Special course in Computer Science: Molecular Computing

# Lecture 9: Gene assembly as graph reduction. Parallelism and complexity. Computing with gene assembly

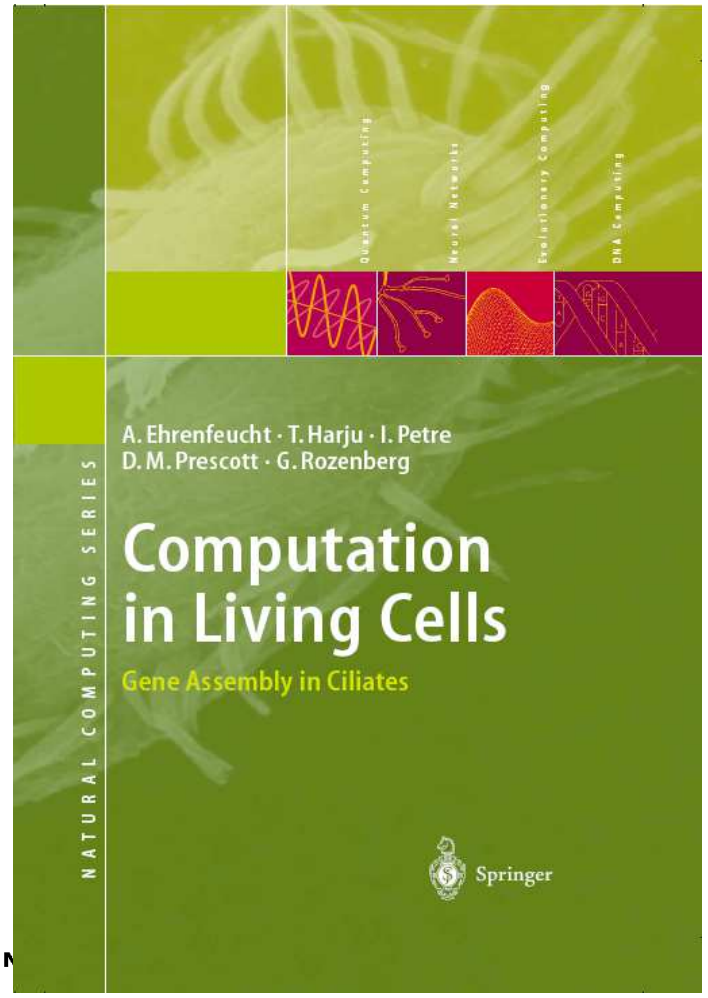
Vladimir Rogojin

Department of CS, Abo Akademi

# Gene assembly in ciliates – course book

Lectures 6 – 9, recommended reading:

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



# LD as operation on MDS descriptors and legal strings

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

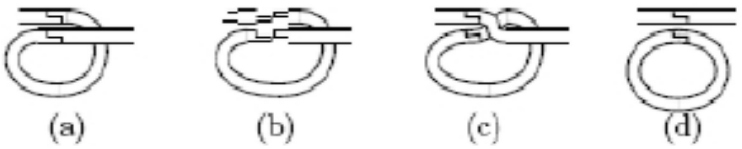



Fig. 1. Illustration of the ld molecular operation.

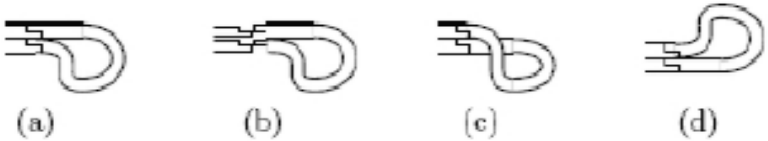



Fig. 2. Illustration of the hi molecular operation.

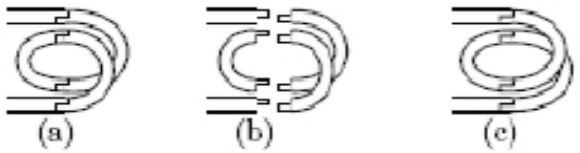



Fig. 3. Illustration of the dlad molecular operation.

- **$ld_p$ 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*

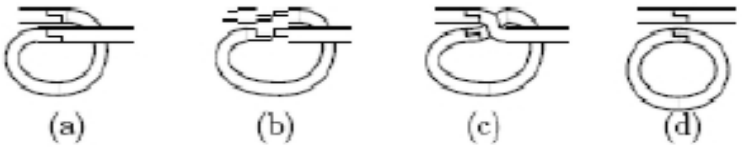



Fig. 1. Illustration of the ld molecular operation.

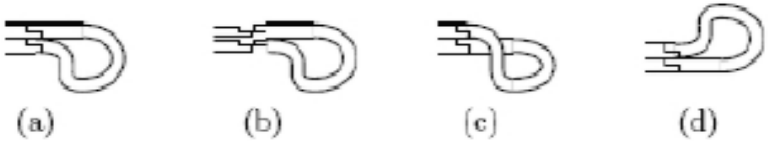



Fig. 2. Illustration of the hi molecular operation.

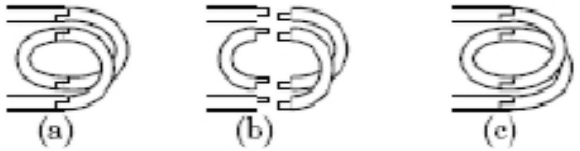



Fig. 3. Illustration of the dlad molecular operation.

