ACGT Algorithms in Computational Genomics at Tau
Ron Shamir's group

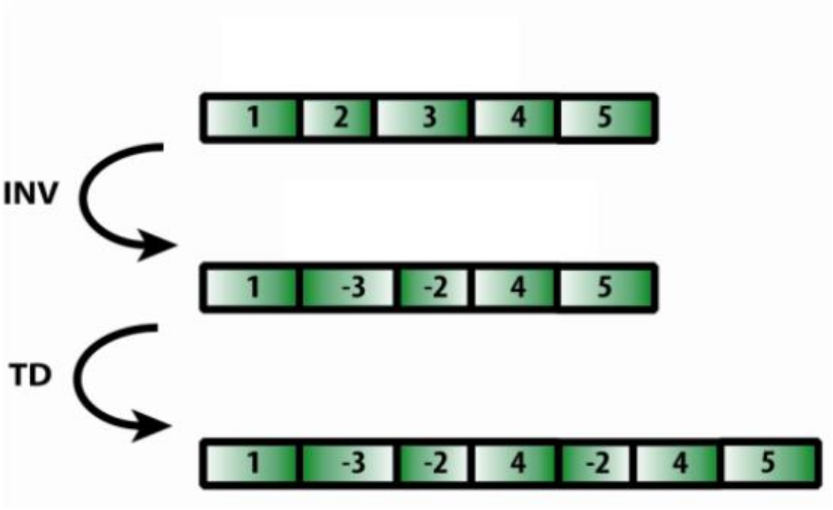
Reconstructing Cancer Karyotypes

Rami Eitan, Tel-Aviv University

13/8/14

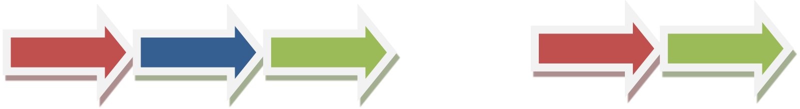
Driving mechanism of cancer

- Chromosomal rearrangements that gradually accumulate over time and create a complex cancer karyotype.



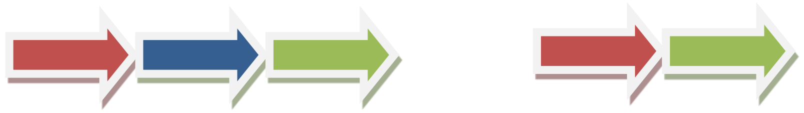
We consider 3 types of intra-chromosomal rearrangements:

- Deletion

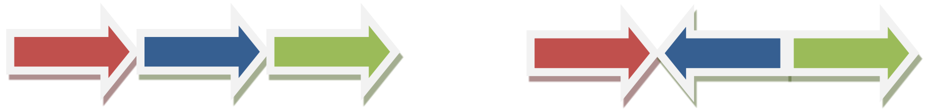


We consider 3 types of intra-chromosomal rearrangements:

- Deletion

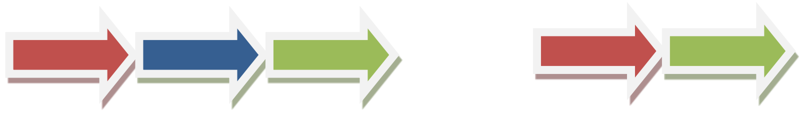


- Inversion

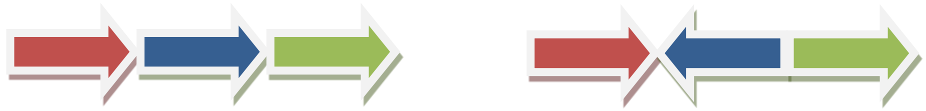


We consider 3 types of intra-chromosomal rearrangements:

