Special course in Computer Science: Molecular Computing

Lecture 13: Gene assembly as graph reduction. Complexity and universality of gene assembly

Vladimir Rogojin
Department of IT, Åbo Akademi
http://combio.abo.fi/teaching/special

Fall 2013

TUCS

LD as operation on MDS descriptors and legal strings

- **LD for MDS descriptors:**
  - $ld_p(\delta1(q,p)\ (p,r)\ \delta2) = \delta1(q,r)\ \delta2$

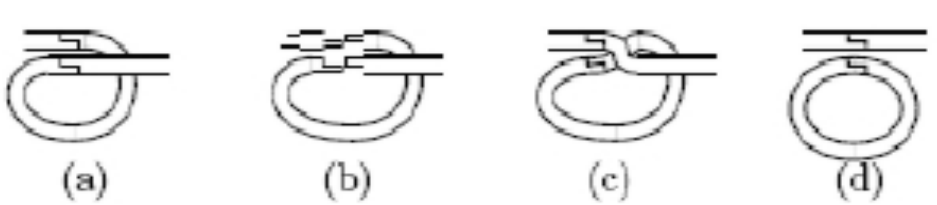


Fig. 1. Illustration of the ld molecular operation.

Fig. 2. Illustration of the hi molecular operation.

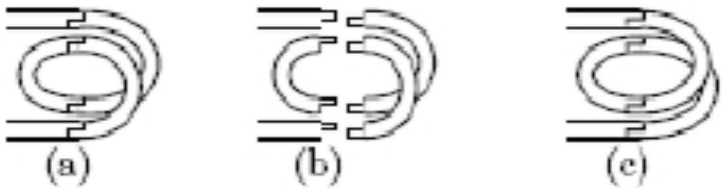


Fig. 3. Illustration of the dlad molecular operation.

- **ld_p for legal strings:**
- **u p p v → uv**
- for any pointer p and any strings u,v

HI as operation on MDS descriptors and legal strings

- **HI for 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*

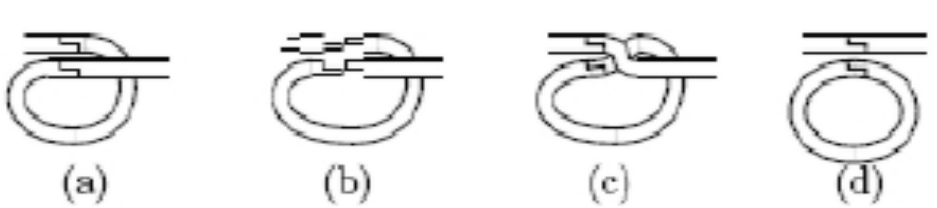


Fig. 1. Illustration of the ld molecular operation.

Fig. 2. Illustration of the hi molecular operation.

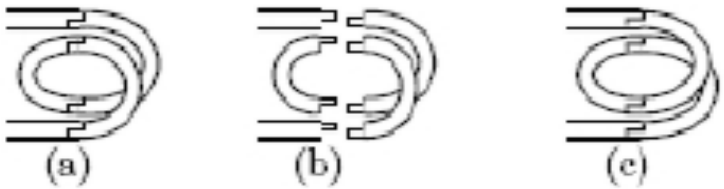


Fig. 3. Illustration of the dlad molecular operation.

- **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*

TUCS

DLAD as operation on MDS descriptors and legal strings

DLAD for MDS descriptors:

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

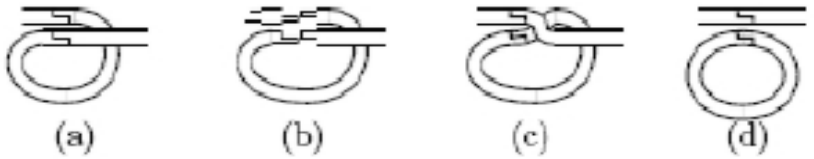


Fig. 1. Illustration of the ld molecular operation.

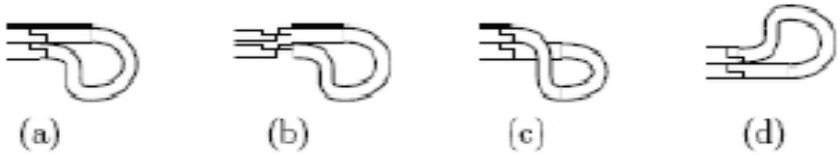


Fig. 2. Illustration of the hi molecular operation.

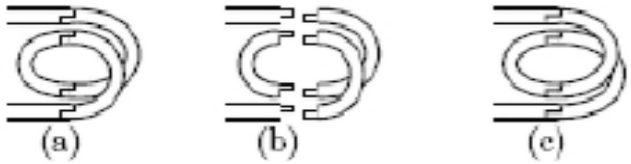


Fig. 3. Illustration of the dlad molecular operation.

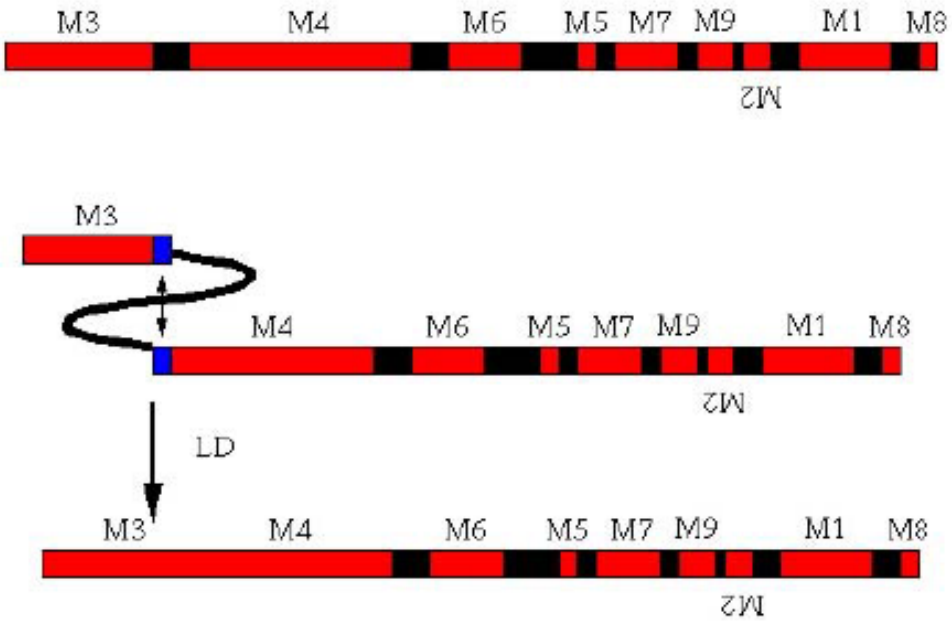
$dlad_{p,q}$ for legal strings:

$u_1\ p\ u_2\ q\ u_3\ p\ u_4\ q\ u_5 \rightarrow u_1\ u_4\ u_3\ u_2\ u_5$

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)

