages accepted by $5' \longrightarrow 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\underset{m}{\rightleftharpoons}\mathbf{WK}$. Such automata are called

- **N**: stateless if they have only one state;
- **F**: all-final if they have only final states;
- **S**: simple if at most one head is moving in every step;
- **1**: 1-limited if exactly one letter is read in every step;

Variants

Definition

The class of languages accepted by $5' \to 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\underset{m}{\rightleftharpoons}\mathbf{WK}$. Such automata are called

- **N**: stateless if they have only one state;
- **F**: all-final if they have only final states;
- **S**: simple if at most one head is moving in every step;
- **1**: 1-limited if exactly one letter is read in every step;
- **D**: deterministic if for all possible configurations c there is at most one configuration such that $c \Rightarrow c'$.

Variants

Definition

The class of languages accepted by $5' \to 3'$ full reading finite Watson-Crick automata in m runs is denoted by $\overset{\rightleftharpoons}{_m}\mathbf{WK}$. Such automata are called

- **N**: stateless if they have only one state;
- **F**: all-final if they have only final states;
- **S**: simple if at most one head is moving in every step;
- **1**: 1-limited if exactly one letter is read in every step;
- **D**: deterministic if for all possible configurations c there is at most one configuration such that $c \Rightarrow c'$.

The corresponding classes of languages are denoted by $U\overset{\rightleftharpoons}{_k}\mathbf{WK}$ where U is one of the symbols associated to the variants in the enumeration above. Also combinations of these variants are possible.

Behaviour of $5' \longrightarrow 3'$ Watson-Crick Automata

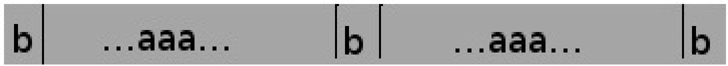
Lemma

Let A be a deterministic $5' \rightarrow 3'$ WK-automaton accepting the language $L := \{ba^nba^nb : n > 0\}$ in one run. For long enough n the computation for an input word ba^nba^nb goes through a configuration where one head is in the first factor a^n while the other is in the second.

Behaviour of $5' \longrightarrow 3'$ Watson-Crick Automata

Lemma

Let A be a deterministic $5' \rightarrow 3'$ WK-automaton accepting the language $L := \{ba^nba^nb : n > 0\}$ in one run. For long enough n the computation for an input word ba^nba^nb goes through a configuration where one head is in the first factor a^n while the other is in the second.



An Infinite Hierarchy of $5' \longrightarrow 3'$ Watson-Crick Automata

Theorem

For every $m > 0$ the class of languages accepted by $5' \rightarrow 3'$ deterministic WK-automaton in m runs is properly contained in the class of languages accepted in $m+1$ runs, i.e., $\mathbf{D}_m^{\rightleftharpoons}\mathbf{WK} \subsetneq \mathbf{D}_{m+1}^{\rightleftharpoons}\mathbf{WK}$.

Theorem

For every $m > 0$ the class of languages accepted by $5' \rightarrow 3'$ deterministic WK-automaton in m runs is properly contained in the class of languages accepted in $m+1$ runs, i.e., $\mathbf{D}_m^{\rightleftharpoons}\mathbf{WK} \subsetneq \mathbf{D}_{m+1}^{\rightleftharpoons}\mathbf{WK}$.

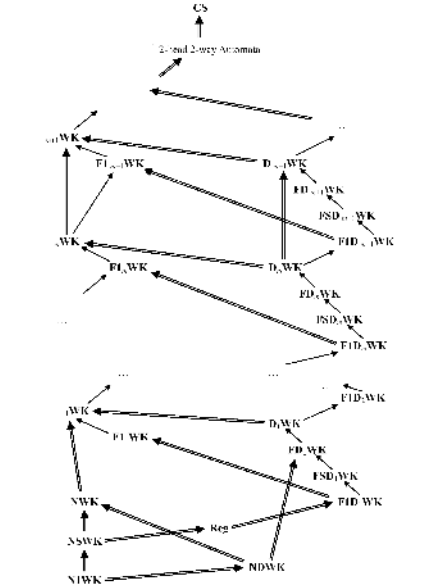
The witness languages are:

$$L_m := \{ww : w \in L'_m\}$$

where

$$L'_m := \{a^{n_1}b^{n_2}a^{n_3}b^{n_4}\ldots a^{n_{2m-1}}b^{n_{2m}} : n_i > 0 \text{ for } 1 \leq i \leq 2m\}.$$

An Infinite Hierarchy of $5' \longrightarrow 3'$ Watson-Crick Automata



Decidability

The fact that $5' \longrightarrow 3'$ Watson-Crick automata accept non-context-free languages even in just one run suggests that most decidability problems related to them will not be solvable.

Decidability

The fact that $5' \to 3'$ Watson-Crick automata accept non-context-free languages even in just one run suggests that most decidability problems related to them will not be solvable.

