Special course in Computer Science: Molecular Computing

Lecture 8: Formalizing gene assembly

Vladimir Rogojin

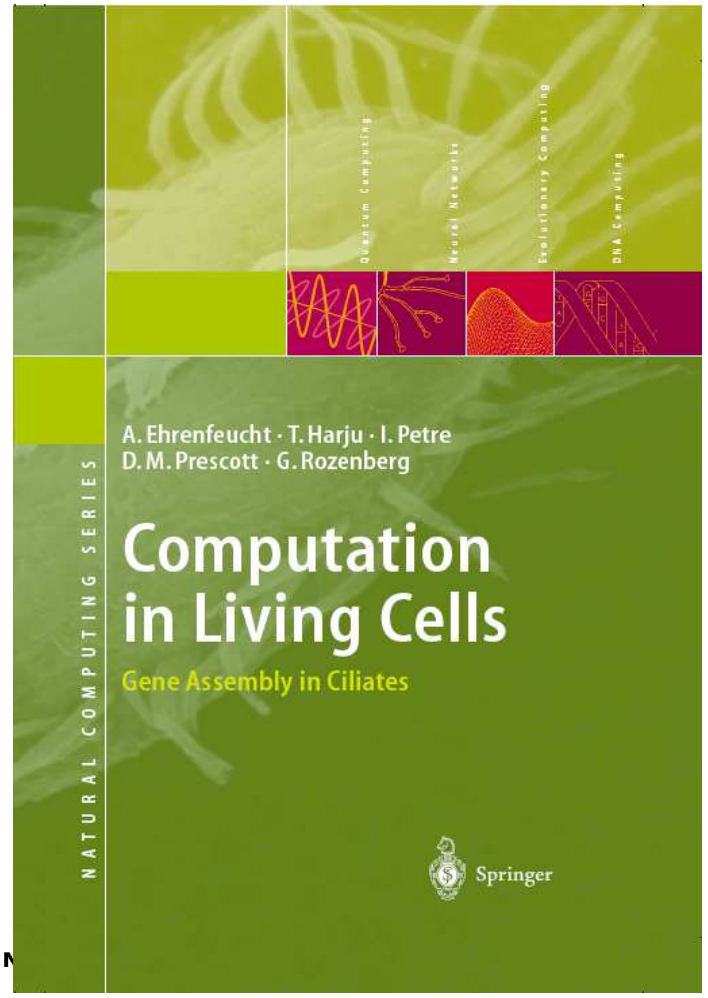
Department of CS, Abo Akademi

http://combio.abo.fi/teaching/special-course-in-computer-science-molecular-computing/

Gene assembly in ciliates – course book

Lectures 6 – 8, recommended reading:

- A. Ehrenfeucht et al., “Computation in Living Cells: Gene Assembly in Ciliates”, Springer, 2003



Model forming

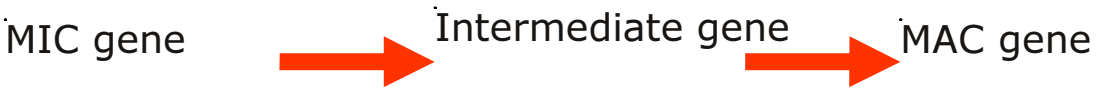
- In this lecture we will give an intuitive idea of how the formal model is built based on the intramolecular model presented in the previous lecture
- The goal of this lecture is to give a rough view of what a formal model means, how could it be built, and why is it useful

TUCS

Model forming

- To formalize gene assembly we need two steps:
  - Formalize the DNA molecules that constitute the micronuclear, intermediate, and the assembled gene
  - Formalize the way these objects are processes in gene assembly
- In our formalization we will keep a minimum of information that is still able to represent the essentials of the gene structure and keep track of gene assembly

WE FORMALIZE:



Model forming

- The model will be expressed on 3 levels of abstraction:
  - MDS descriptors
  - (Signed double occurrence) strings
  - (Signed overlap) graphs

Formalizing the genes

- Focus: the process itself
  - The sequence of operations used in the assembly
  - The macronuclear gene and its precursors

- The gene assembly is essentially a process of ordering and assembling MDSs
  - In the beginning: a sequence of k MDSs
  - With each operation, the number of MDSs decreases by 1 (ld, hi) or by 2 (dlad); bigger composite MDSs are formed
  - In the end: only one big composite MDS consisting of all k original MDSs spliced together in the right order: the macronuclear gene
  - Question: how much of the MDS should one represent?

Formalizing the genes

- Throughout the process, pointers play a central role: they facilitate all recombinations involved in the 3 operations
  - Pointers get eliminated throughout the process
  - Initially there are n-1 pointers (for n MDSs), each present in two copies
  - LD and HI eliminate (both copies of) 1 pointer, DLAD eliminates (both copies) of 2 pointers
  - The assembled gene is a contiguous sequence from the beginning marker to the ending marker: no pointers present anymore

Formalizing the genes

Idea:

- Represent the gene structure through the sequence of MDSs and pointers
- Represent the gene assembly as a process of sorting of MDSs, of composing bigger MDSs from its smaller parts and of pointer removals

First level: genes as MDS descriptors

- Idea: the whole structural information about the micronuclear gene and the intermediary molecules is given by the sequence of MDSs
  - Simplification: forget about the IESs, keep only the sequence of MDSs

- Keep the sequence of MDSs only: the essential part of each MDS (as far as gene assembly is concerned) is the pointers/markers of each MDS

First level: genes as MDS descriptors

- Each micronuclear MDS will be represented by its order number as it occurs in the assembled gene as well as by pair of its incoming and outgoing pointers or markers

- Each intermediate MDS will be represented by the order number of MIC MDSs composing it as well as by its incoming and outgoing pointers or markers

First level: genes as MDS descriptors

- Although the full structure of an MDS is $M_i=(\pi_i, \mu_i, \pi_{i+1})$, where

  – The outgoing pointer π_{i+1} of M_i is identical with the incoming pointer of M_{i+1}
  – $M_1=(b, \mu_1, \pi_2)$
  – $M_k=(\pi_k, \mu_k, e)$

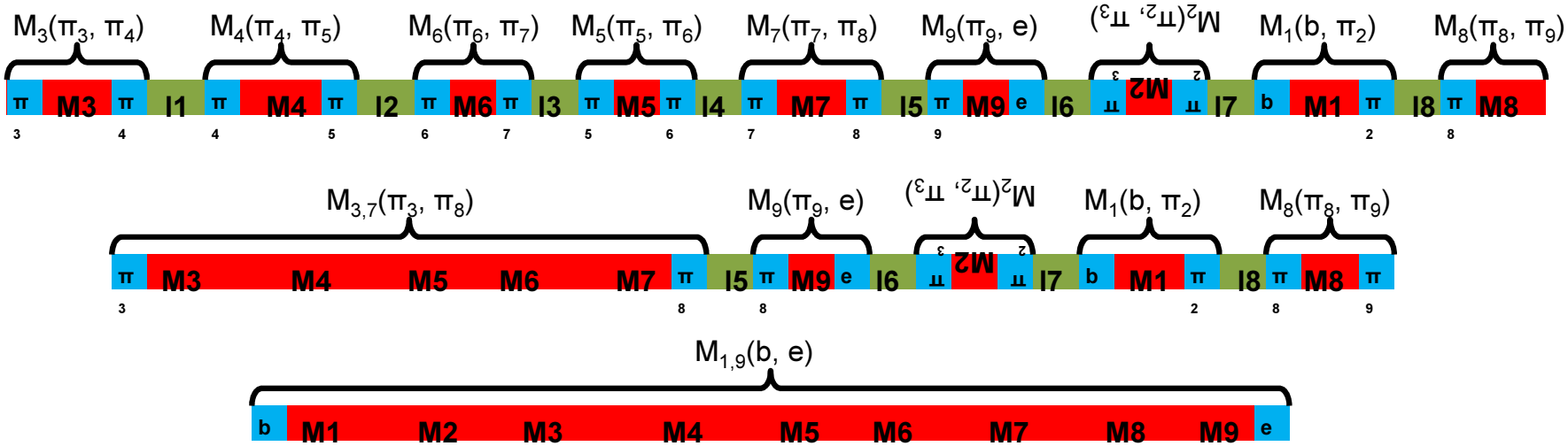
- We will ignore the content μ of an MDS