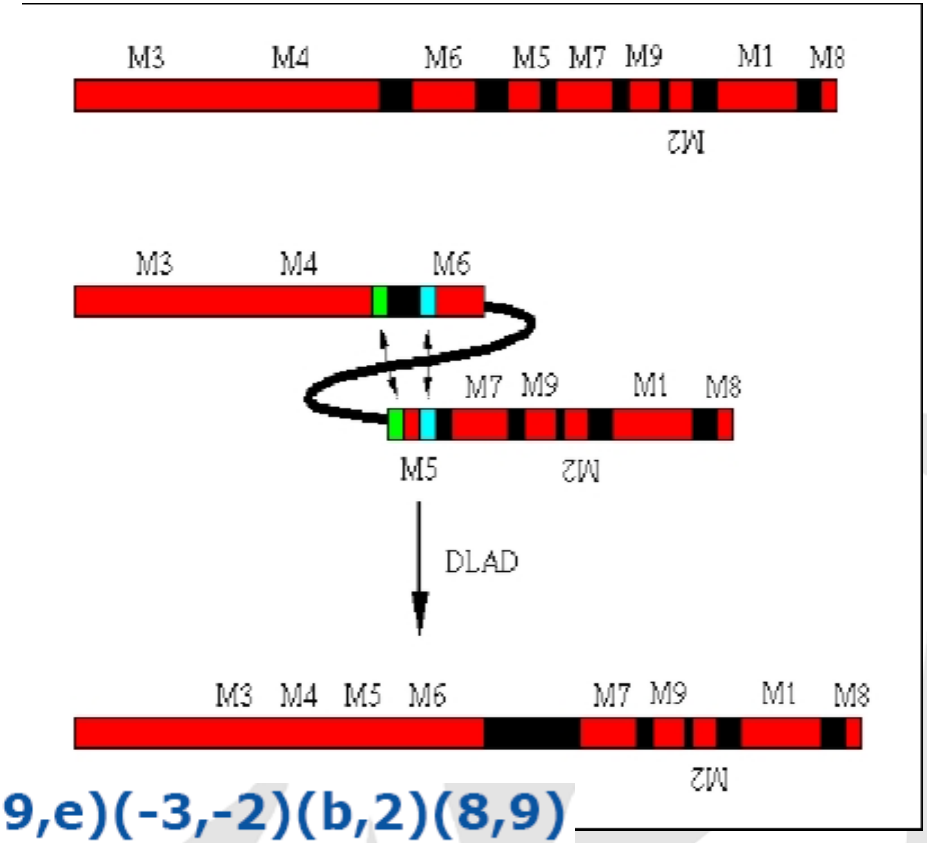
Example: assembling gene *actinI* *in S.Nova*

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*

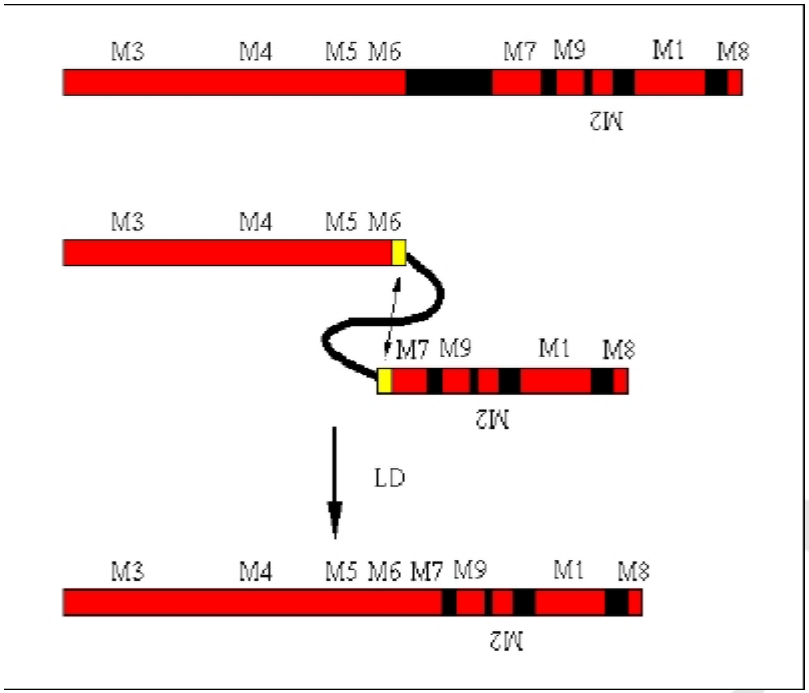
Step 3: u p p v → uv

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

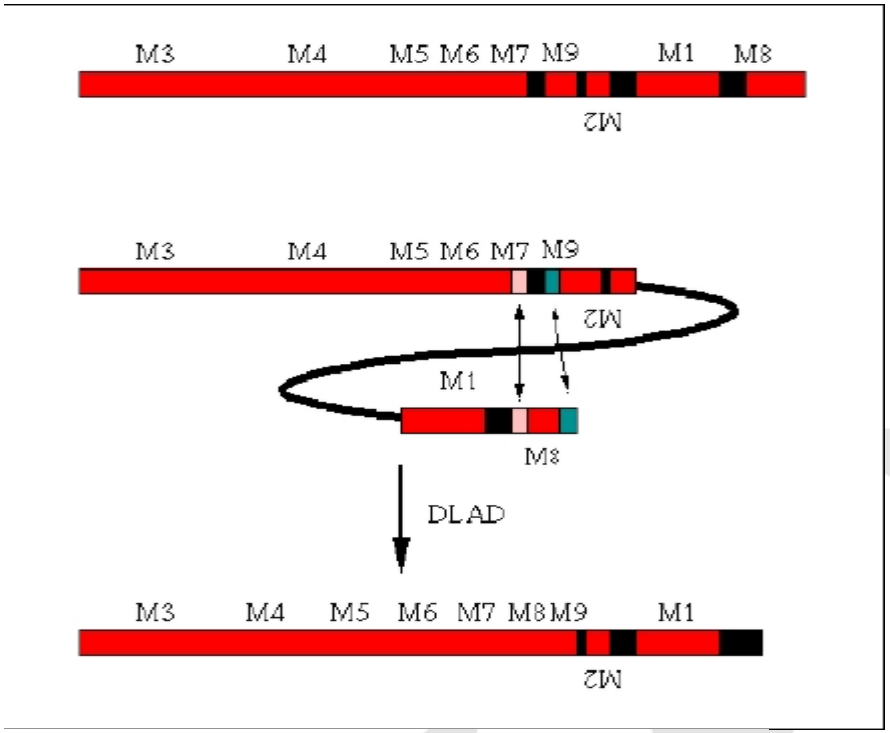
u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

$snr_4(u)$=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

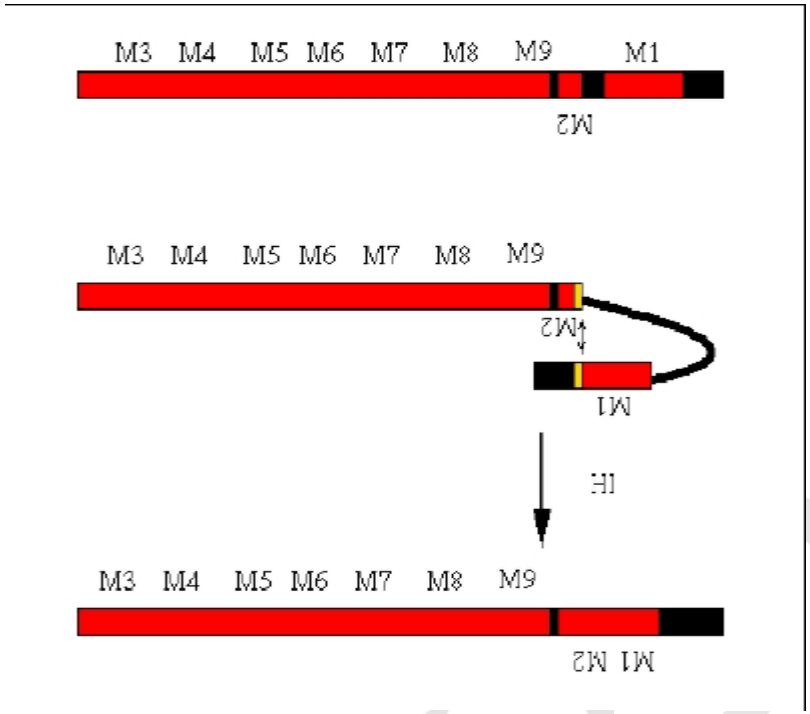
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9

$sdr_{5,6}(snr_4(u))$= 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

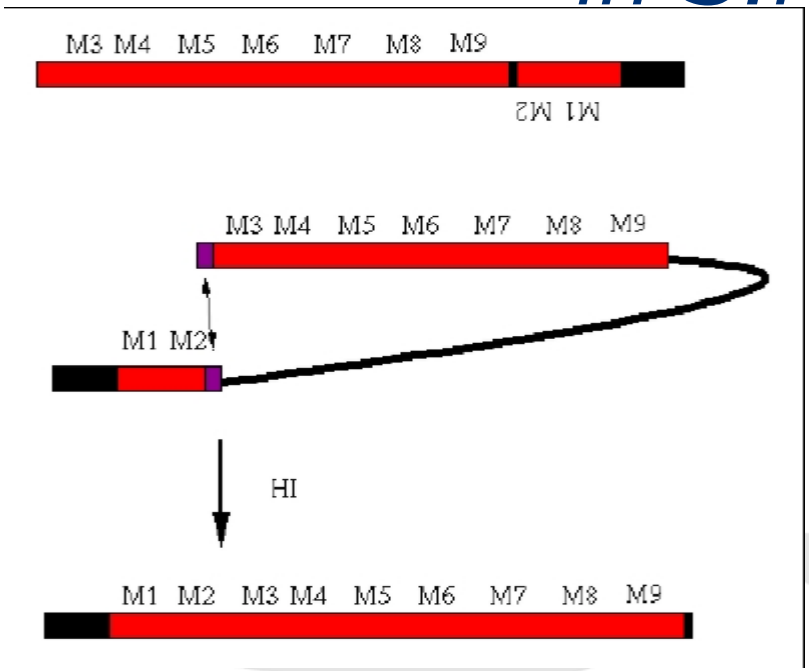
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9