- **$Hi_p$ 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*

# DLAD as operation on MDS descriptors and legal strings

DLAD for MDS descriptors:

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

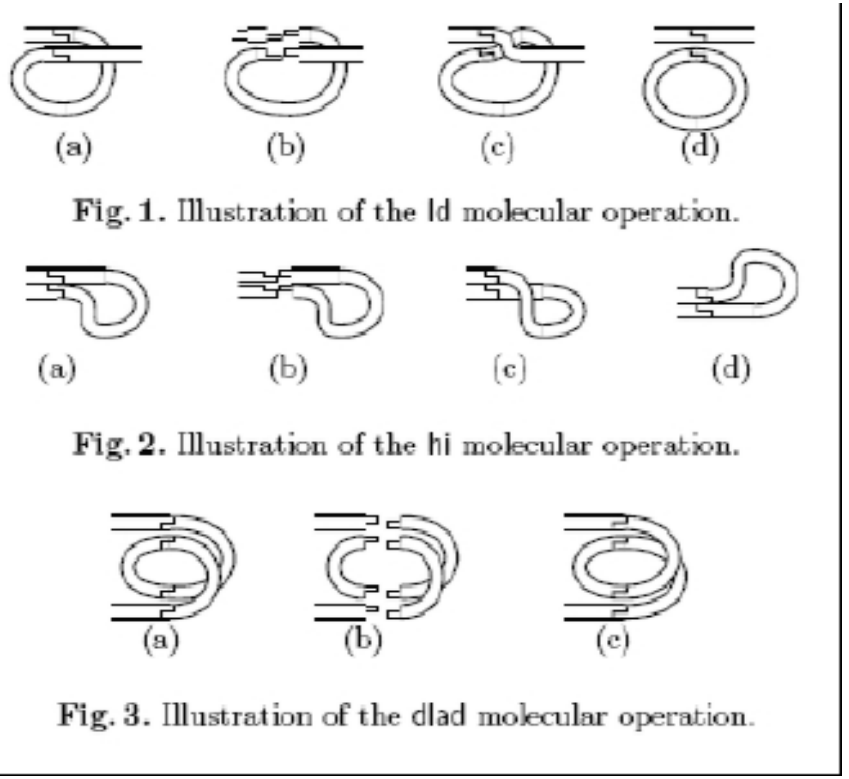



Fig. 1. Illustration of the ld molecular operation.

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 $u_1$, $u_2$, $u_3$, $u_4$, $u_5$

# 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}

# Signed overlap graphs

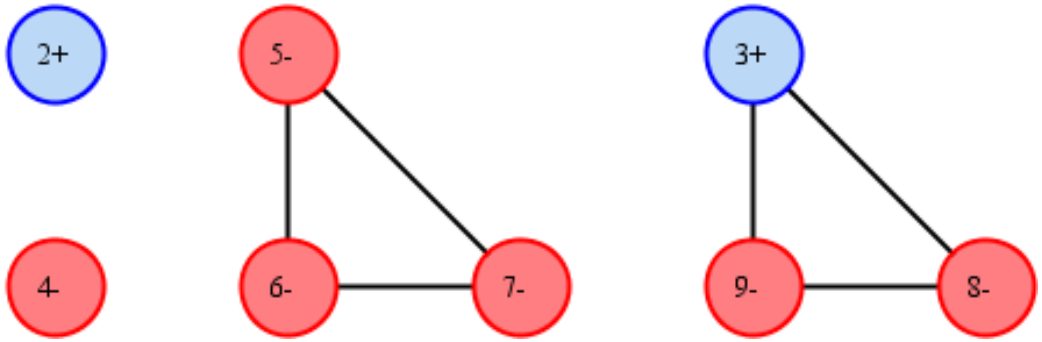
- For each pointer in legal string u we associate a vertex in the graph $G_u$ – the vertex is positive/negative if pointer is positive/negative
- A pointer p is positive in u if both p and -p occur in u and it is negative otherwise
- There is an edge between p and q in $G_u$ iff p and q overlap in u

# Signed overlap graphs: Example


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

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

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)

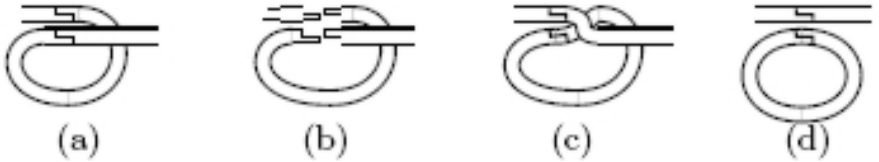



**Fig. 1.** Illustration of the ld molecular operation.

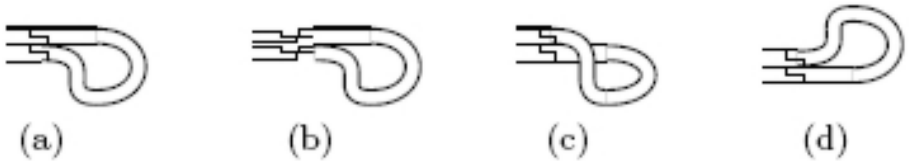



**Fig. 2.** Illustration of the hi molecular operation.

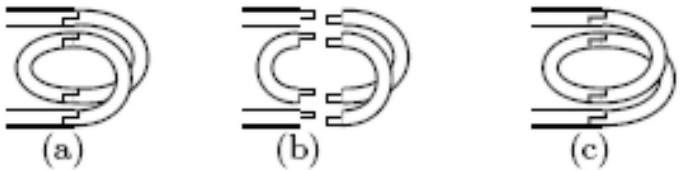



**Fig. 3.** Illustration of the dlad molecular operation.

# 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

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 $\mathbf{spr_p}(u)$.