- we will represent each MIC MDS by its order number and by the pair of its pointers/markers as $M_i(\pi_i, \pi_{i+1})$

- We will represent each intermediate MDS by the order numbers of its MIC MDSs and by its pointers/markers as $M_{i,j}(\pi_i, \pi_{j+1})$

First level: genes as MDS descriptors

Example: gene *actin I* in *S.Nova.* Formalization



First level: genes as MDS descriptors

- This is already a big abstraction from the real gene structure
  - At this point we completely ignore the IESs and the bodies of the MDSs, i.e., most of the nucleotide sequence of the micronuclear gene
  - However, the information that we do keep is sufficient to keep track of all intermediary structures produced in the gene assemble and to keep track of the assembly itself
- One more simplification: the real DNA sequence of each pointer is not important as long as we know their positions on the gene
  - What is essential and we must keep is that the nucleotide sequence that is the incoming pointer of MDS i is identical with the nucleotide sequence making the outcoming pointer of MDS i-1
  - Remember also that MDS 1 starts with a beginning marker and the last MDS ends with an ending marker

First level: genes as MDS descriptors

- Idea: denote each pointer by an integer (the integer that denotes the incoming pointer of an MDS also identifies the MDS):
  - $M_i(i,i+1)$, $M_1(b,2)$, $M_k(k,e)$
  - Note: the integers are used here just as a notation for a DNA sequence
- Example:
  - M_3 is **ACTGTTTAAA…TATAATCGTA**
  - M_4 is **CGTATAATA…AATCTAGAGG**
  - $M_3(3,4)$, where 3 stands for ACTG and 4 stands for CGTA
  - $M_4(4,5)$, where 4 stands for CGTA and 5 stands for CTAGAGG

First level: MDS descriptors

- Ignore the order numbers of MDSs
- Keep only pointers
- Resulting formalism: MDS descriptors

- Each MIC MDS $M_i(i,i+1)$ we will represent as (i,i+1)
- Each intermediate (composite) MDS $M_{i,j}(i,j+1)$ we will represent as (i,j+1)
- First MDS $M_1(b,2)$ we will represent as (b,2)
- The last MDS $M_k(k,e)$ we will represent as (k,e)
- Inverted MIC MDS $\overline{M_i(i+1,i)} \Rightarrow \overline{(i+1,i)}$
- Inverted intermediate MDS $\overline{M_{i,j}(j+1,i)} \Rightarrow \overline{(j+1,i)}$
- Inverted first MDS $\overline{M_1(2,b)} \Rightarrow \overline{(2,b)}$
- Inverted last MDS $\overline{M_k(e,k)} \Rightarrow \overline{(e,k)}$

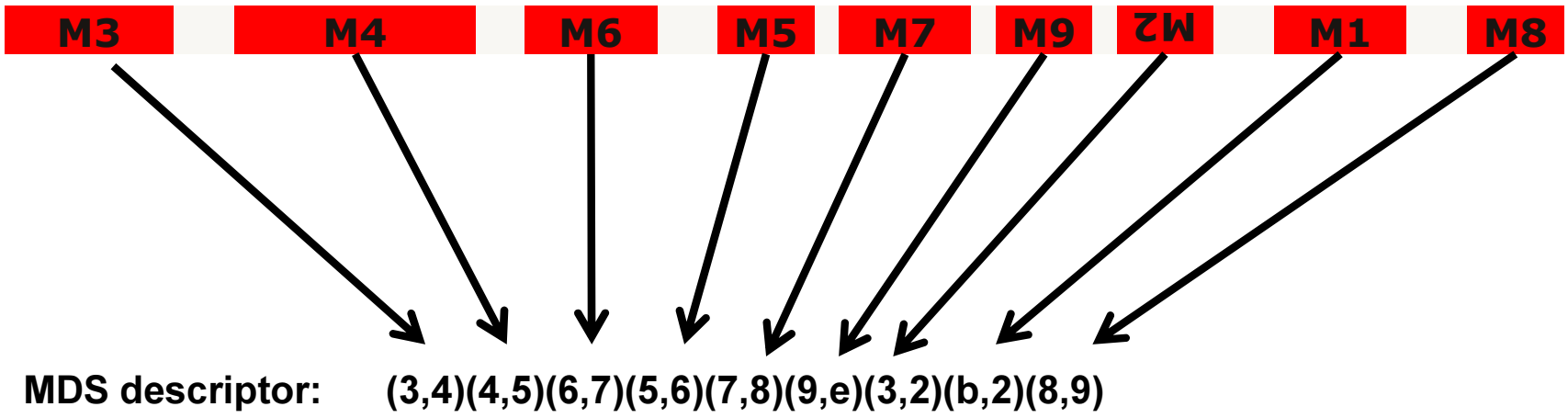
First level: MDS descriptors

- Assembled gene: $M_{1,k}(b,e)$ →
- (b,e)
- Inverted assembled gene: $\overline{M_{1,k}(e,b)}$ →
- $\overline{(e,b)}$

- $M_{3,5}(3,6)M_{1,2}(b,3)\overline{M_6(7,6)}M_{7,8}(7,e)$ →
- $(3,6)(b,3)\overline{(7,6)}(7,e)$