$snr_7(sdr_{5,6}(snr_4(u)))$= 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

$spr_3(spr_{-2}(sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))))$=λ



δ_6=(3,e)(-3,-b)

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

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
  - snr3◦snr4◦spr2 is a reduction strategy of u and 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}

TUCS

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

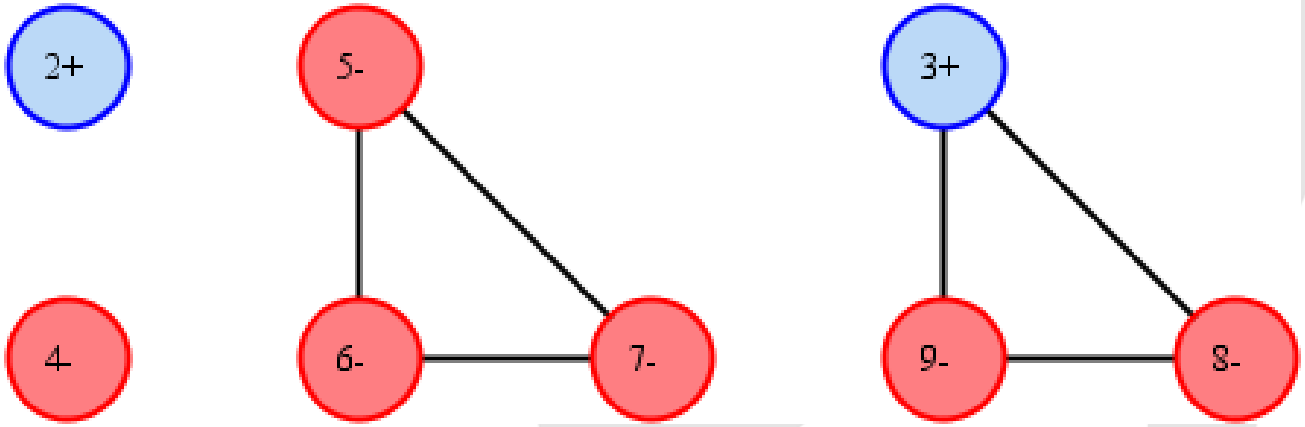
Signed overlap graphs: Example

M3 M4 M6 M5 M7 M9 M2 M1 M8

MDS descriptor: (3,4)(4,5)(6,7)(5,6)(7,8)(9,e)$\overline{(3,2)}$(b,2)(8,9)

Legal string: 3 4 4 5 6 7 5 6 7 8 9 $\overline{3}$ $\overline{2}$ 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

Formalizing the operations for overlap graphs

- For each of the operations {ld, hi, dlad}/ {snr,spr,sdr} we define its correspondent transformation rule for overlap graphs
  - The rules will be called gnr, gpr, gdr, in analogy with the string rewriting rules
  - Each rule on overlap graphs will remove one or two vertices and do some local transformations (on the neighborhoods)

From snr,spr,sdr to graph rules: **GNR**

LD for MDS descriptors:

- $ld_p(\ \delta_1\ (q,p)\ (p,r)\ \delta_2\ ) = \delta_1\ (q,r)\ \delta_2$
- $ld_p\ (\ (p,r)\ \delta\ (s,p)\ ) = (s,r)\ \delta$

SNR for legal strings:

- u p p v → uv

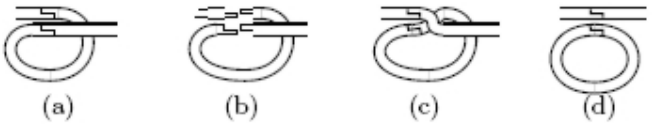


Fig. 1. Illustration of the ld molecular operation.

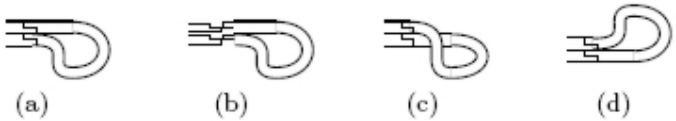


Fig. 2. Illustration of the hi molecular operation.

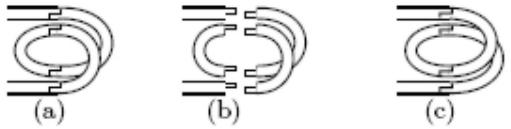


Fig. 3. Illustration of the dlad molecular operation.

The correspondent of $\mathbf{snr_p}$ for signed overlap graphs is the transformation rule $\mathbf{gnr_p}$:

$$\mathbf{gnr_p}(G)=G-\{p\},$$

for any *isolated negative* vertex p

If G is the overlap graph of u, then $\mathbf{gnr_p}(G)$ is the overlap graph of $\mathbf{snr_p}(u)$

(assuming $\mathbf{snr_p}$ is applicable to u)

From snr,spr,sdr to graph rules: **GPR**

HI for MDS descriptors:

- $hi_p(\ \delta_1\ (p,q)\ \delta_2\ (p,r)\ \delta_3\ ) = \delta_1\ -\delta_2\ (q,r)\ \delta_3$
- $hi_p\ (\ \delta_1\ (q,p)\ \delta_2\ (r,p)\ \delta_3\ ) = \delta_1\ (q,r)\ -\delta_2\ \delta_3$

SPR for legal strings:

- u p v -p w → u -v w

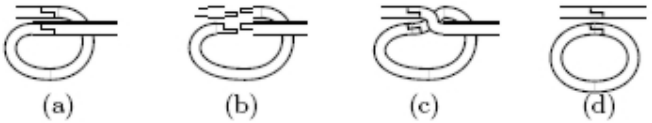


Fig. 1. Illustration of the ld molecular operation.

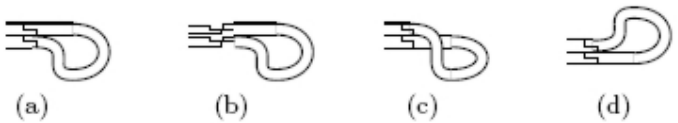


Fig. 2. Illustration of the hi molecular operation.

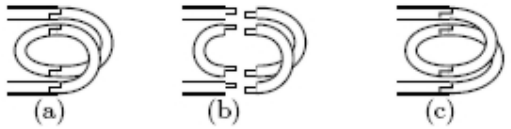


Fig. 3. Illustration of the dlad molecular operation.

The correspondent of $\mathbf{spr_p}$ for signed overlap graphs is the transformation rule $\mathbf{gpr_p}$:

$$\mathbf{gpr_p}(G)=loc_p(G)-\{p\},$$

for any *positive* vertex p, where loc_p is the local complementation at p

If G is the overlap graph of u, then $\mathbf{gpr_p}(G)$ is the overlap graph of $\mathbf{spr_p}(u)$.

From snr,spr,sdr to graph rules: **GDR**

SDR for legal strings:

- $u_1\ p\ u_2\ q\ u_3\ p\ u_4\ q\ u_5 \rightarrow u_1\ u_4\ u_3\ u_2\ u_5$

The correspondent of $\mathbf{sdr_{p,q}}$ for signed overlap graphs is the transformation rule $\mathbf{gdr_{p,q}}$.

The rule is applicable to G if p,q are negative adjacent vertices in G:

> **$gdr_{p,q}$**(G) is the graph obtained by complementing the edge relationship between $N_G(p)$ and $N_G(q)$, then removing p and q.

