Special course in Computer Science: Molecular Computing

Lecture 14: Complexity and universality of gene assembly. Invariants.

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

Fall 2013

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

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

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
  - $(\bar{i},\overline{i-1})\ldots(\overline{2},\bar{b})(\overline{e},\bar{k})\ldots(\overline{i+2},\overline{i+1})(\overline{i+1},\bar{i})$, $1 \le i \le k$
- 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*

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

TUCS

Simple hi

The hi_p operation:

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

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

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:

TUCS

Simple dlad

$sdlad_{p,q}(\delta 1(p,q)\delta 3(r3,p)(q,r4)\delta 5)=\delta 1\delta 3\ (r3,r4)\delta 5$

$sdlad_{p,q}(\delta 1(r1,p)(q,r2)\delta 3(p,\ q)\delta 5)=\delta 1(r1,r2)\delta 3\delta 5$

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

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 ?

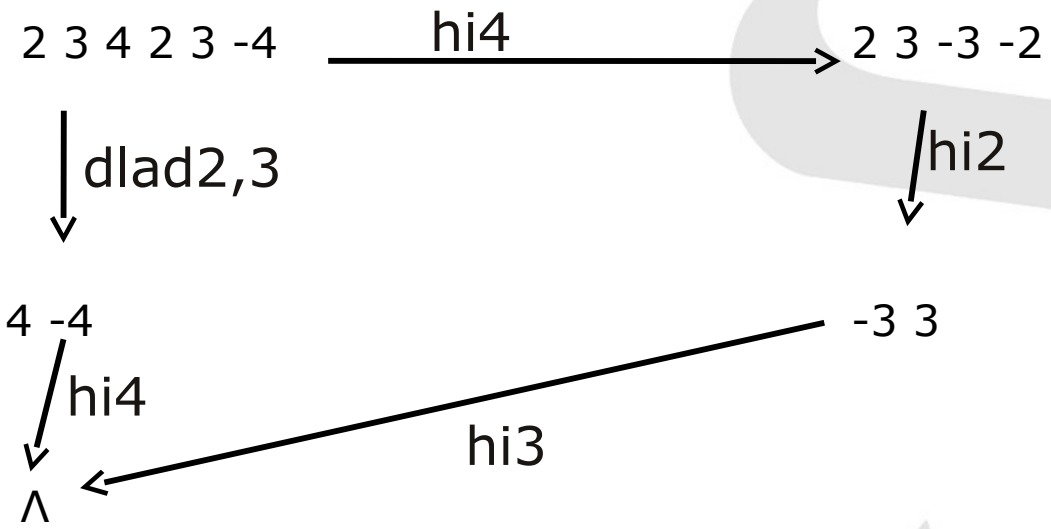
TUCS

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

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:



TUCS

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

Assembly is nondeterministic

- Assembling MDS descriptors is nondeterministic!

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

$hi_3(u)$=(3, 5)(6,7)(5,6)(7,8)(9,e)(-3,-2)(b,2)(8,9)

$ld_{-4}(hi_3(u))$=(-e,-9)(-8,-7)(-6,-5)(-7,-6)(-5,-2)(b,2)(8, 9)

$hi_8(ld_{-4}(hi_3(u)))$=(-e,-9)(-2,-b)(2,5)(6,7)(5,6)(7,9)

$hi_{-2}(hi_8(ld_{-4}(hi_3(u))))$= (-e,-9)(b,5)(6,7)(5,6)(7,9)

$dlad_{5,7}(hi_{-2}(hi_8(ld_{-4}(hi_3(u)))))$=(-e,-9)(b, 6)(6,9)

$ld_6(dlad_{5,7}(hi_{-2}(hi_8(ld_{-4}(hi_3(u))))))$=(-e,-9)(b,9)

$hi_9(ld_6(dlad_{5,7}(hi_{-2}(hi_8(ld_{-4}(hi_3(u)))))))$= (-e,-b)