- $M_{2,3}(2,4)M_1(b,2)M_{5,6}(5,e)M_4(4,5)$ →
- $(2,4)(b,2)(5,e)(4,5)$

Genes as MDS descriptors

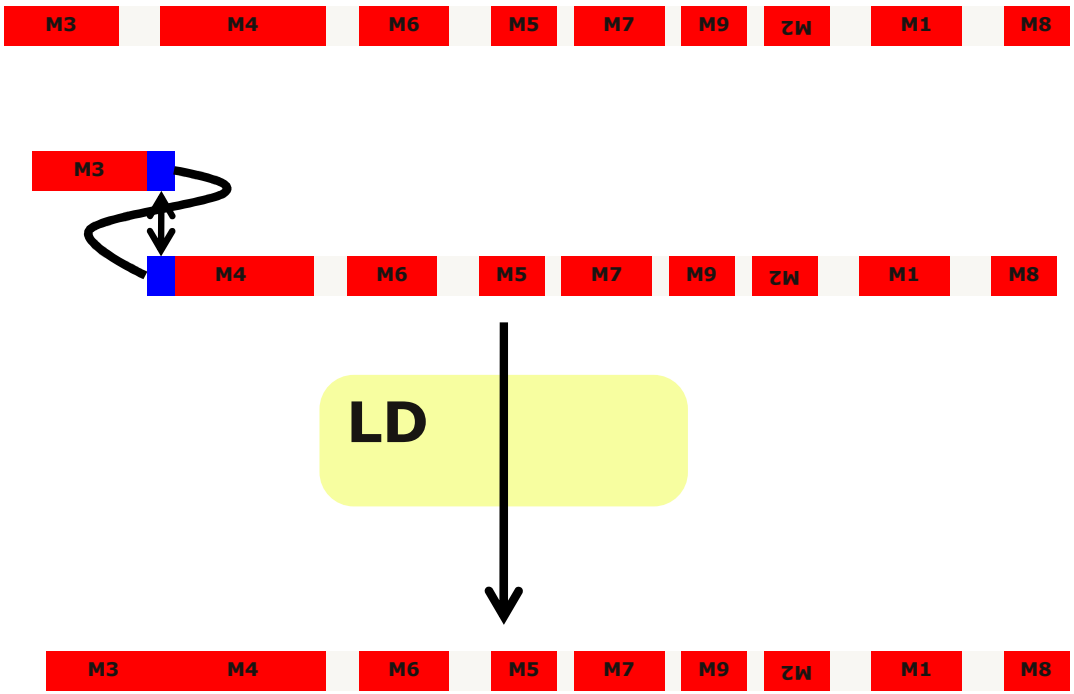


MDS descriptors -examples

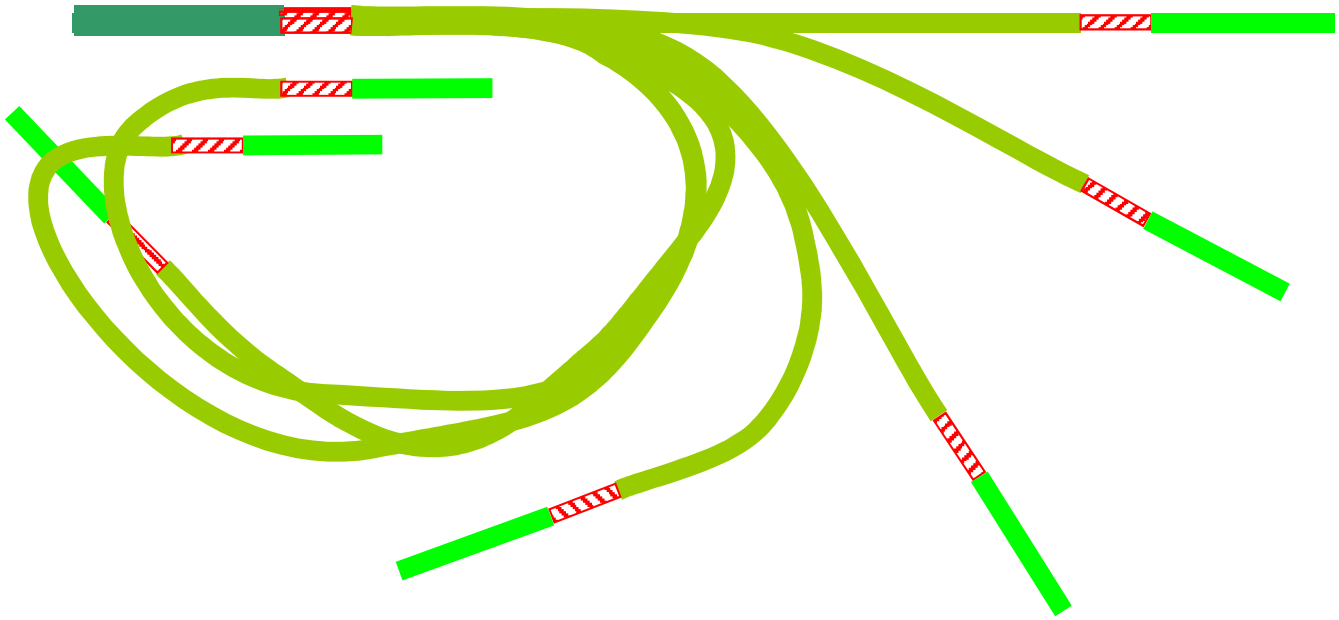
- $_{1}^{8}$(b,2)(5,e)(3,4)(4,5)(2,3) is an MDS descriptor
- (3,6)(6,e)(b,3) is an MDS descriptor
- (3,6)(6,e)(b,3)(4,5) is **not** an MDS descriptor
- (b,e), (-e,-b) are ***assembled*** MDS descriptors

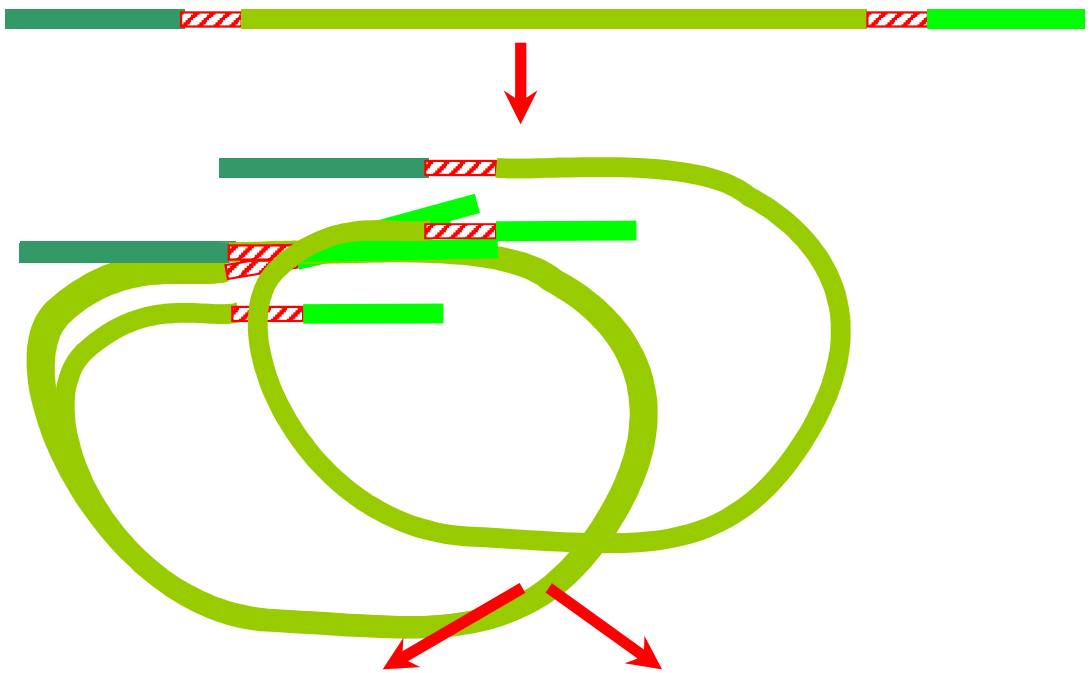
Example

- The micronuclear form of the gene on the right is
- $M_3(3,4)\ M_4(4,5)\ M_6(6,7)\ M_5(5,6)\ M_7(7,8)\ M_9(9,e)\ \overline{M_2(3,2)}\ M_1(b,2)\ M_8(8,9)$ - generic
- and also (3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9) – MDS descriptor



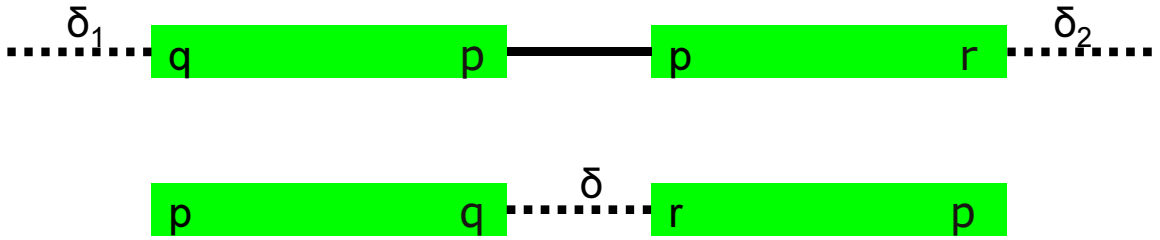
- The result after applying LD is $M_{3,4}(3,5)M_6(6,7)\ M_5(5,6)\ M_7(7,8)\ M_9(9,e)\ \overline{M_2(3,2)}\ M_1(b,2)\ M_8(8,9)$
- and also
- (3,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)
- (b,2)(8,9)




The repeat
The fold
The result

LD as an operation on MDS descriptors

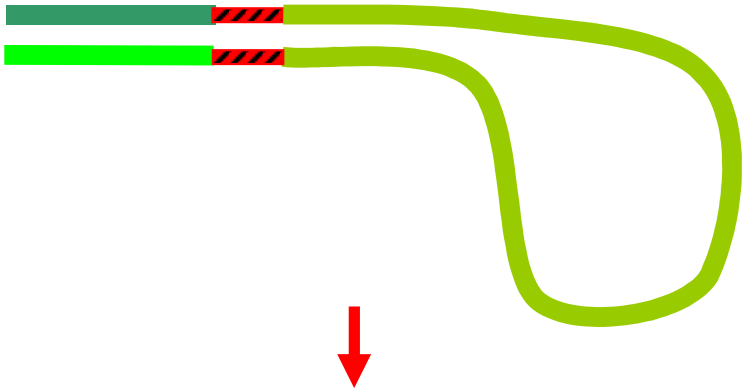
- Case 1: simple ld –one IES only separates the two occurrences of p
  - $ld_p(\ \delta_1(q,p)(p,r)\ \delta_2) = \delta_1(q,r)\ \delta_2$
- Case 2: boundary ld –the gene is bounded by the two occurrences of p
  - $ld_p(\ (p,q)\ \delta\ (r,p)\ ) = (r,q)\ \delta$