In other words, the status of a pair (x,y), for $x,y \in G-\{p,q\}$ will change if and only if

- $x \in N_G(p)-N_G(q)$, $y \in N_G(q)$
- $x \in N_G(q)-N_G(p)$, $y \in N_G(p)$
- $x \in N_G(p) \cap N_G(q)$, $y \in (N_G(p)-N_G(q)) \cup (N_G(q)-N_G(p))$

Fig. 1. Illustration of the ld molecular operation.

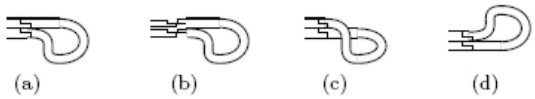


Fig. 2. Illustration of the hi molecular operation.

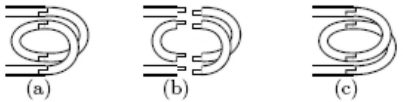


Fig. 3. Illustration of the dlad molecular operation.

DLAD for MDS descriptors:

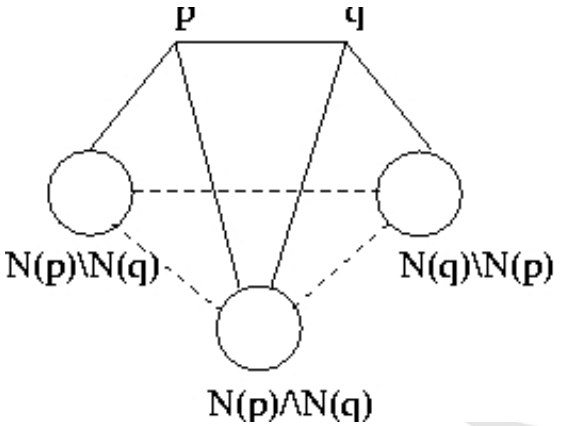
- $dlad_{p,q}(\delta_1(p,r_1)\delta_2(q,r_2)\delta_3(r_3,p)\delta_4(r_4,q)\delta_5) = \delta_1\delta_4(r_4,r_2)\delta_3(r_3,r_1)\delta_2\delta_5$
- $dlad_{p,q}(\delta_1(p,r_1)\delta_2(r_2,q)\delta_3(r_3,p)\delta_4(q,r_4)\delta_5) = \delta_1\delta_4\delta_3(r_3,r_1)\delta_2(r_2,r_4)\delta_5$
- $dlad_{p,q}(\delta_1(r_1,p)\delta_2(q,r_2)\delta_3(p,r_3)\delta_4(r_4,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

From snr,spr,sdr to graph rules: **GDR**

The correspondent of $\mathbf{sdr}_{p,q}$ for signed overlap graphs is the transformation rule $\mathbf{gdr}_{p,q}$.

The rule is applicable to G if p,q are negative adjacent vertices in G:

> $\mathbf{gdr}_{p,q}(G)$ is the graph obtained by complementing the edge relationship between $N_G(p)$ and $N_G(q)$, then removing p and q.

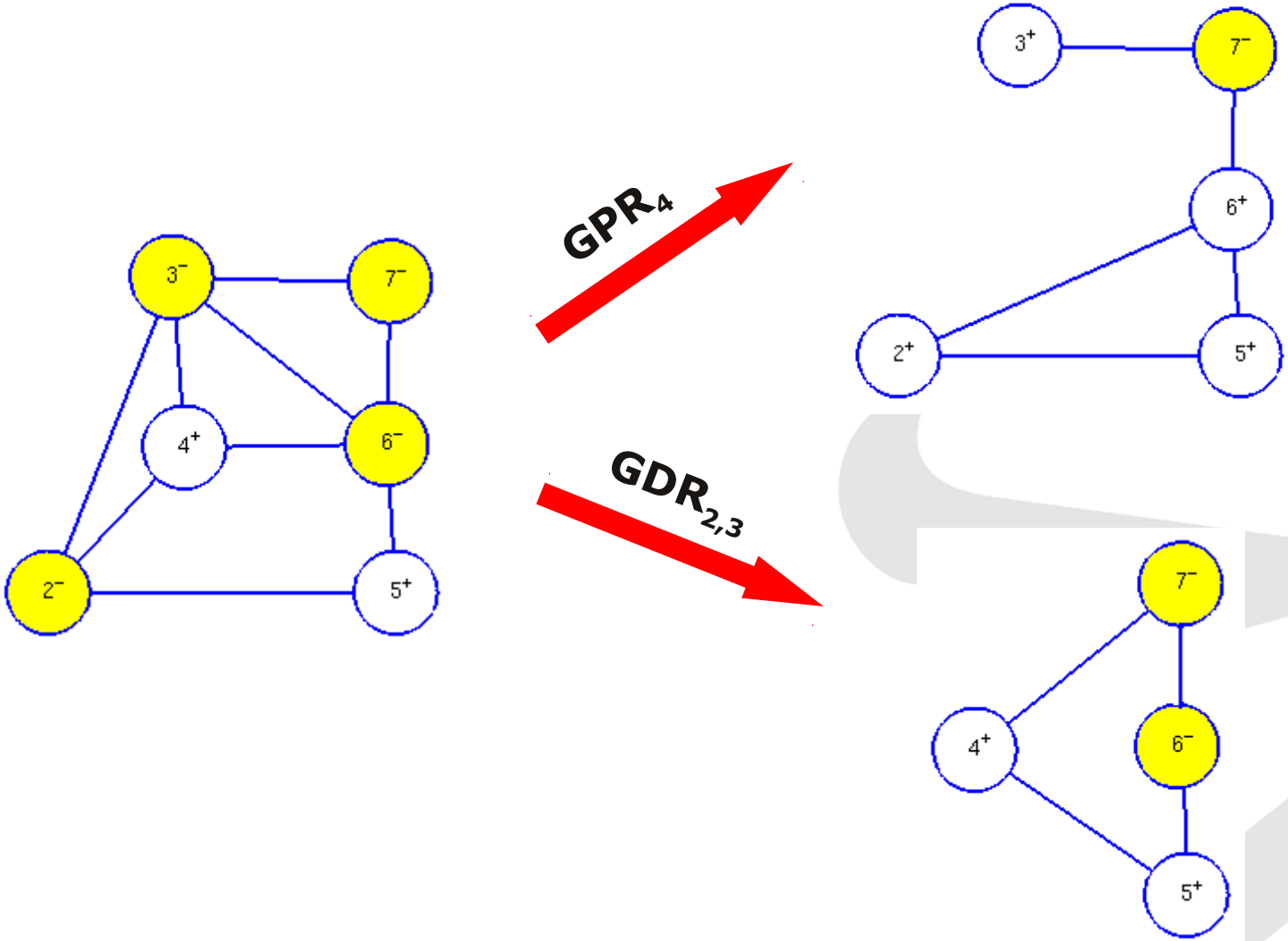


In other words, the status of a pair (x,y), for $x,y \in G-\{p,q\}$ will change if and only if

- $x \in N_G(p)-N_G(q)$, $y \in N_G(q)$
- $x \in N_G(q)-N_G(p)$, $y \in N_G(p)$
- $x \in N_G(p) \cap N_G(q)$, $y \in (N_G(p)-N_G(q)) \cup (N_G(q)-N_G(p))$

If G is the overlap graph of u, then $\mathbf{gdr}_{p,q}(G)$ is the overlap graph of $\mathbf{sdr}_{p,q}(u)$.