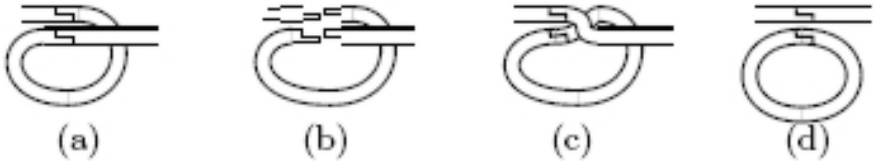



**Fig. 1.** Illustration of the ld molecular operation.

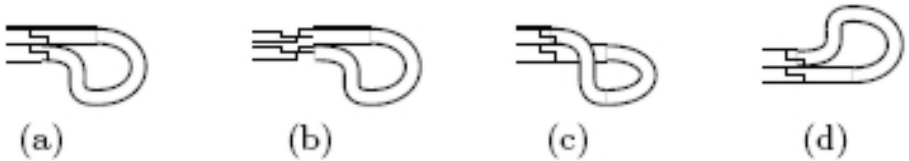



**Fig. 2.** Illustration of the hi molecular operation.

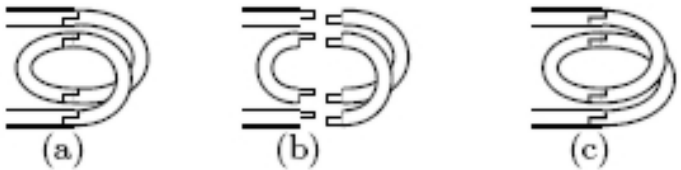



**Fig. 3.** Illustration of the dlad molecular operation.

# 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

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

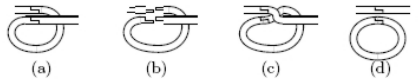



**Fig. 1.** Illustration of the ld molecular operation.

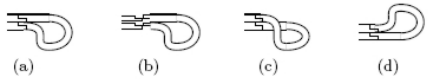



**Fig. 2.** Illustration of the hi molecular operation.

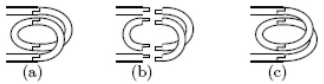



**Fig. 3.** Illustration of the dlad molecular operation.

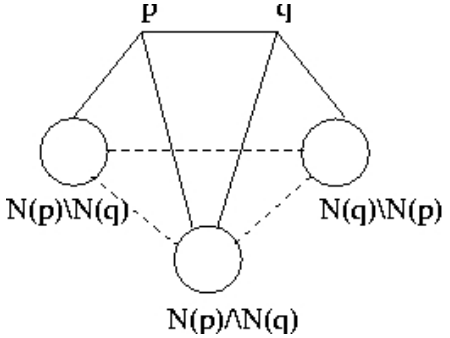
DLAD for MDS descriptors:

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

# 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.

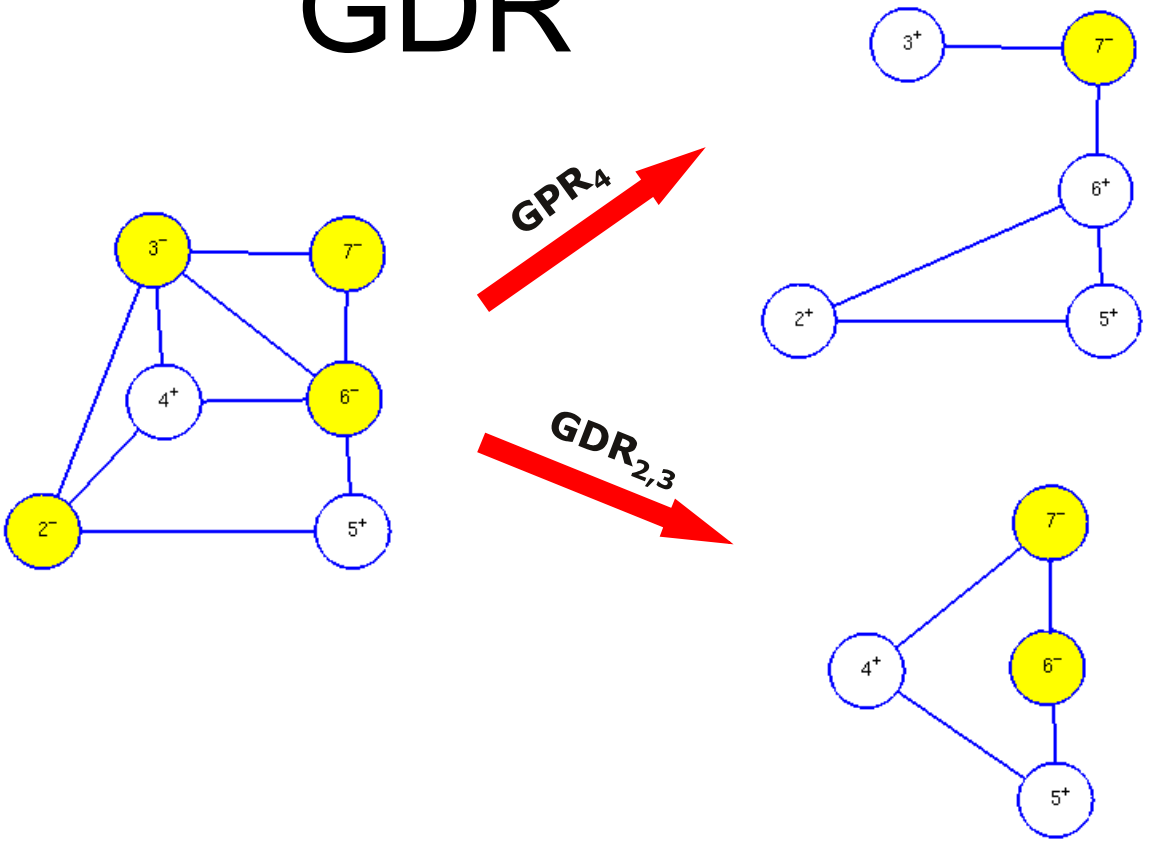



In other words, the status of a pair (x,y), for $x,y \in G-\{p,q\}$ will change if and only if