Courtesy of Ion Petre

The repeat

HI

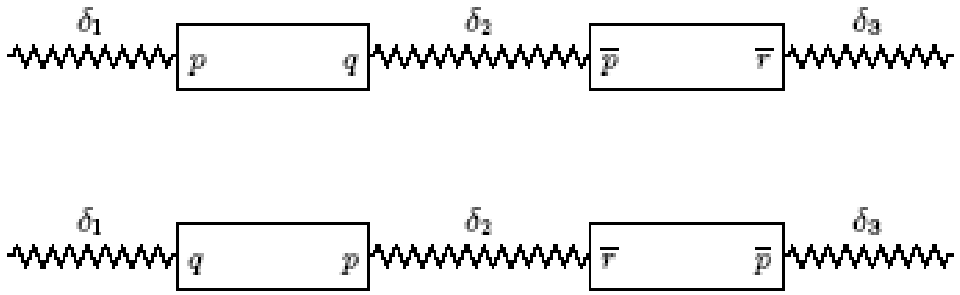


The fold

The result

HI as an operation on MDS descriptors

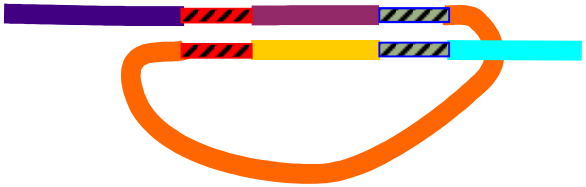
- hi_p(δ1 (p,q) δ2 (-p,-r) δ3) = δ1 δ2 (-q,-r) δ3
- hi_p (δ1 (q,p) δ2 (-r,-p) δ3) = δ1 (q,r) δ2 δ3



Courtesy of Ion Petre

The repeat

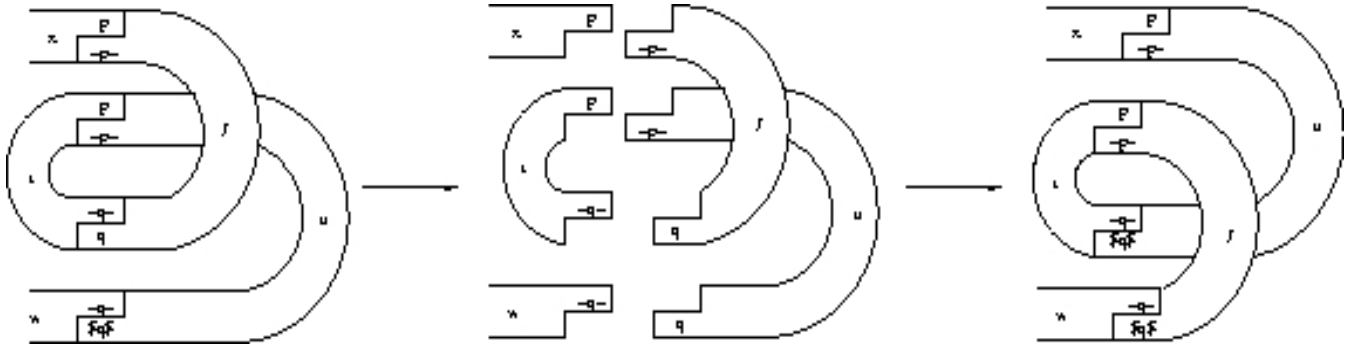
DLAD



The fold

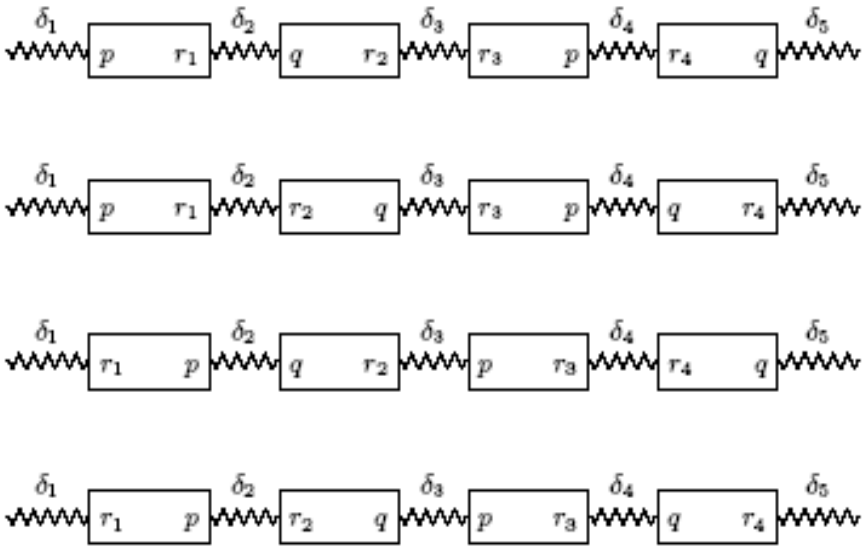
The result

Dlad on MDS descriptors



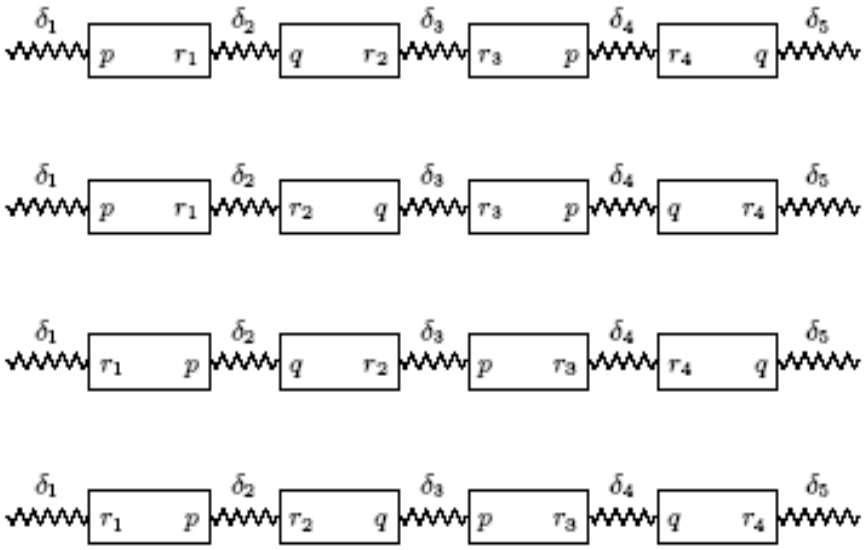
1. The first occurrence of p is *incoming*, the first occurrence of q is *incoming*
2. The first occurrence of p is *incoming*, the first occurrence of q is *outgoing*
3. The first occurrence of p is *outgoing*, the first occurrence of q is *incoming*
4. The first occurrence of p is *outgoing*, the first occurrence of q is *outgoing*