Examples: GPR and GDR

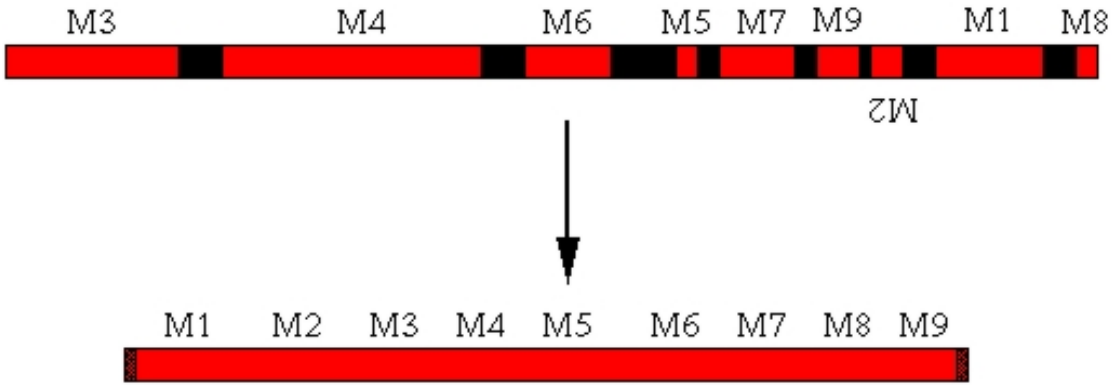


Reduction strategies

- A composition ϕ of graph transformation rules **gnr**, **gpr**, and **gdr** is a **reduction strategy** for the signed overlap graph G if $\phi(G)=\varnothing$

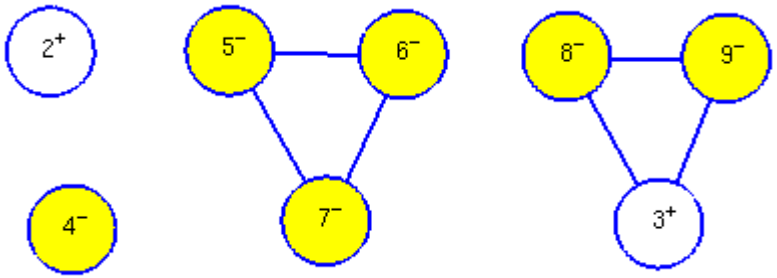
Example: actinI gene in S.nova

The MIC/MAC form
of gene *actin I* in *S.Nova*



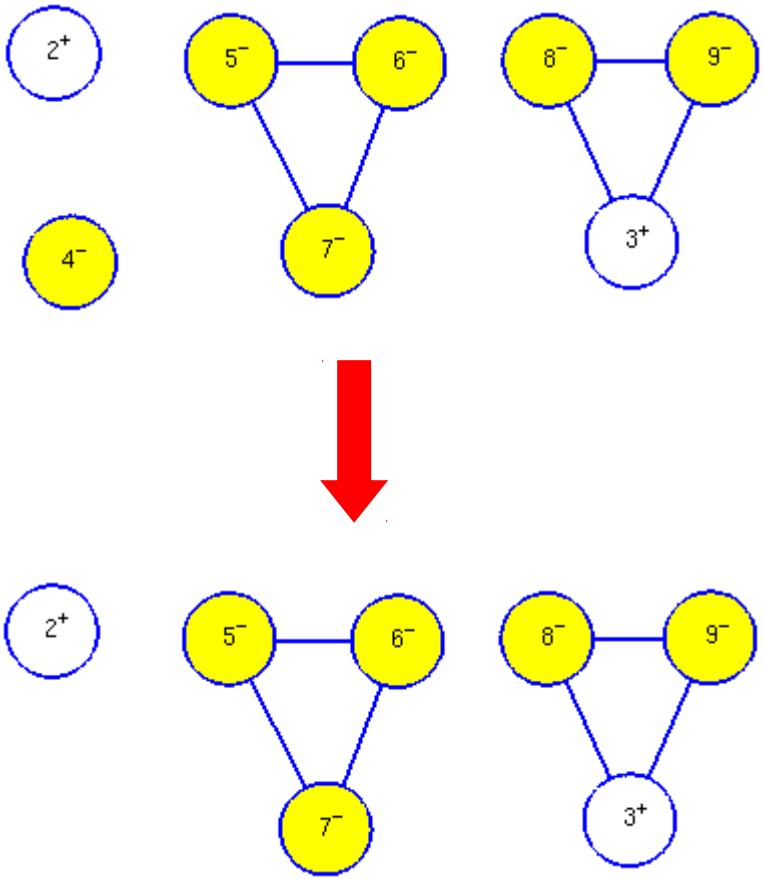
MDS descriptor: **δ=**(3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
Legal string: u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

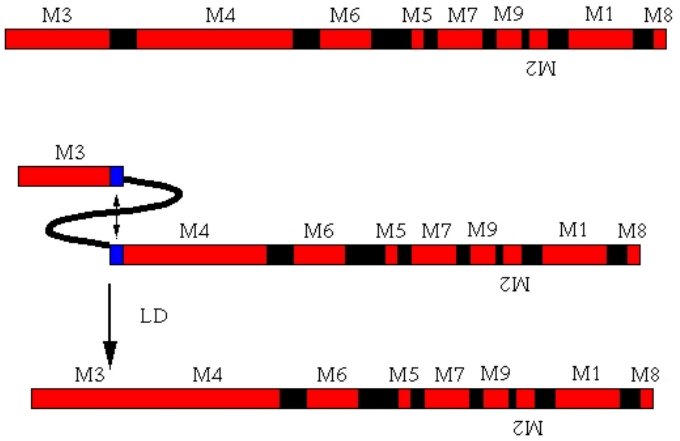
Overlap graph:



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

Step 1: **GNR_4**





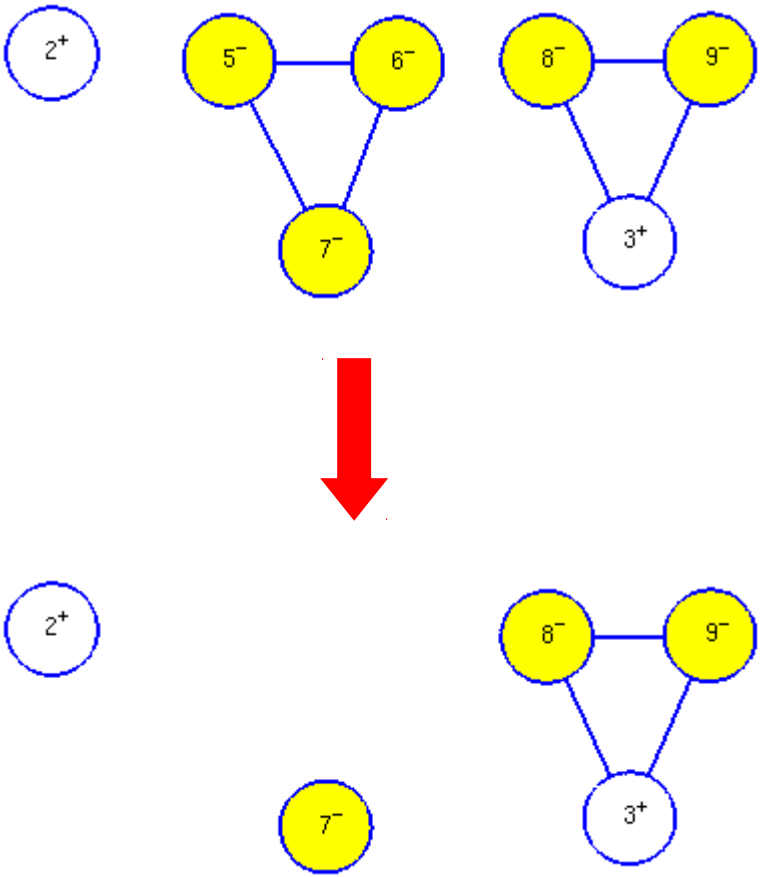
δ=(3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
Id_4(δ)=(3,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)

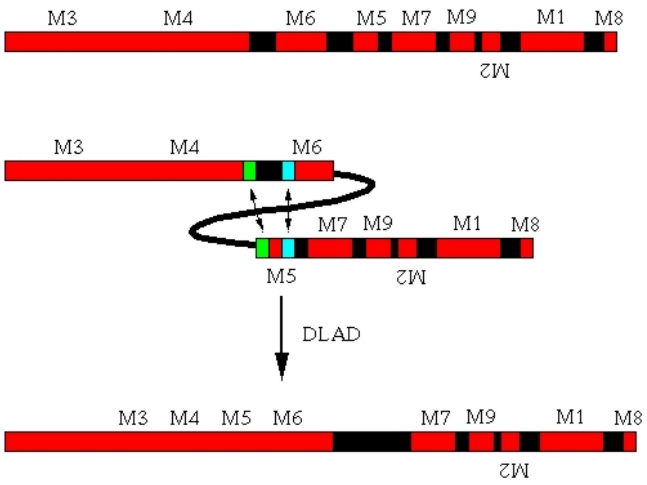
u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

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

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

Step 2: **$GDR_{5,6}$**





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)