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

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

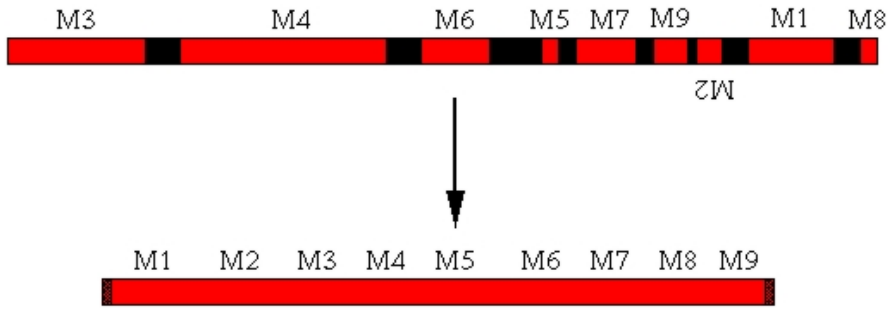


# Reduction strategies

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

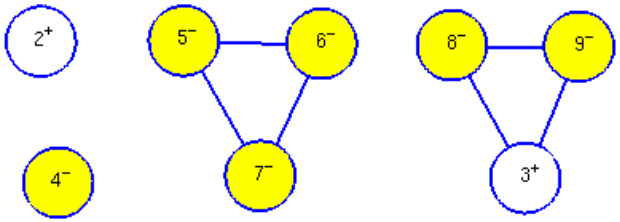
# Example: actinI gene in S.nova

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



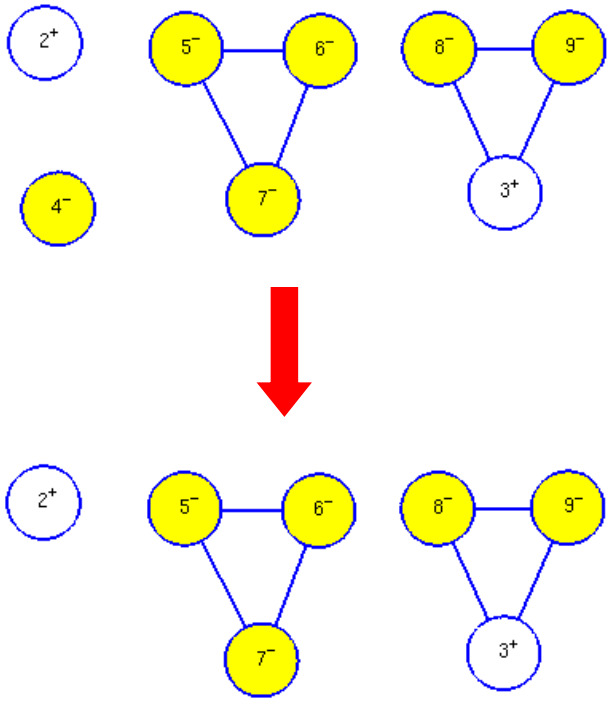

MDS descriptor: δ=(3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
Legal string: u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

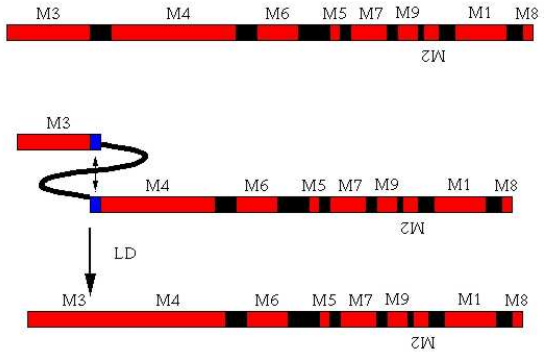
Overlap graph:



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

Step 1: **$GNR_4$**



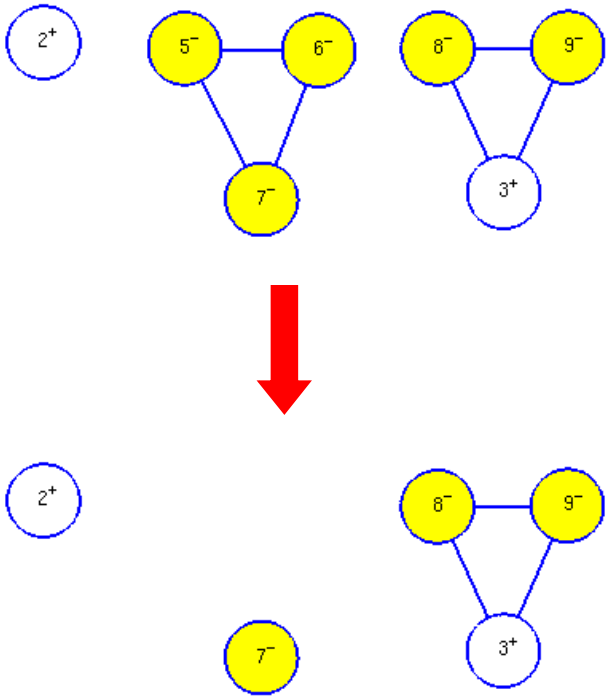

$\delta$=(3,4)(4,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)
$ld_4(\delta)$=(3,5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)

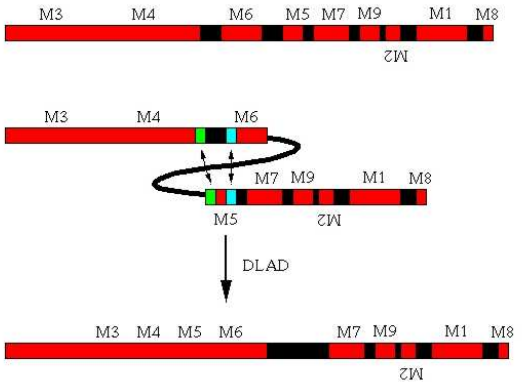
u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