Dlad on MDS descriptors



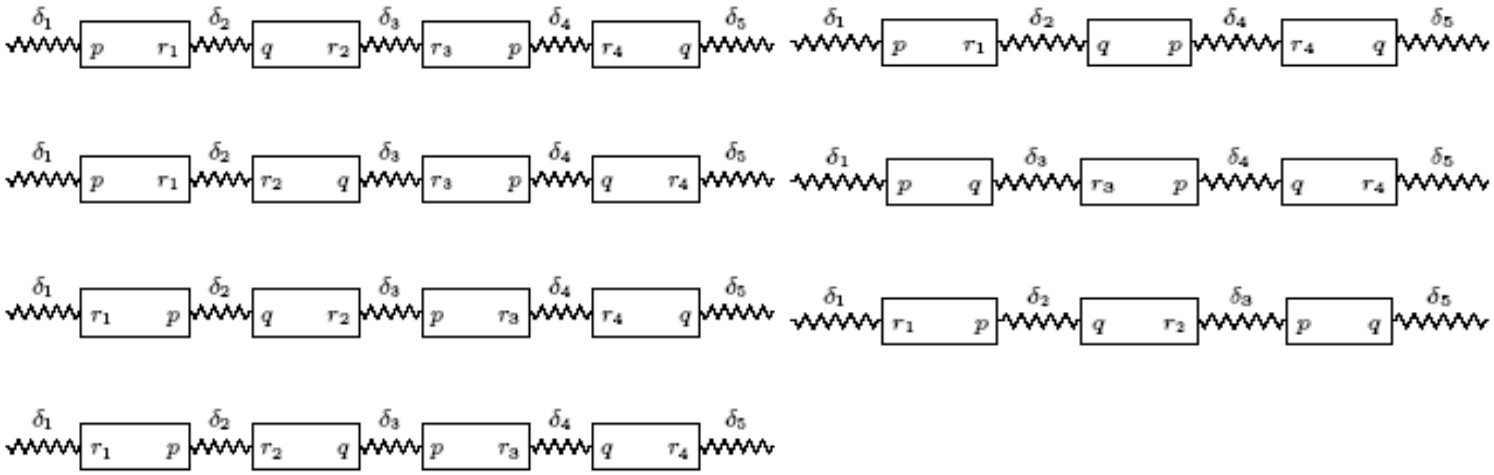
1. The first occurrence of p is *incoming*, the first occurrence of q is *incoming*
2. The first occurrence of p is *incoming*, the first occurrence of q is *outgoing*
3. The first occurrence of p is *outgoing*, the first occurrence of q is *incoming*
4. The first occurrence of p is *outgoing*, the first occurrence of q is *outgoing*

Dlad on MDS descriptors



1. $dlad_{p,q}(\delta1(p,r1)\delta2(q,r2)\delta3(r3,p)\delta4(r4,q)\delta5) = \delta1\delta4(r4,r2)\delta3(r3,r1)\delta2\delta5$
2. $dlad_{p,q}(\delta1(p,r1)\delta2(r2,q)\delta3(r3,p)\delta4(q,r4)\delta5) = \delta1\delta4\delta3(r3,r1)\delta2(r2,r4)\delta5$
3. $dlad_{p,q}(\delta1(r1,p)\delta2(q,r2)\delta3(p,r3)\delta4(r4,q)\delta5) = \delta1(r1,r3)\delta4(r4,r2)\delta3\delta2\delta5$
4. $dlad_{p,q}(\delta1(r1,p)\delta2(r2,q)\delta3(p,r3)\delta4(q,r4)\delta5) = \delta1(r1,r3)\delta4\delta3\ \delta2(r2,r4)\delta5$

Dlad on MDS descriptors



5. $dlad_{p,q}(\delta 1(p,r1)\delta 2(q, p)\delta 4(r4,q)\delta 5) = \delta 1\delta 4(r4,r1)\delta 2\delta 5$
6. $dlad_{p,q}(\delta 1(p,q)\delta 3(r3,p)\delta 4(q,r4)\delta 5) = \delta 1\delta 4\delta 3(r3,r4)\delta 5$
7. $dlad_{p,q}(\delta 1(r1,p)\delta 2(q,r2)\delta 3(p,q)\delta 5) = \delta 1(r1,r2)\delta 3\delta 2\delta 5$

Dlad on realistic MDS descriptors

1. $dlad_{p,q}(\delta1(p,r1)\delta2(q,r2)\delta3(r3,p)\delta4(r4,q)\delta5) = \delta1\delta4(r4,r2)\delta3(r3,r1)\delta2\delta5$
2. $dlad_{p,q}(\delta1(p,r1)\delta2(r2,q)\delta3(r3,p)\delta4(q,r4)\delta5) = \delta1\delta4\delta3(r3,r1)\delta2(r2,r4)\delta5$
3. $dlad_{p,q}(\delta1(r1,p)\delta2(q,r2)\delta3(p,r3)\delta4(r4,q)\delta5) = \delta1(r1,r3)\delta4(r4,r2)\delta3\delta2\delta5$
4. $dlad_{p,q}(\delta1(r1,p)\delta2(r2,q)\delta3(p,r3)\delta4(q,r4)\delta5) = \delta1(r1,r3)\delta4\delta3\ \delta2(r2,r4)\delta5$
5. $dlad_{p,q}(\delta1(p,r1)\delta2(q,\ p)\delta4(r4,q)\delta5) = \delta1\delta4(r4,r1)\delta2\delta5$
6. $dlad_{p,q}(\delta1(p,q)\delta3(r3,p)\delta4(q,r4)\delta5) = \delta1\delta4\delta3(r3,r4)\delta5$
7. $dlad_{p,q}(\delta1(r1,p)\delta2(q,r2)\delta3(p,q)\delta5) = \delta1(r1,r2)\delta3\delta2\delta5$

TUCS

Gene assembly as a transformation of MDS descriptors

- $ld_p(\ \delta 1\ (q,p)\ (p,r)\ \delta 2\ ) = \delta 1\ (q,r)\ \delta 2$
- $ld_p\ (\ (p,r)\ \delta\ (s,p)\ ) = (s,r)\ \delta$
- $hi_p(\ \delta 1\ (p,q)\ \delta 2\ (-p,r)\ \delta 3\ ) = \delta 1\ \delta 2\ (q,r)\ \delta 3$
- $hi_p\ (\ \delta 1\ (q,p)\ \delta 2\ (r,-p)\ \delta 3\ ) = \delta 1\ (q,r)\ \delta 2\ \delta 3$
- $\mathbf{dlad_{p,q}(\delta 1(p,r1)\delta 2(q,r2)\delta 3(r3,p)\delta 4(r4,q)\delta 5) = \delta 1\delta 4(r4,r2)\delta 3(r3,r1)\delta 2\delta 5}$
- $\mathbf{dlad_{p,q}(\delta 1(p,r1)\delta 2(r2,q)\delta 3(r3,p)\delta 4(q,r4)\delta 5) = \delta 1\delta 4\delta 3(r3,r1)\delta 2(r2,r4)\delta 5}$
- $\mathbf{dlad_{p,q}(\delta 1(r1,p)\delta 2(q,r2)\delta 3(p,r3)\delta 4(r4,q)\delta 5) = \delta 1(r1,r3)\delta 4(r4,r2)\delta 3\delta 2\delta 5}$
- $\mathbf{dlad_{p,q}(\delta 1(r1,p)\delta 2(r2,q)\delta 3(p,r3)\delta 4(q,r4)\delta 5) = \delta 1(r1,r3)\delta 4\delta 3\delta 2(r2,r4)\delta 5}$
- $\mathbf{dlad_{p,q}(\delta 1(p,r1)\delta 2(q,p)\ \delta 4(r4,q)\delta 5) = \delta 1\delta 4(r4,r1)\ \delta 2\delta 5}$
- $\mathbf{dlad_{p,q}(\delta 1(p,\ q)\delta 3(r3,p)\delta 4(q,r4)\delta 5) = \delta 1\delta 4\delta 3(r3,r4)\ \delta 5}$
- $\mathbf{dlad_{p,q}(\delta 1(r1,p)\delta 2(q,r2)\delta 3(p,\ q)\delta 5) = \delta 1(r1,r2)\delta 3\delta 2\delta 5}$

Assembly strategies

A composition ϕ of operations ld, hi, and dlad is an **assembly strategy** for the MDS descriptor u if ϕ(u) is an *assembled MDS descriptor* (i.e. (b,e) or (-e,-b))

Assembly strategies, Example

- Actin I gene in S.nova is δ=(3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)

- **A successful reduction of δ is:**
  - $Ld_4(\delta)$=(3,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
  - $Dlad_{5,6}(ld_4(\delta))$=(3,7)(7,8)(9,e)(-3,-2)(b,2)(8,9)
  - $Ld_7(dlad_{5,6}(ld_4(\delta)))$=(3,8)(9,e)(-3,-2)(b,2)(8,9)
  - $Dlad_{8,9}(ld_7(dlad_{5,6}(ld_4(\delta))))$=(3,e)(-3,-2)(b,2)
  - $Hi_2(dlad_{8,9}(ld_7(dlad_{5,6}(ld_4(\delta)))))$=(3,e)(-3,-b)
  - $Hi_3(hi_2(dlad_{8,9}(ld_7(dlad_{5,6}(ld_4(\delta))))))$=(-e,-b)