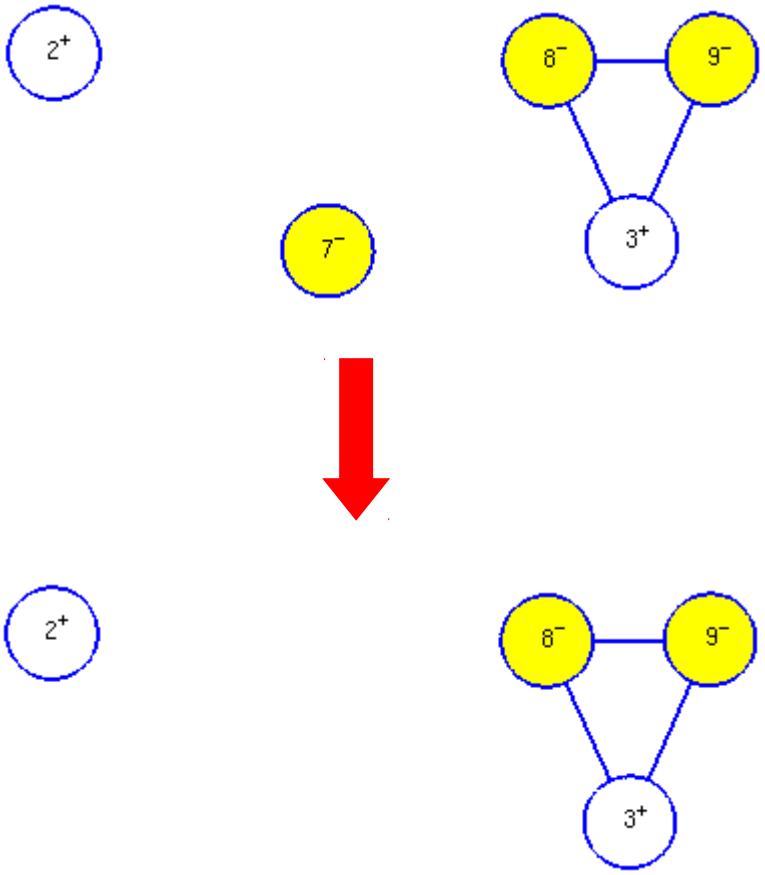
u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

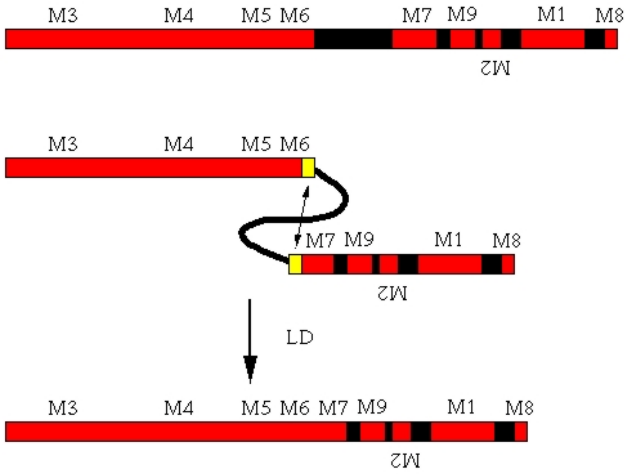
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9

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

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

Step 3: **GNR_7**





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)

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

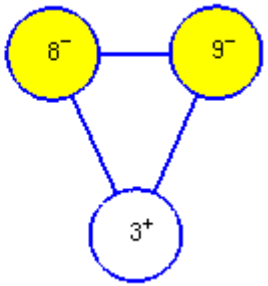
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9

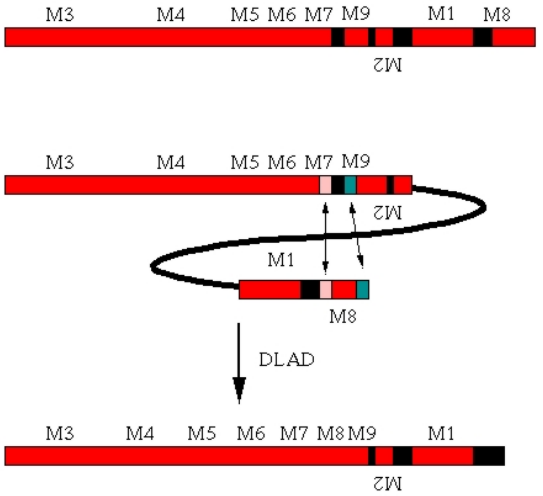
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9

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

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

Step 4: **$GDR_{8,9}$**





δ_4=(3,8)(9,e)(-3,-2)(b,2)(8,9)

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

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

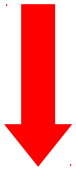
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9

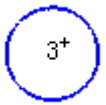
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9

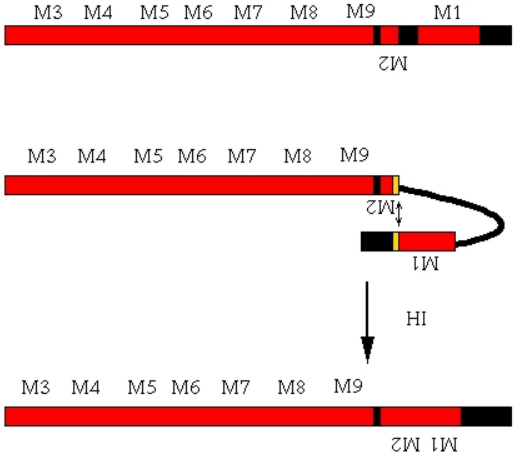
$sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))$= 3 -3 -2 2

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

Step 5: **GPR_2**







δ_5=(3,e)(-3,-2)(b,2)

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

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

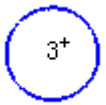
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9

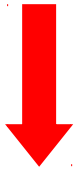
$sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))$= 3 -3 -2 2

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

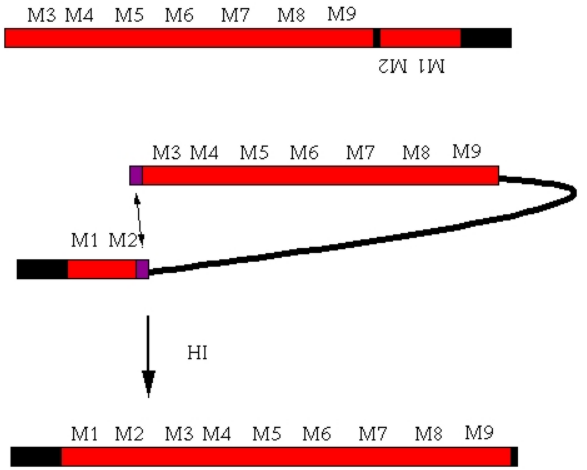
Example: assembling gene *actin I* in *S.Nova*

Step 6: **GPR_3**





∅



δ_6=(3,e)(-3,-b)

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

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

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

Reducing legal strings vs. reducing overlap graphs

- u is a legal string, G_u is the corresponding overlap graph
  - If a rewriting rule f (snr, spr, sdr) is applicable to u, then the corresponding graph transformation rule F (gnr, gpr, gdr, resp.) is applicable to G_u. Moreover, $F(G_u)=G_{f(u)}$
  - The reverse is also true for GPR and GDR, but not for GNR – consider the example 2 3 3 2
  - **Result**: For any graph reduction strategy for G_u, there is an equivalent string reduction strategy for u (in which some of the **snr** rules could be done in a different order)
- **Conclusion:** as far as the gene assembly process (successful reduction strategy) is concerned, legal strings and overlap graphs are "equivalent"!
- In many cases, it is easier (or more elegant) to work with graphs – they ignore the linear structure of the string

Complexity measures, simplest gene patterns

One may consider a *computational type of complexity giving an 'objective'* complexity measure for ciliate genes - it shows how much that gene has evolved and how involved its assembling is

- In this way, the simplest genes are those than can be assembled using ld only, because this is the 'simplest' operation
- Indeed, this corresponds to the intuition – the genes than can be assembled using ld have the MDSs in the orthodox order, or circularly shifted

TUCS

Complexity of gene assembly

- Gene Assembly - a computational process consisting of a sequence of ld, hi, dlad
- The *complexity of the process – consider the number of the operations* and/or the complexity of the operations
- The *complexity of the gene – the minimal complexity of an assembly* for that gene
- *Similarity measure – assembly processes with similar complexity*

Types of complexity measures

- Compare the operations used in the assembly - some of them are more complex than the others. This leads to considering the genes that can be assembled using a given subset of operations.