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

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

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




$\delta_2$=(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)**

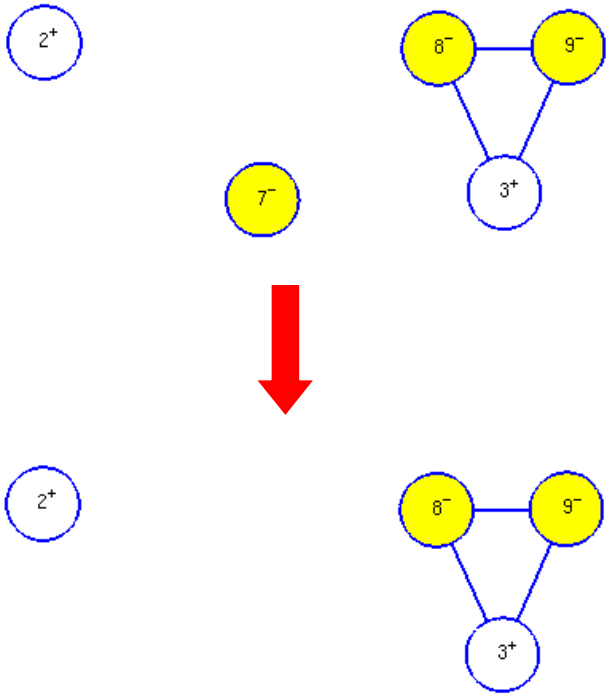
u=3 4 4 5 6 7 5 6 7 8 9 -3 -2 2 8 9

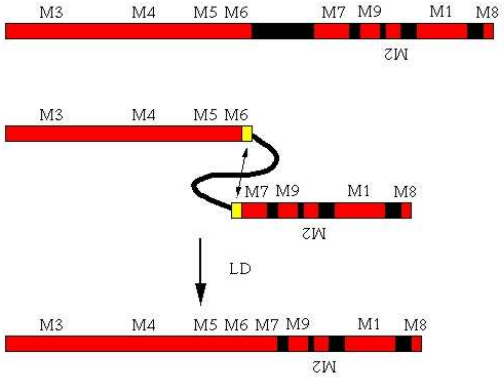
$snr_4$(u)=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9

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

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

Step 3: **$GNR_7$**




$\delta_3$=(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

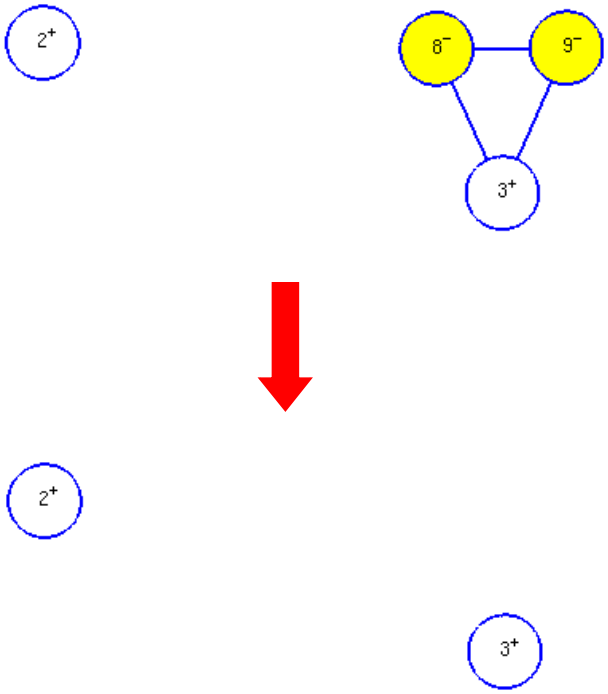
$snr_4(u)$=3 5 6 7 5 6 7 8 9 -3 -2 2 8 9

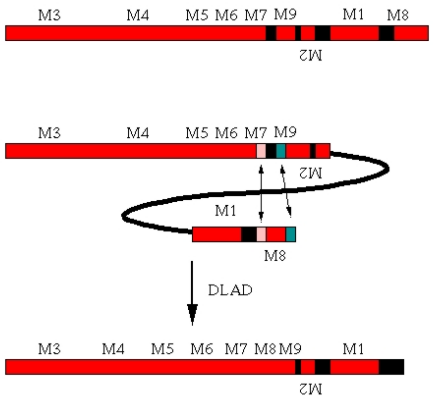
$sdr_{5,6}(snr_4(u))$= 3 7 7 8 9 -3 -2 2 8 9

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

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

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




$\delta_4$=(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

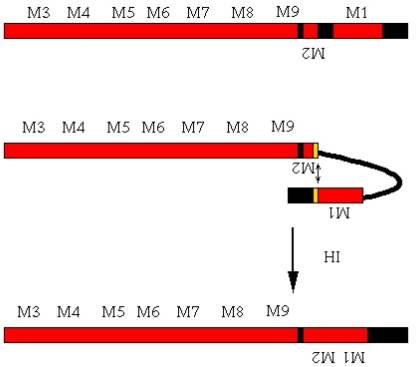
$snr_7(sdr_{5,6}(snr_4(u)))$= 3 8 9 -3 -2 2 8 9

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

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

Step 5: **GPR$_2$**




$\delta_5$=(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

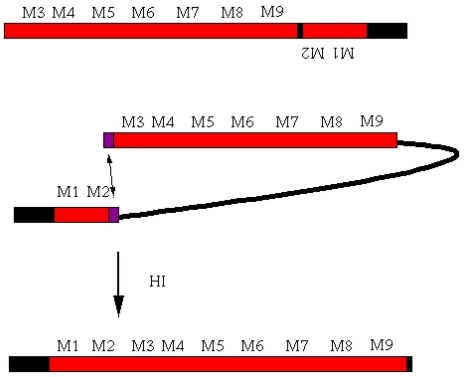
$sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u))))$= 3 -3 -2 2

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

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

Step 6: **$GPR_3$**


∅




$\delta_6$=(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

# Computing with gene assembly

- We solved HPP
- By gene assembly
- In intramolecular model

- Inspired by Adleman's experiment

- Given an HPP instance: *G, b, e*
- We design hypothetical *gene pattern*
- Assembling the gene pattern equivalent to solving HPP