However, since they do not accept all context-free languages, but language classes somewhat orthogonal to the Chomsky Hierarchy, things are not obvious.

The Emptiness Problem

Theorem

For the class $\mathbf{D}_{1}^{\rightleftharpoons}\mathbf{WK}$ *the non-empiness problem is undecidable.*

The Emptiness Problem

Theorem

For the class $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ the non-emptiness problem is undecidable.

For a Turing Machine M we define the language L_M that contains all words $w_1\#w_2\ldots\#w_{i-1}\#\#w_i^R\#w_{i-1}^R\ldots\#w_3^R\#w_2^R$ where $w_1, w_2, \ldots, w_i$ is a sequence of configurations of a computation of M, w_1 is an initial configuration, and w_i is a final and accepting configuration.

The Emptiness Problem

Theorem

For the class $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ *the non-emptiness problem is undecidable.*

For a Turing Machine M we define the language L_M that contains all words $w_1\#w_2\ldots\#w_{i-1}\#\#w_i^R\#w_{i-1}^R\ldots\#w_3^R\#w_2^R$ where $w_1, w_2, \ldots, w_i$ is a sequence of configurations of a computation of M, w_1 is an initial configuration, and w_i is a final and accepting configuration.

- read w_1 and w_2^R simultaneously

The Emptiness Problem

Theorem

For the class $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ *the non-emptiness problem is undecidable.*

For a Turing Machine M we define the language L_M that contains all words $w_1\#w_2\ldots\#w_{i-1}\#\#w_i^R\#w_{i-1}^R\ldots\#w_3^R\#w_2^R$ where $w_1, w_2, \ldots, w_i$ is a sequence of configurations of a computation of M, w_1 is an initial configuration, and w_i is a final and accepting configuration.

- read w_1 and w_2^R simultaneously
- check whether w_i and w_i^R are really the same

The Emptiness Problem

Theorem

For the class $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ the non-emptiness problem is undecidable.

For a Turing Machine M we define the language L_M that contains all words $w_1\#w_2\ldots\#w_{i-1}\#\#w_i^R\#w_{i-1}^R\ldots\#w_3^R\#w_2^R$ where $w_1, w_2, \ldots, w_i$ is a sequence of configurations of a computation of M, w_1 is an initial configuration, and w_i is a final and accepting configuration.

- read w_1 and w_2^R simultaneously
- check whether w_i and w_i^R are really the same
- accepts iff the string represents an accepting TM computation

Corollaries

Corollary

For the class $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ *the finiteness problem is undecidable.*

A deterministic TM accepts words in only one possible computation. Therefore L_M is finite iff M's language is finite.

Corollaries

Corollary

For the class $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ the finiteness problem is undecidable.

A deterministic TM accepts words in only one possible computation. Therefore L_M is finite iff M's language is finite.

Corollary

Every recursively enumerable language is a morphic image of a language from $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$.

The word accepted by a computation can be extracted from L_m by a simple morphism.

Open Problems

Open Problem

We have establishes the undecidability of the emptiness problem for deterministic $5' \to 3'$ WK-automata with one run. Close to the bottom of our hierarchy, the regular languages appear. By definition we have the inclusions $\mathbf{F1D}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{FSD}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{FD}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{D}_m^{\rightleftharpoons}\mathbf{WK}$ on the path between them, but as mentioned above, it is unclear, which of them are proper.

Open Problems

Open Problem

We have establishes the undecidability of the emptiness problem for deterministic $5' \rightarrow 3'$ *WK-automata with one run. Close to the bottom of our hierarchy, the regular languages appear. By definition we have the inclusions* $\mathbf{F1D}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{FSD}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{FD}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{D}_m^{\rightleftharpoons}\mathbf{WK}$ *on the path between them, but as mentioned above, it is unclear, which of them are proper. Somewhere on this path there must also lie the border between decidability and undecidability of the emptiness problem, because for regular languages it is decidable, even for context-free languages.*

Open Problems

Open Problem

We have establishes the undecidability of the emptiness problem for deterministic $5' \to 3'$ WK-automata with one run. Close to the bottom of our hierarchy, the regular languages appear. By definition we have the inclusions $\mathbf{F1D}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{FSD}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{FD}_m^{\rightleftharpoons}\mathbf{WK} \subseteq \mathbf{D}_m^{\rightleftharpoons}\mathbf{WK}$ on the path between them, but as mentioned above, it is unclear, which of them are proper. Somewhere on this path there must also lie the border between decidability and undecidability of the emptiness problem, because for regular languages it is decidable, even for context-free languages.

Open Problem

Secondly, we believe that $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK}$ is incomparable to the class of linear languages.

There are examples for $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK} \subsetneq LIN$, but no example for $\mathbf{D}_1^{\rightleftharpoons}\mathbf{WK} \supsetneq LIN$.