Translating the operations to realistic MDS descriptors

From molecular operations based on patterns, folding, and splicing we went to a formal calculus on pairs of letters

The gene assembly process is now a formal rewriting process based on three rules

- The input of the process: a realistic MDS descriptor
- The output of the process: the sequence of rewriting rules that transforms the input into (b,e) or (e,b); any such sequence is called a ***successful reduction***

Advantage: one can reason in formal terms about the process of gene assembly

Note: Although the IESs are ignored in this rewriting calculus, note that they can be easily considered in the operations

- In dealing with certain applications (invariants) we will redefine the operations to keep track of the IESs also

Universality result

Universality:

- Any realistic MDS descriptor has a successful reduction

Consequence:

- Any micronuclear ciliate gene can be successfully assembled using a sequence of ld, hi, and dlad

Universality result

Proof:

- For any realistic MDS descriptor δ, there is an operation applicable to it (thus reducing its length)
- If δ has a positive pointer, then apply hi to δ on that pointer; otherwise, all pointers are negative
- If δ has any alternating direct repeat pattern (…p…q…p…q…), then apply dlad on p an q
- Otherwise, consider a (negative) pointer p of δ such that the distance (in number of pointers) between the two occurrences of p in δ is *minimal*
  - Then the distance must be 0 and so, ld_p is applicable to δ

Representing genes as legal strings

- Abstract from the information whether a pointer is input or output and from MDSs in general
- Represent the gene by the sequence of its pointers only –each such string is a "pointer snapshot" of the gene at a given stage of the assembly
  - Omit the parenthesis and the markers
  - This leads to the so-called "legal" strings –signed double occurrence strings

Example: from MDS descriptors to legal strings

(b,2)(3,4)(2,3)(4,e) → 2 3 4 2 3 4

(3,4)(b,2)(6,e)(5,6)(2,3)(4,5) → 3 4 2 6 5 6 2 3 4 5

(b,2)(2,3)(3,4)(6,e)(4,5)(5,6) → 2 2 3 3 4 6 4 5 5 6

(b,3)(3,4)(6,e)(4,6) → 3 3 4 6 4 6

(b,4)(6,e)(4,6) → 4 6 4 6

Representing genes as legal strings

- Note the degree of simplification at this level: from the nucleotide sequence of the gene we only represent at this level the order in which pointers (but not markers) occur along the sequence

Generic: $M_3(3,4)M_4(4,5)M_6(6,7)M_5(5,6)M_7(7,8)M_9(9,e)\overline{M_2(3,2)M_1(b,2)}M_8(8,9)$
MDS descriptor: (3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(3,2)(b,2)(8,9)
Legal string: 3 4 4 5 6 7 5 6 7 8 9 3 2 2 8 9 denoted also as
3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

Signed double occurrence strings

- Alphabet (of pointers): {2, 3,…, k, -2, -3,…, -k}
- *Legal strings: for any pointer i, u contains either 0 or 2 occurrences of letters from the set {i,-i}*

- Example:
  - 3 -2 4 2 -4 3 is a legal string
  - 3 -2 4 2 3 is not a legal string because it only has one occurrence from the set {4,-4}
  - The MDS descriptor associated to actinI gene in S.nova is
    - (3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
    - Its corresponding legal string is
      - 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

LD, HI, DLAD as operations on legal strings

- **LD for MDS descriptors:**
  - $ld_p(\ \delta1(q,p)\ (p,r)\ \delta2) = \delta1(q,r)\ \delta2$
- **ld_p for legal strings:**
  - **u p p v → uv**
- for any pointer p and any strings u,v

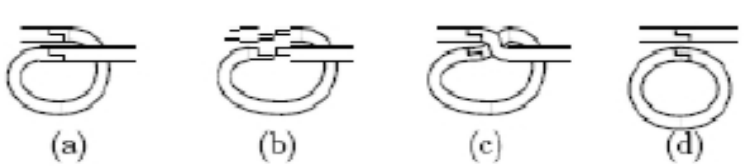


Fig. 1. Illustration of the ld molecular operation.

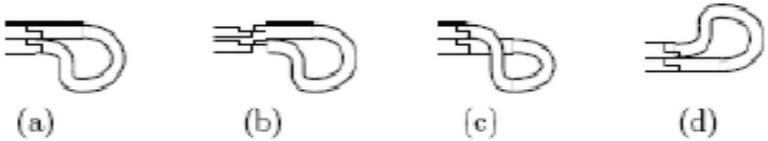


Fig. 2. Illustration of the hi molecular operation.

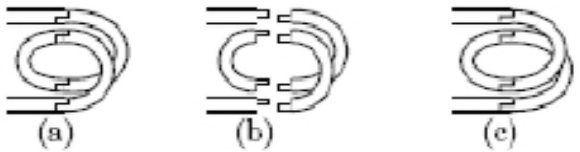


Fig. 3. Illustration of the dlad molecular operation.

LD, HI, DLAD as operations on legal strings

- **HI for MDS descriptors:**
  - $hi_p(\ \delta1(p,q)\ \delta2(p,r)\ \delta3) = \delta1\ \delta2\ (q,r)\ \delta3$
  - $hi_p(\ \delta1(q,p)\ \delta2(r,p)\ \delta3) = \delta1(q,r)\ \delta2\ \delta3$

- **Hi_p for legal strings:**
  - **u p v -p w → u -*v w***
  - *for any pointer p and any strings u,v, w,where -(q1q2…qn)=-qn… -q2 -q1*

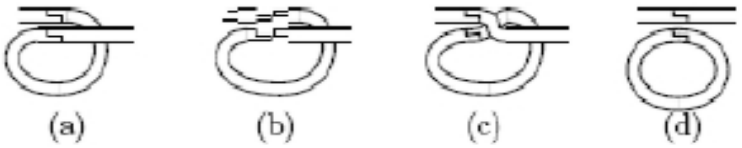


Fig. 1. Illustration of the ld molecular operation.

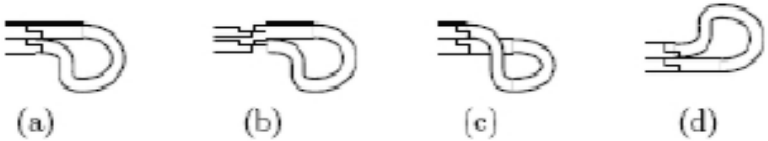


Fig. 2. Illustration of the hi molecular operation.

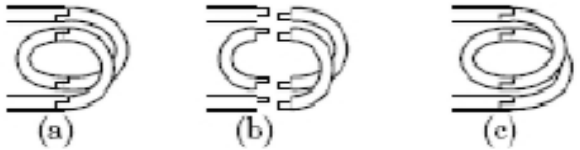


Fig. 3. Illustration of the dlad molecular operation.

LD, HI, DLAD as operations on legal strings

DLAD for MDS descriptors:

- $dlad_{p,q}(\delta_1(p,r_1)\delta_2(q,r_2)\delta_3(r_3,p)\delta_4(r_4,q)\delta_5) = \delta_1\delta_4(r_4,r_2)\delta_3(r_3,r_1)\delta_2\delta_5$
- $dlad_{p,q}(\delta_1(p,r_1)\delta_2(r_2,q)\delta_3(r_3,p)\delta_4(q,r_4)\delta_5) = \delta_1\delta_4\delta_3(r_3,r_1)\delta_2(r_2,r_4)\delta_5$
- $dlad_{p,q}(\delta_1(r_1,p)\delta_2(q,r_2)\delta_3(p,r_3)\delta_4(r_4,q)\delta_5) = \delta_1(r_1,r_3)\delta_4(r_4,r_2)\delta_3\delta_2\delta_5$
- $dlad_{p,q}(\delta_1(r_1,p)\delta_2(r_2,q)\delta_3(p,r_3)\delta_4(q,r_4)\delta_5) = \delta_1(r_1,r_3)\delta_4\delta_3\delta_2(r_2,r_4)\delta_5$
- $dlad_{p,q}(\delta_1(p,r_1)\delta_2(q,p)\ \delta_4(r_4,q)\delta_5) = \delta_1\delta_4(r_4,r_1)\ \delta_2\delta_5$
- $dlad_{p,q}(\delta_1(p,\ q)\delta_3(r_3,p)\delta_4(q,r_4)\delta_5) = \delta_1\delta_4\delta_3(r_3,r_4)\ \delta_5$
- $dlad_{p,q}(\delta_1(r_1,p)\delta_2(q,r_2)\delta_3(p,\ q)\delta_5) = \delta_1(r_1,r_2)\delta_3\delta_2\delta_5$