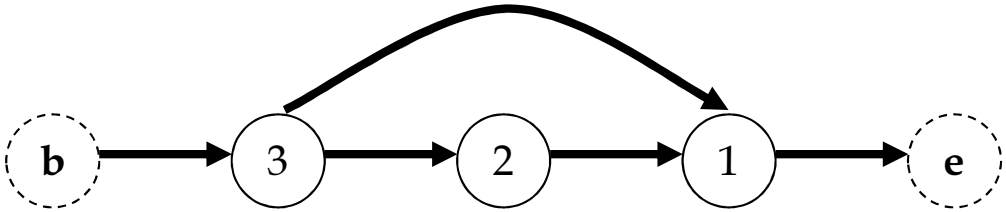
# Computing with gene assembly

- Applying non-deterministically LD, HI, DLAD
- To the gene pattern
- We obtain molecules
- Each of them corresponds to a path in G

- Molecule corresponding to HP can be assembled
- IF AND ONLY IF
- The HP exists in *G*

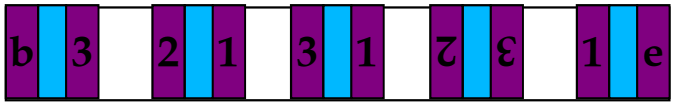
- According to Laura Landweber et al. from Princeton University, gene assembly in ciliates is driven by maternal MAC DNA/RNA.
- It should be possible to “reprogram” gene assembly.
- Natasa Jonoska et al. from University of South Florida developed a formal model predicting template-based DNA recombination in ciliates

# Example: an Instance of HPP Encoded for LD, HI and DLAD

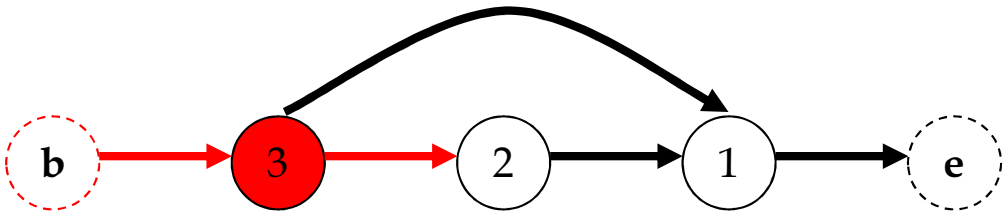



-Vertices → pointers

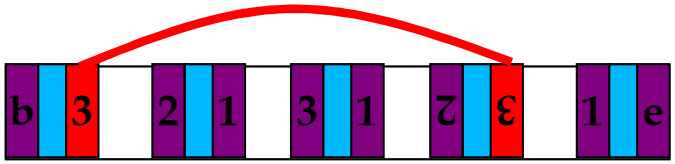
-Edges → MDS's



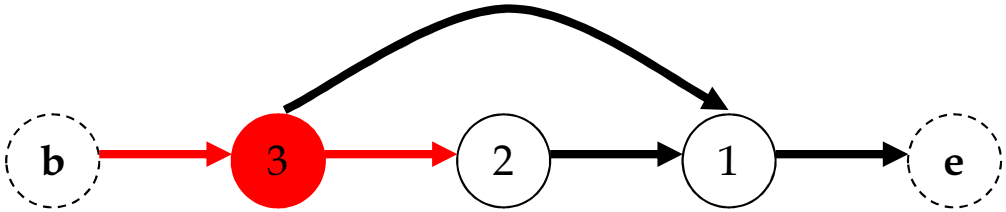
# Example: an Instance of HPP Encoded for LD, HI and DLAD



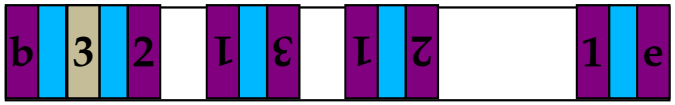

-$HI_3$:(b,3), (3,2)



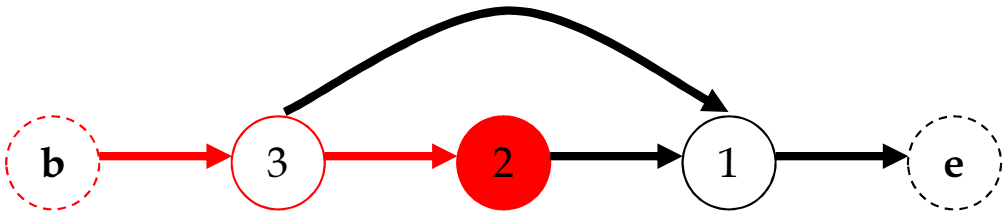
# Example: an Instance of HPP Encoded for LD, HI and DLAD



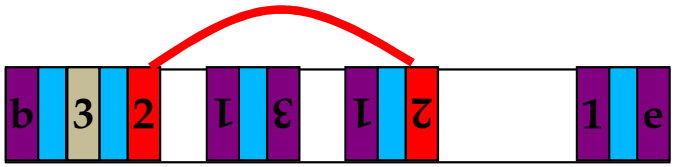

-$HI_3$:(b,3), (3,2) → b32



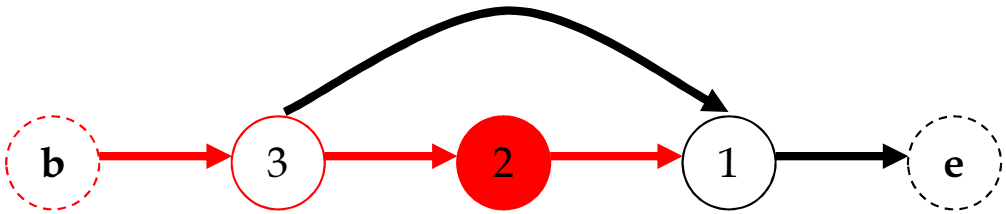
# Example: an Instance of HPP Encoded for LD, HI and DLAD