- Compare the folds involved in the operations applied in some assembly - some of them are more complex than the others. This leads to considering *simple operations.*

- Consider the operations to be applied in parallel and the number of parallel steps in an assembly strategy. This leads to the parallel complexity investigation.

Complexity: types of operations

- **Idea 1:** Define the complexity as the number of operations needed in an assembly
  - The operations are considered here to have the same complexity

- **Idea 2:** Extend the previous idea by considering different complexities for different types of operations; consider weights for each operation
  - Clearly, ld is the simplest of the operations, while dlad is the most complex one. Thus: ld < hi < dlad

Complexity measures: Example

Actin I gene in *Sterkiella nova:*

v = 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9.

V = 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (snr_4)

3 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (spr_2)

3 5 6 7 5 6 7 8 9 -3 8 9 → (spr_3)

-9 -8 -7 -6 -5 -7 -6 -5 8 9 → (spr_9)

-8 5 6 7 5 6 7 8 → (spr_8)

-7 -6 -5 -7 -6 -5 → $(sdr_{7,6})$

-5 -5 → (snr_5)

Λ.

Complexity measures: Example

V = 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (snr_4)

3 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (spr_2)

3 5 6 7 5 6 7 8 9 -3 8 9 → (spr_3)

-9 -8 -7 -6 -5 -7 -6 -5 8 9 → (spr_9)

-8 5 6 7 5 6 7 8 → (spr_8)

-7 -6 -5 -7 -6 -5 → ($sdr_{7,6}$)

-5 -5 → (snr_5)

Λ.

- 2 snr, 4 spr, 1 sdr operations. In total: 7 op.
- The complexity of this reduction in the above sense, is 2 · csnr + 4 · cspr+1 · csdr, where csnr, cspr, csdr are the weights associated to snr, spr, sdr.

Complexity measures: Example

- Another reduction for v:

V = 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (snr_4)

3 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (spr_2)

3 5 6 7 5 6 7 8 9 -3 8 9 → ($sdr_{5,6}$)

3 7 7 8 9 -3 8 9 → ($sdr_{8,9}$)

3 7 7 -3 → (snr_7)

3 -3 → (spr_3)

Λ.

- 2 snr, 2 spr, 2 sdr operations. In total: 6 op.
- The complexity of this reduction is 2 · csnr +2 · cspr +2 · csdr.
- Recall: the complexity of the former reduction was 2 · csnr + 4 · cspr +1 · csdr.

Complexity measures: Example

V = 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (snr_4)

3 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (spr_2)

3 5 6 7 5 6 7 8 9 -3 8 9 → (spr_3)

-9 -8 -7 -6 -5 -7 -6 -5 8 9 → (spr_9)

-8 5 6 7 5 6 7 8 → (spr_8)

-7 -6 -5 -7 -6 -5 → ($sdr_{7,6}$)

-5 -5 → (snr_5)

Λ.

V = 3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (snr_4)

3 5 6 7 5 6 7 8 9 -3 -2 2 8 9 → (spr_2)

3 5 6 7 5 6 7 8 9 -3 8 9 → ($sdr_{5,6}$)

3 7 7 8 9 -3 8 9 → ($sdr_{8,9}$)

3 7 7 -3 → (snr_7)

3 -3 → (spr_3)

Λ.

- The latter reduction is easier, if only operations are counted: the former reduction uses 7 and the latter uses only 6 operations.
- On the other hand, the latter reduction is harder, if csdr > 2 · cspr.

Complexity: types of patterns

- One can go even deeper and consider that the weight depends also on the type of pattern to which is applied: e.g., if $u = u_1 p u_2 - p u_3$, then $cspr_p = |u_2|$ and if $u = u_1 p u_2 q u_3 p u_4 q u_5$, then $csdr_{p,q} = 2 \cdot (|u_2| + |u_4|)$.
  - In this case, the complexity of the first reduction is 22, while that of the second is 0!
  - The second strategy only uses 'simple folds', while the first one uses long folds

Complexity classes and similarity measure

- The complexity classes give as usual a measure of similarity
  - Two genes may be considered ‘similar’ from a computational point of view if they can be assembled using the same subset of operations
  - Question: what is the set of micronuclear genes that can be assembled using a given subset of operations ?
  - The answer to this question defines the complexity classes

Complexity defined through subsets of operations

- We consider all possible subsets of {ld, hi, dlad} and characterize those micronuclear gene patterns that can be assembled using only those operations
- Each of the characterizations can be stated in any of the three levels of the intramolecular model: MDS descriptors, strings, or graphs
- In each case, we choose here the level giving the 'simplest' way of stating the result

TUCS

Patterns that can be assembled using ld only

- An MDS descriptor can be assembled using ld *only if and only if it can* be obtained from an orthodox sequence of MDSs through cyclic shifts

  - $(i,\overline{i+1})(i+1,\overline{i+2}) \ldots (k,\overline{e})(\overline{b},2) \ldots (\overline{i-1},i)$, or
  - $(\overline{i},\overline{i-1}) \ldots (\overline{2},\overline{b})(\overline{e},\overline{k}) \ldots (\overline{i+2},\overline{i+1})(\overline{i+1},\overline{i})$, $1 \leq i \leq k$

- A signed overlap graph can be assembled using gnr *only if and only* if it consists of isolated negative vertices only (it is a discrete negative graph)

Patterns that can be assembled using ld and hi only

- A signed overlap graph can be assembled using gnr *and* *gpr only if and* only if every non-trivial (more than two nodes) connected component contains at least one positive vertex

Patterns that can be assembled using ld and dlad only

- An MDS-descriptor can be assembled using ld, dlad only if and only if either none of its pairs, or all of them are signed.
- A signed double occurrence string can be assembled using snr *and sdr only if and only if all the pointers are negative.*
- A signed overlap graph can be assembled using gnr *and gdr only if and* only if all the vertices are negative.

Patterns that can be assembled using hi only - example

- Example: M1M4-M2M3, with the legal string u = 2 4 -3 -2 3 4.

- A successful assembly using spr only:

  $V = 2\ 4\ -3\ -2\ 3\ 4 \rightarrow (spr_2)\ 3\ -4\ 3\ 4 \rightarrow (spr_4)\ -3\ 3 \rightarrow (spr_3)\ \Lambda$.

- An unsuccessful one:

  $V \rightarrow (spr_3)\ 2\ 4\ 2\ 4$,

- but the resulting legal string is not successful in Spr.

Patterns that can be assembled using hi only - example

Another example: -M1M2M4M3, with the legal string

v = -2 2 3 4 3 4,

is not successful in Spr, because of the legal substring
3 4 3 4 with no positive pointers.

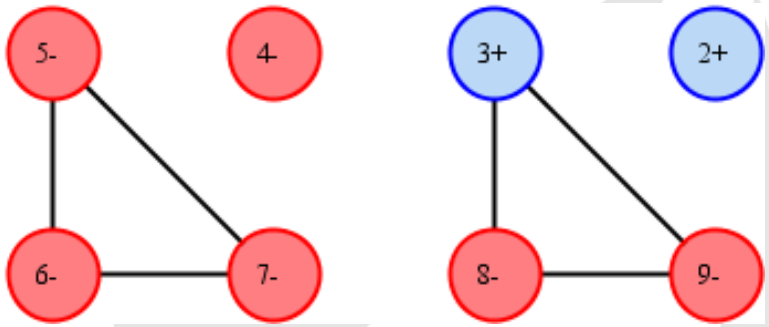
Patterns that can be assembled using ld, hi and dlad – universality result

- *Universality result: Any MDS descriptor can be assembled using a* sequence of ld, hi, dlad.