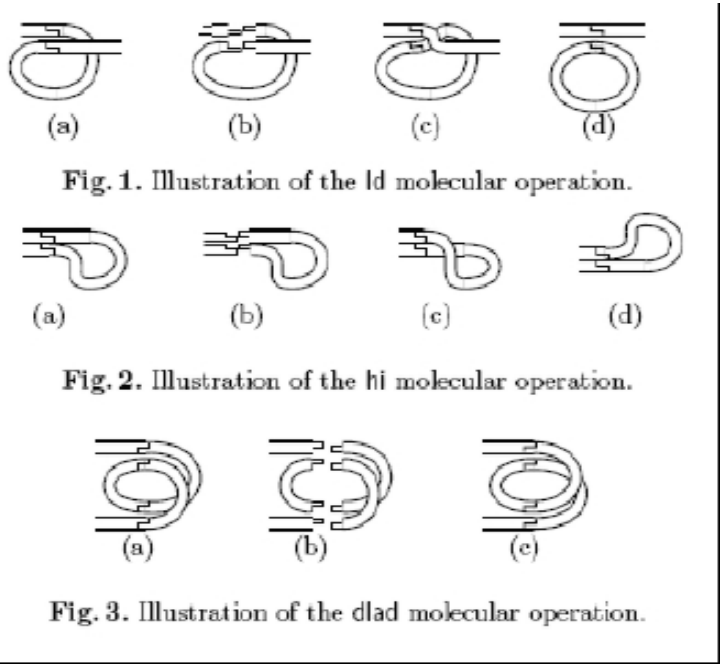


Fig. 1. Illustration of the ld molecular operation.

Fig. 2. Illustration of the hi molecular operation.

Fig. 3. Illustration of the dlad molecular operation.

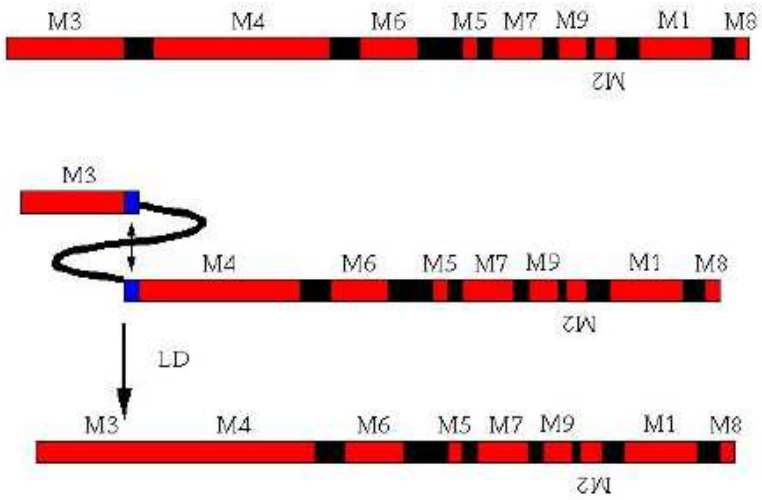
$dlad_{p,q}$ for legal strings:

u_1 p u_2 q u_3 p u_4 q u_5 → u_1 u_4 u_3 u_2 u_5

for any pointers p,q and any strings u_1, u_2, u_3, u_4, u_5

Example: assembling gene *actinI in S.Nova*

Step 1: u p p v → uv



u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

$snr_4(u)$ =3 5 6 7 5 6 7 8 9 -3 -2 2 8

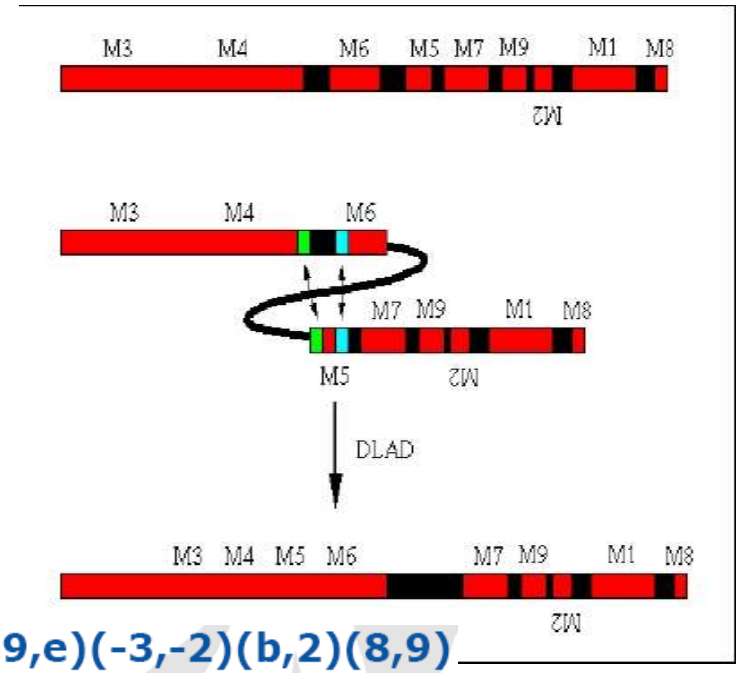
δ=(3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)

$Id_4(\delta)$=(3,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)

Example: assembling gene *actinI in S.Nova*

Step 2: u1 p u2 q u3 p u4 q u5 → u1 u4 u3 u2 u5

u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9

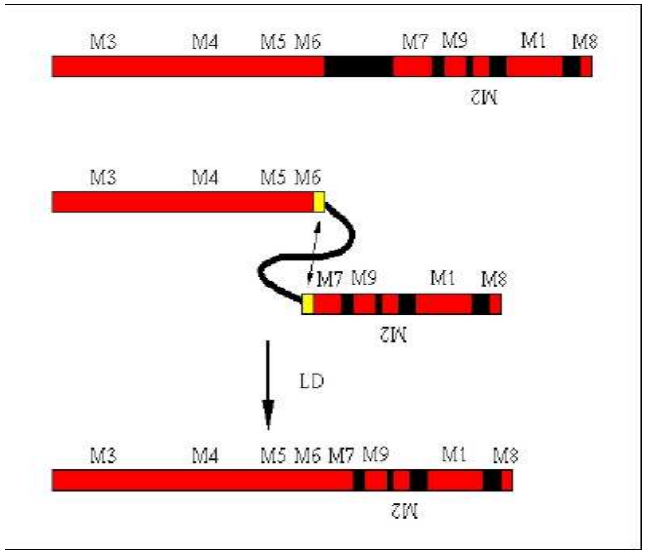


δ_2=(3,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
$dlad_{5,6}(\delta_2)$=(3,7)(7,8)(9,e)(-3,-2)(b,2)(8,9)

Example: assembling gene *actinI in S.Nova*

Step 3: u p p v → uv

u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9

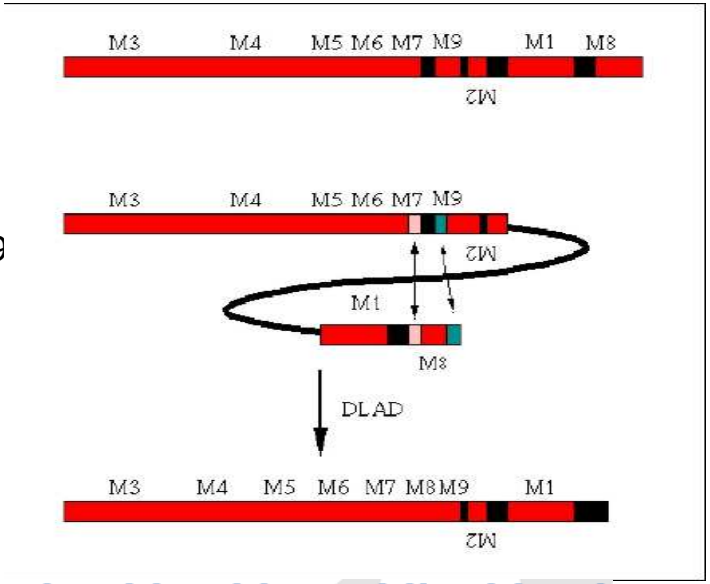


δ_3=(3,7)(7,8)(9,e)(-3,-2)(b,2)(8,9)
$Id_7(\delta_3)$=(3,8)(9,e)(-3,-2)(b,2)(8,9)