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

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

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

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

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

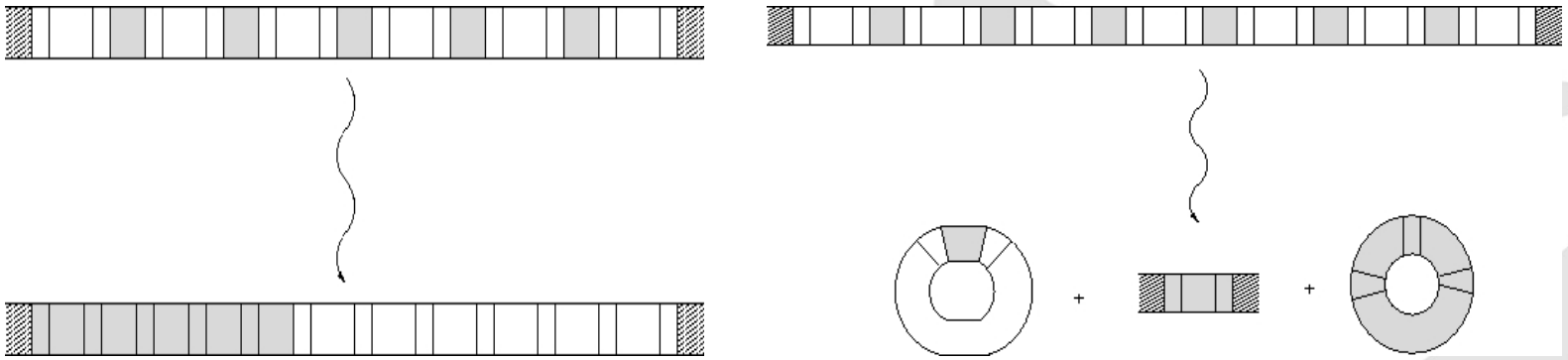
$spr_{-2}(sdr_{8,9}(snr_7(sdr_{5,6}(snr_4(u)))))$=(3,e)(-3,-b)

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

Reductions can have (in principle) different outcomes

- Assembly is nondeterministic!
- A descriptor can be assembled linearly or circularly

- $ld_p(\ \delta_1\ (q,p)\ (p,r)\ \delta_2\ ) = \delta_1\ (q,r)\ \delta_2$ **(linear molecule)**
- $ld_p\ (\ (p,r)\ \delta\ (s,p)\ ) = (s,r)\ \delta$ ***(circular molecule)***



Reductions can have (in principle) different outcomes

- Assembly is nondeterministic!
- A descriptor can be assembled linearly or circularly
  - $ld_p(\ \delta_1\ (q,p)\ (p,r)\ \delta_2\ ) = \delta_1\ (q,r)\ \delta_2$ **(linear molecule)**
  - $ld_p\ (\ (p,r)\ \delta\ (s,p)\ ) = (s,r)\ \delta$ ***(circular molecule)***

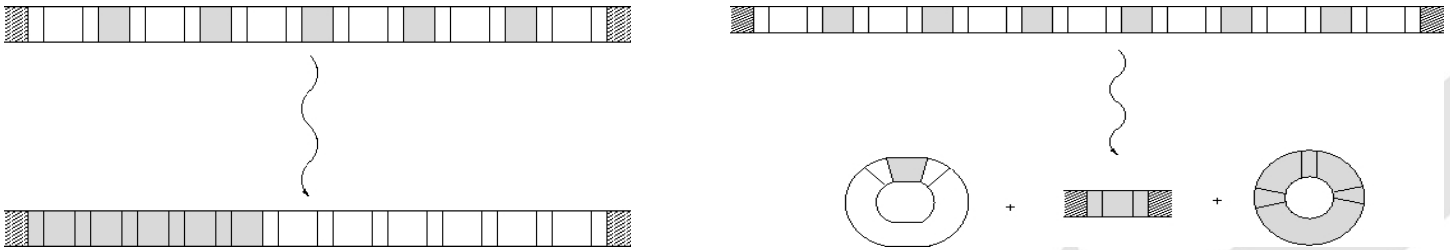
x=(3,4)(b,2)(-e,-4)(2,3) is assembled on a circular molecule