- Note: Some genes may need all three operations to be assembled - see Actin I in O.nova,

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

  - ld is certainly needed: pointer 4
  - so it is hi: pointers 2 and 3
  - dlad is also needed since the associated graph has one non-trivial negative component



Complexity measures: length of the interval

- We have concentrated so far on the type of operations that are applied in a gene assembly
- However, two applications of the same operations may have different complexities, depending on the intervals involved in the operation
- The simplest possible intervals involved in the operations give rise to the *simple applications of our operations*

TUCS

Simple Id

- The Id_p operation:

$$Id_p(\delta 1(q,p)(p,r)\delta 2) = \delta 1(q,r)\delta 2,$$
$$Id_p((p,m1)(m2, p)) = (m2,m1).$$

- Id is always simple: there is only one IES between p.
  - A boundary application of Id is always the last step in a circular assembly

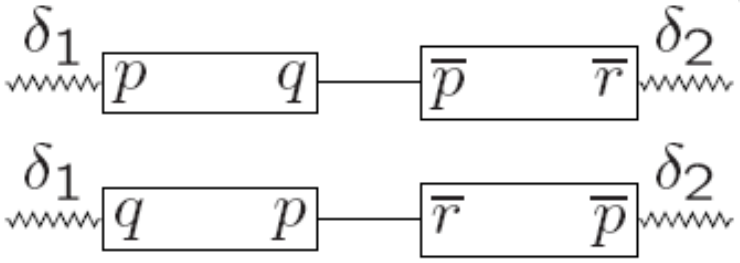
Simple hi

The hi_p operation:

$$hi_p(\delta1(p,q)\delta2(-p,-r)\delta3) = \delta1\ \text{-}\delta2\ (-q,-r)\ \delta3,$$

$$hi_p(\delta1(q,p)\delta2(-r,-p)\delta3) = \delta1\ (q,r)\ \text{-}\delta2\ \delta3,$$

Simple hi_p: there is only one IES between p and -p:



Simple hi

The hi_p operation:

$$hi_p(\delta 1(p,q)\delta 2(-p,-r)\delta 3) = \delta 1\ -\delta 2\ (-q,-r)\ \delta 3,$$

$$hi_p(\delta 1(q,p)\delta 2(-r,-p)\delta 3) = \delta 1\ (q,r)\ -\delta 2\ \delta 3,$$

Simple hi_p operation:

$$shi_p(\delta 1(p,q)(-p,-r)\delta 3) = \delta 1\ (-q,-r)\ \delta 3,$$

$$shi_p(\delta 1(q,p)(-r,-p)\delta 3) = \delta 1\ (q,r)\ \delta 3,$$

Effect: p is removed from the pattern and at most one pointer is inverted, when ship is applied

Simple dlad

- The $dlad_{p,q}$ operation:

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

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

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

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

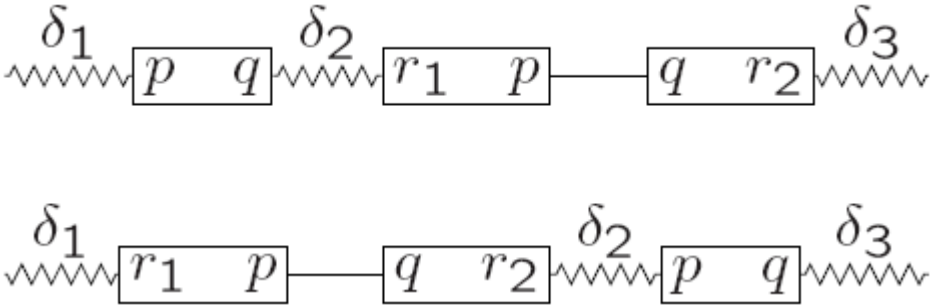
$dlad_{p,q}(\delta1(p,r1)\delta2(q,\ p)\delta4(r4,q)\delta5)=\delta1\delta4(r4,\ r1)\delta2\delta5$

$dlad_{p,q}(\delta1(p,q)\delta3(r3,p)\delta4(q,r4)\delta5)=\delta1\delta4\delta3(r3,r4)\delta5$

$dlad_{p,q}(\delta1(r1,p)\delta2(q,r2)\delta3(p,\ q)\delta5)=\delta1(r1,r2)\delta3\delta2\delta5$

Simple $dlad_{p,q}$: there is exactly one IES in the two sequences between p and q:

Simple dlad



$sdlad_{p,q}(\delta1(p,q)\delta3(r3,p)(q,r4)\delta5)=\delta1\delta3\ (r3,r4)\delta5$

$sdlad_{p,q}(\delta1(r1,p)(q,r2)\delta3(p,\ q)\delta5)=\delta1(r1,r2)\delta3\delta5$

Effect: p and q are simply removed from the pattern, when $sdlad_{p,q}$ is applied

TUCS

Simple operations: Example

(3, 4) (4, 5) (6, 7) (5, 6) (7, 8) (9, e) (-3, -2) (b, 2) (8, 9) → (ld_4)

(3, 5) (6, 7) (5, 6) (7, 8) (9, e) (-3, -2) (b, 2) (8, 9) → ($dlad_{5,6}$)

(3, 7) (7, 8) (9, e) (-3, -2) (b, 2) (8, 9) → (ld_7)

(3, 8) (9, e) (-3, -2) (b, 2) (8, 9) → ($dlad_{8,9}$)

(3, e) (-3, -2) (b, 2) → (hi_2)

(3, e) (-3, -b) → (hi_3)

(-e,-b)

Simple operations are not universal

- The set of our simple operations is NOT universal - there are MDS descriptors / legal strings that cannot be assembled using simple operations only

Example: δ = (-2,-b)(4,e)(3,4)(2,3) – no simple operation is applicable to δ

- Question: are there any ciliate micronuclear patterns that cannot be assembled using simple operations ?

TUCS

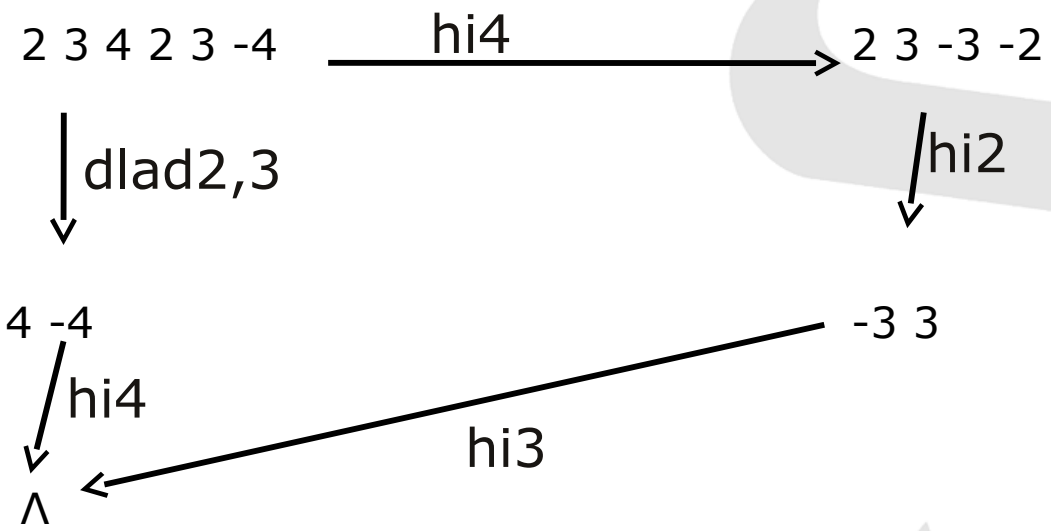
A conjecture on simple operations

- **Conjecture:** The ciliates only use simple operations in the gene assembly process
- The conjecture has been verified for *all existing experimental data*
- It makes sense from a biological point of view
- Justifies the current high interest in the simple operations and their patterns

TUCS

General assembly strategies

- Gene assembly process is non-deterministic:
  - General model may assemble the same gene pattern with strategies of different lengths and even with different types of operations
  - Example:



Simple assembly strategies

- Simple assembly process is also non-deterministic
  - However, for a gene pattern numbers of using of each of simple ld, hi and dlad operations are preserved from one to another assembly strategy
  - Formaly: All strategies applicable to the same gene pattern have the same complexity (Cld, Chi, Cdlad), where Cld is the number of ld operations, Chi is the number of hi operations, Cdlad is the number of dlad operations.
- Simple assembly process is confluent
  - All strategies applicable to a gene pattern either assemble it to the MAC gene, or all of them fail to do that
  - Immediate consequence: one can decide in quadratic time whether a MIC gene pattern may be assembled to MAC gene – just apply any simple strategy