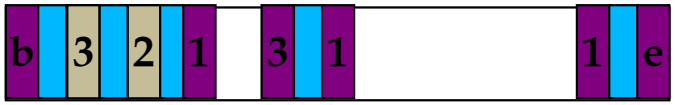

-$HI_2$:(b,3,2), (-1,-2)→



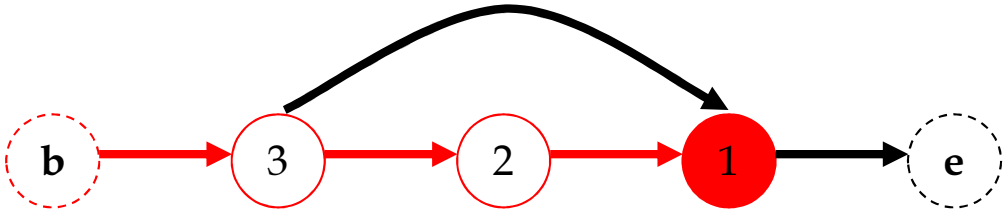
# Example: an Instance of HPP Encoded for LD, HI and DLAD



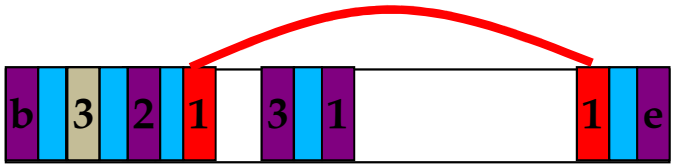

-$HI_2$:(b,3,2), (-1,-2)→(b,32,1)



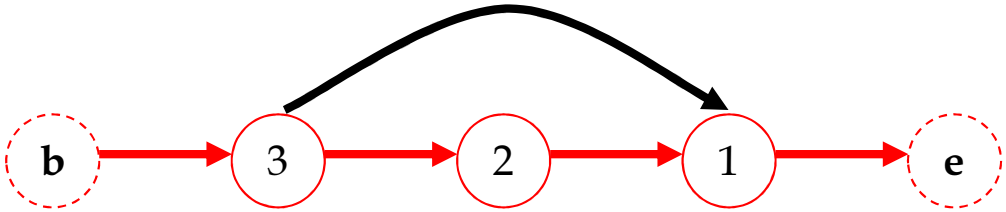
# Example: an Instance of HPP Encoded for LD, HI and DLAD



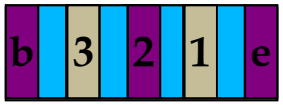

-$LD_1$:(b,32,1), (1,e)→



# Example: an Instance of HPP Encoded for LD, HI and DLAD




-$LD_1$:(b,32,1), (1,e)→(b,321,e)



# Computing with gene assembly – Turing universality

- Proved both for the intermolecular and the intramolecular models
- By gene assembly simulated a Turing Machine
- Both the “tape” and the “program” encoded as a sequence of pointers on a MIC molecule
- TM rewriting rules represented by combination of dlad’s and ld’s
- The word is accepted by the Turing Machine iff its encoding is assembled to the molecule containing the special substring

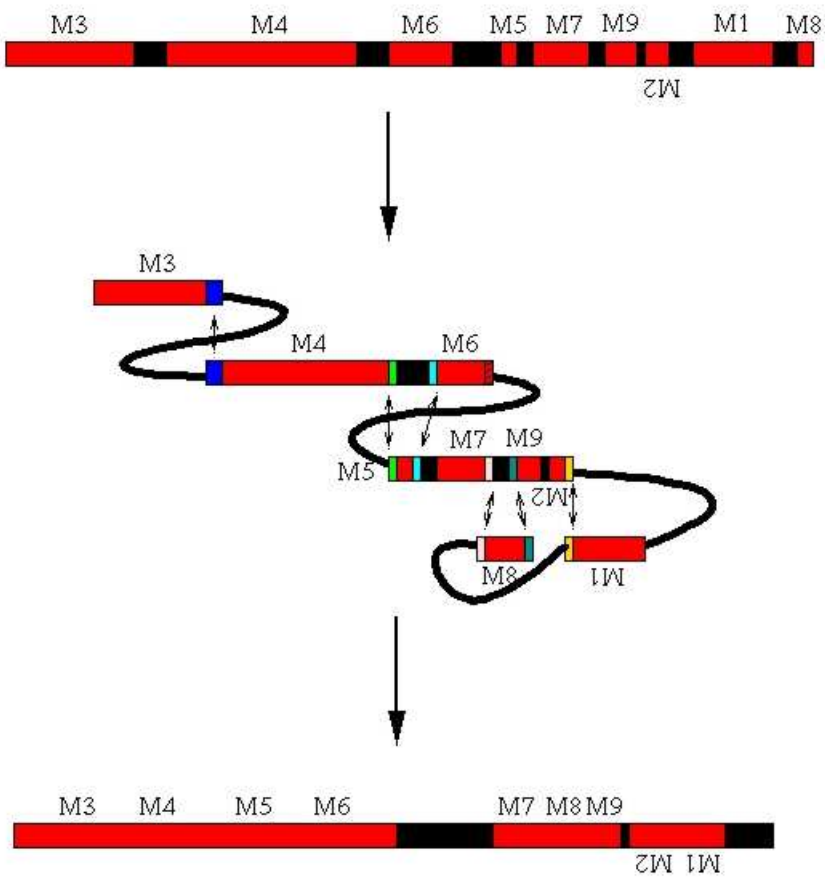
# Parallelism of gene assembly

- Bio-molecular processes tend to be **highly parallel, in contrast** with the example on the previous slides

- A more realistic assembly of actin I could run as follows:

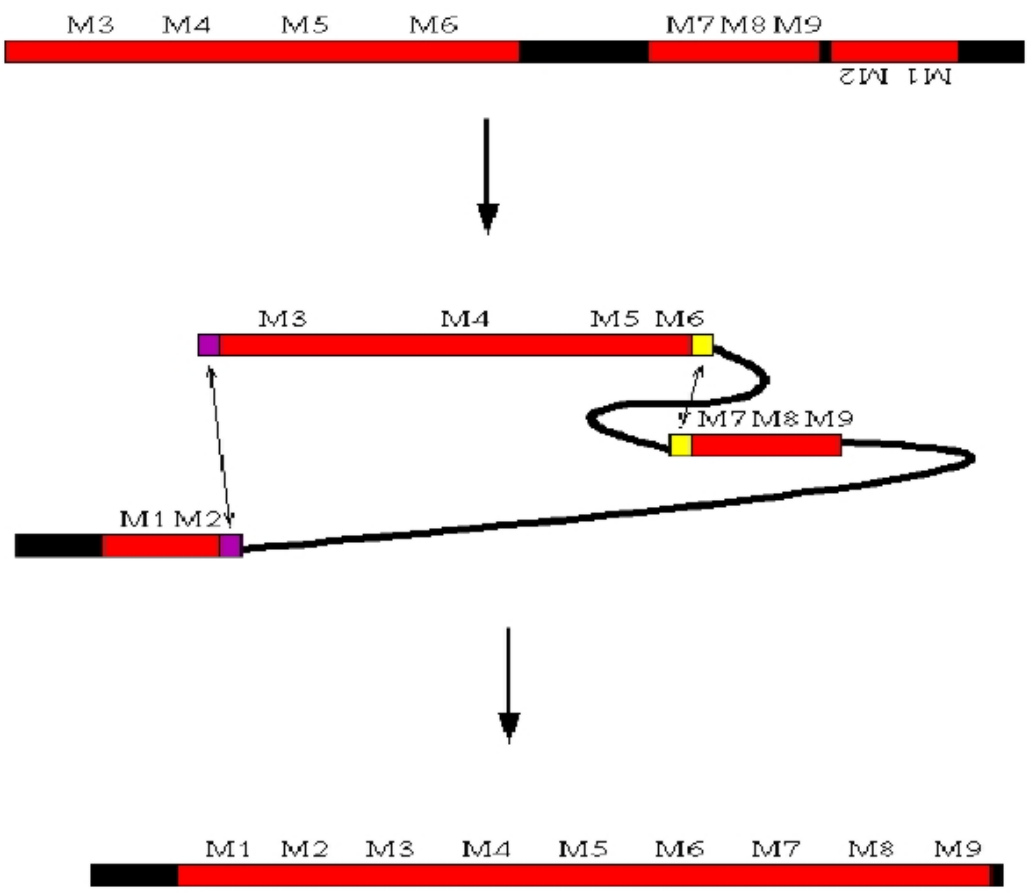
# Parallelism of gene assembly

**Step 1:**



# Parallelism of gene assembly

**Step 2:**



# Parallelism in gene assembly

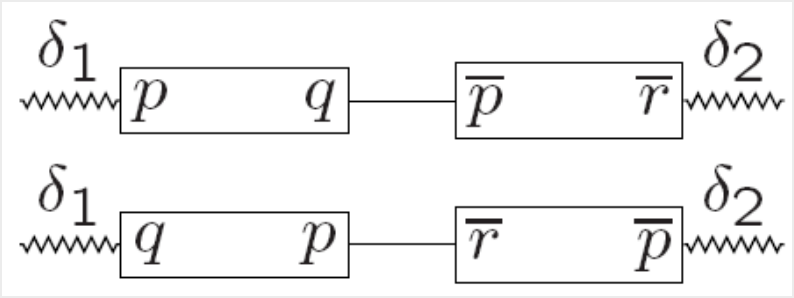
- **Goal:** Reason about the parallelism
- • **Question 1:** How quickly can a given gene be assembled in parallel (measure of complexity)?
- • **Question 2:** For a given gene, find (efficiently!) a minimal parallel reduction.
- • **Question 3:** For a given gene and a given set of operations, decide if that set can be applied in parallel to the gene.
- **Answer:** there are several solutions in the $NP^{NP}$ time complexity
- **Main formalism:** signed graphs

# Measures of complexity: simple operations

- 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

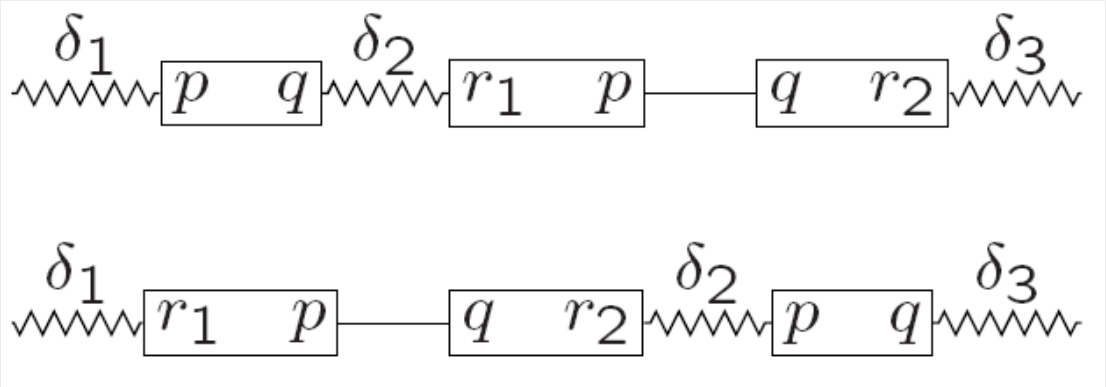
# Simple operations

- ld is always simple: there is only one IES between p
  - Note: A boundary application of ld is always the last step in a circular assembly
- **Simple $hi_p$: there is only one IES between p and p:**
  - 

- Effect: p is removed from the pattern and at most one pointer is inverted, when simple $hi_p$ is applied

# Simple operations

- **Simple dladp,q: there is exactly one IES in the two sequences between** p and q:




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

# Non-universality result for simple operations

- The set of our simple operations is NOT universal - there are MDS descriptors that cannot be assembled using simple operations only
- Example: (2, b)(4, e)(3, 4)(2, 3) – no simple operation is applicable
- Question: are there any ciliate micronuclear patterns that cannot be assembled using 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


# Summary

- We can reason formally about complex processes in living cells
  - We can discover/explain some properties in living organisms by applying mathematical/computational methods
  - We can regard processes in living organisms as computations and use those processes in order to compute
  - We can even ”reprogram” cellular behavior