$hi_4(x)=(3,e)(-2,-b)(2,3)$; $hi_2(hi_4(x))=(3,e)(b,3)$; $\mathbf{ld_3(hi_2(hi_4(x)))=[b,e]}$

y=(3,4)(b,2)(-e,-4)(-3,-2) is assembled on a linear molecule

$hi_4(y)=(3,e)(-2\text{-}b)(-3,-2)$, $hi_3(hi_4(y))=(b,2)(-e,-2)$, $hi_2(hi_3(hi_4(y)))=(b,e)$

$hi_2(y)=(3,4)(b,3)(4,3)$, $\mathbf{dlad_{3,4}(hi_2(y))=(b,e)}$



Outcome of the gene assembly process

A gene can have several assembling strategies – nondeterministic process

The gene assembly process always produces:

- One linear molecule
- Possibly several circular molecules (due to *ld*)

The gene is placed on one of these molecules, possibly attached to a number of IESs

Questions:

- Is it possible than one strategy assembles the gene on a circular molecule, while another strategy assembles it on a linear one ?
- Is the set of excised molecules dependent on the strategy?

Invariants of the gene assembly process

Results

- If the gene is assembled on a circular (linear, resp.) molecule by one *particular strategy*, then *all possible assembly strategies* will assemble the gene on a circular (linear, resp.) molecule
- The *context* of the gene (the sequence of IESs attached to the gene): *always the same, regardless of the reduction strategy*
- *The set of excised molecules always the same, regardless of the reduction strategy*
- **Note**: the *ld* operations (those that excise molecules) need not be applied to the same pointers in every assembling strategy

Example

To show the context of the gene , we keep track of IESs here

δ=(5,6) I_1 (2,3) I_2 (b,2) I_3 (4,5) I_4 (7,e) I_5 (3,4) I_6 (6,7)

- One possible assembly:
  - $dlad_{2,3}(\delta)$ = (5,6) I_1 I_3 (4,5) I_4 (7,e) I_5 I_2 (b,4) I_6 (6,7)
  - $dlad_{5,6}(dlad_{2,3}(\delta))$ = I_4 (7,e) I_5 I_2 (b,4) I_6 I_1 I_3 (4,7)
  - $ld_4(dlad_{5,6}(dlad_{2,3}(\delta)))$ = I_4 (7,e) I_5 I_2 (b,7) + [I_6 I_1 I_3]
  - $ld_7(ld_4(dlad_{5,6}(dlad_{2,3}(\delta))))$ = **I_4 + [(b,e) I_5 I_2] + [I_6 I_1 I_3]**

- Another one:
  - $dlad_{4,7}(\delta)$ = (5,6) I_1 (2,3) I_2 (b,2) I_3 I_6 (6,e) I_5 (3, 5) I_4
  - $dlad_{2,3}(dlad_{4,7}(\delta))$ = (5,6) I_1 I_3 I_6 (6,e) I_5 I_2 (b,5) I_4
  - $ld_6(dlad_{2,3}(dlad_{4,7}(\delta)))$ = (5, e) I_5 I_2 (b,5) I_4 + [I_1 I_3 I_6]
  - $ld_5(ld_6(dlad_{2,3}(dlad_{4,7}(\delta))))$ = **I_4 + [(b,e) I_5 I_2] + [I_1 I_3 I_6]**

Invariants of gene assembly: other results

Results

- A gene is assembled on a *linear molecule* with *no context* (no IES attached to it) if and only if its micronuclear form contains either the substring

  (k,e)(b,2) or *(-2,-b)(-e,-k)*

- A gene is assembled on a *circular molecule* with *no context* (no IES attached to it) if and only if its micronuclear form starts/ends with the MDSs

  (b,2)/(k,e) or *(-e,-k)/(-2,-b)*

Acknowledgements

- The course slides used material from:
  - Ion Petre's course “Introduction to Biocomputing, Fall 2004”
  - Ion Petre's course “Computational processes in living cells, Spring 2010”