Example: assembling gene *actinI in S.Nova*

Step 4: u1 p u2 q u3 p u4 q u5 → u1 u4 u3 u2 u5

u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9
$sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))$= 3 -3 -2 2

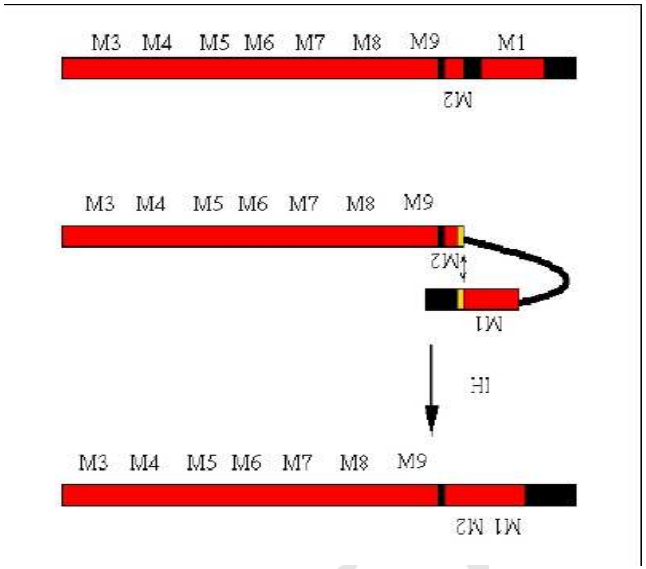


δ_4=(3,8)(9,e)(-3,-2)(b,2)(8,9)
$dlad_{8,9}(\delta_4)$=(3,e)(-3,-2)(b,2)

Example: assembling gene *actinI in S.Nova*

Step 5: u p v -p w → u -*(v)* *w*

u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9
$sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))$= 3 -3 -2 2
$spr_{-2}(sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u)))))$= 3 -3

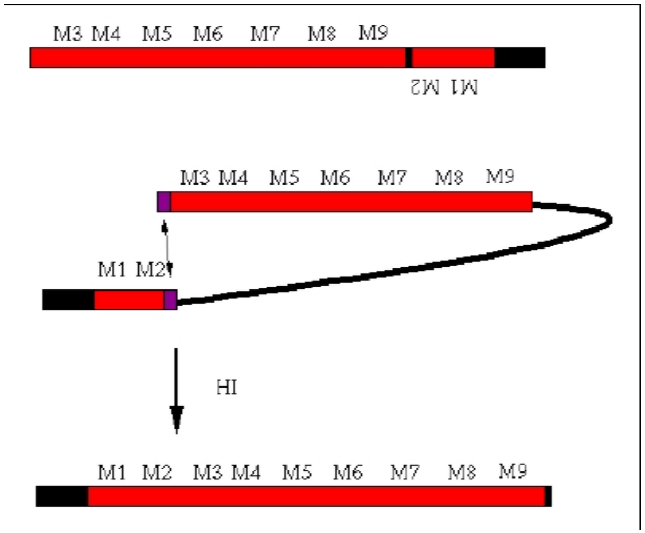


δ_5=(3,e)(-3,-2)(b,2)
$hi_{-2}(\delta_5)$=(3,e)(-3,-b)

Example: assembling gene *actinI in S.Nova*

Step 6: u p v -p w → u -*(v)* *w*

u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9
$sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))$= 3 -3 -2 2
$spr_{-2}(sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u)))))$= 3 -3
$spr_3(spr_{-2}(sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))))=\lambda$



δ_6=(3,e)(-3,-b)
$hi_3(\delta_6)$=(-e,-b)

TUCS

Comments

- Strings are useful because their structure is simpler than that of MDS descriptors and the operations LD, HI, DLAD are much easier to define and to work with on strings rather than on MDS descriptors and on permutations

- Useful in applications such as study of micronuclear gene patterns that can be assembled when some of the operations are not available

Beyond strings

- Example: legal strings u=2 3 4 4 3 -2 and v=3 4 4 2 -2 3 have *very similar behavior under the three rewriting rules*
  - spr2◦snr3◦snr4 is a reduction strategy for both of them
  - snr-3◦snr-4◦spr2 is a reduction strategy of u, snr3◦snr4◦spr2 is a reduction strategy of v
  - the pointers in u and v are in the same overlap relation!

- **Idea:** consider only the overlap relation between pointers
  - This leads to signed overlap graphs
- Two pointers p,q overlap in u if u = …p'…q'…p''…q''…, where p',p''∈{p,-p} and q',q''∈{q,-q}

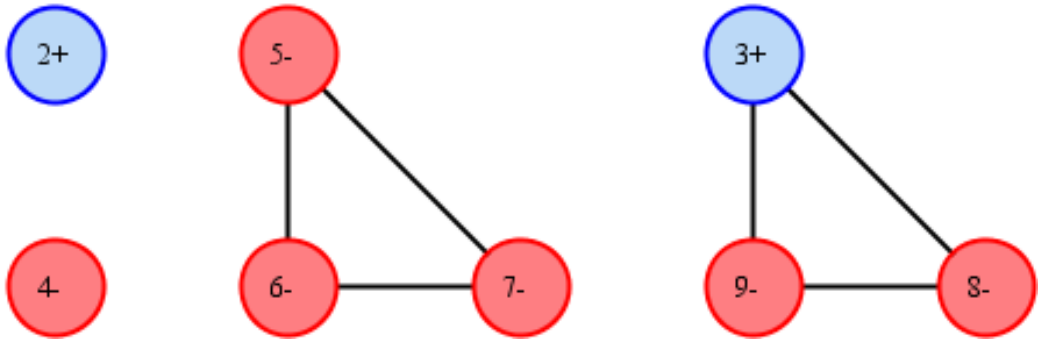
Signed overlap graphs

- For each pointer in legal string u we associate a vertex in the graph G_u – the vertex is positive/negative if pointer is positive/negative
- A pointer p is positive in u if both p and -p occur in u and it is negative otherwise
- There is an edge between p and q in G_u iff p and q overlap in u

Signed overlap graphs: Example

Generic: $M_3(3,4)M_4(4,5)M_6(6,7)M_5(5,6)M_7(7,8)M_9(9,e)\overline{M_2(3,2)}M_1(b,2)M_8(8,9)$
MDS descriptor: (3,4)(4,5)(6,7)(5,$\underline{6}$)($\underline{7}$,8)(9,e)$\overline{(3,2)}$(b,2)(8,9)
Legal string: 3 4 4 5 6 7 5 6 7 8 9 $\underline{3}$ $\underline{2}$ 2 8 9 denoted also as
3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

Overlap graph:



Graph structure

- The graph structure of a gene keeps only the essential information about the gene structure
  - It is not obvious that the graph still has any connection to the gene
  - It is proved that, e.g., if an operation is applicable to the gene, then the corresponding operation is applicable to the graph
  - A reverse result can also be proved
- The most useful one in, e.g., studying parallelism in gene assembly

Levels of abstraction

- The MDS descriptor representation is the most fateful to the biological representation of a gene
  - Two genes have the same MDS descriptors if and only if they have the same number of MDSs in the same order
- Two different MDS descriptors may have the same associated legal string
  - The string level is more abstract
  - However, for an MDS M and its string u_M, an operation is applicable to M if and only if the corresponding operation is applicable to u_M
  - The string level is equivalent to the MDS descriptor level as far as gene assembly is concerned
- Two different strings may have the same associated graph
  - The graph level is more abstract
  - The graph level is equivalent to the other as far as successful gene assemblies are concerned