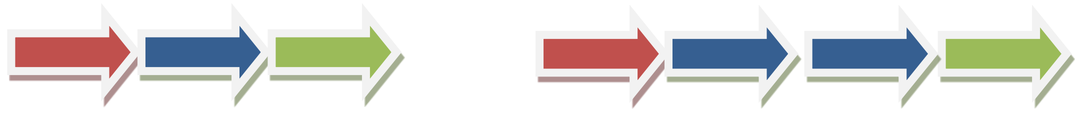
- Deletion



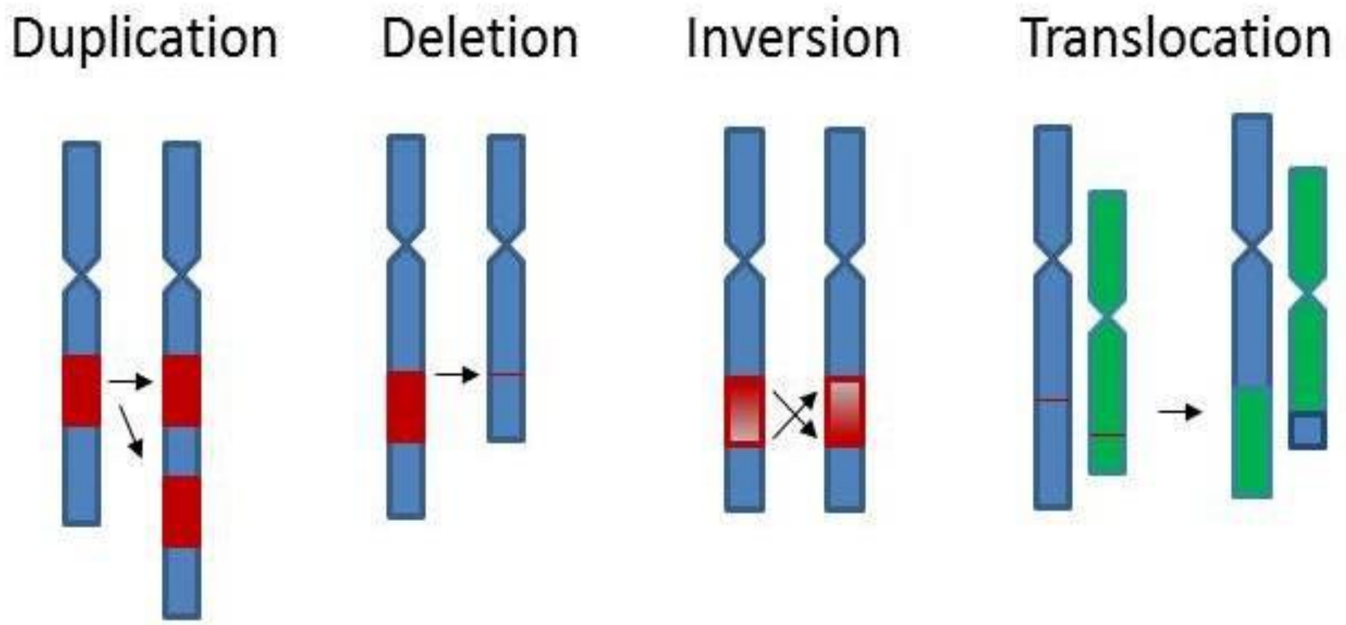
- Inversion



- Tandem Duplication



- In addition to duplication, inversion and deletion we also consider the inter-choromosomal variation of translocation.



Methods for inferring structural variations (SV) in a cancer genome:

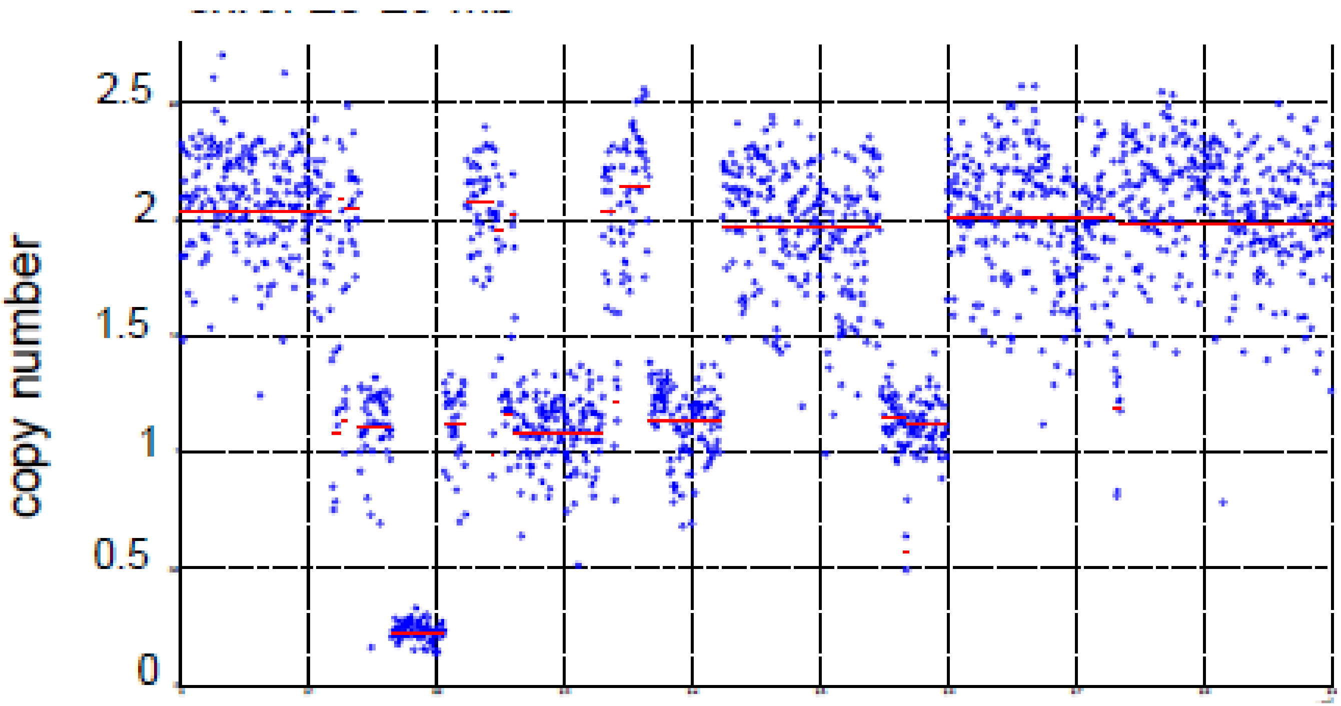
Methods for inferring structural variations (SV) in a cancer genome:

- CGH array – Detect Copy Number Variations (CNV), resolution as low as 100 Kilo bases.

Methods for inferring structural variations (SV) in a cancer genome:

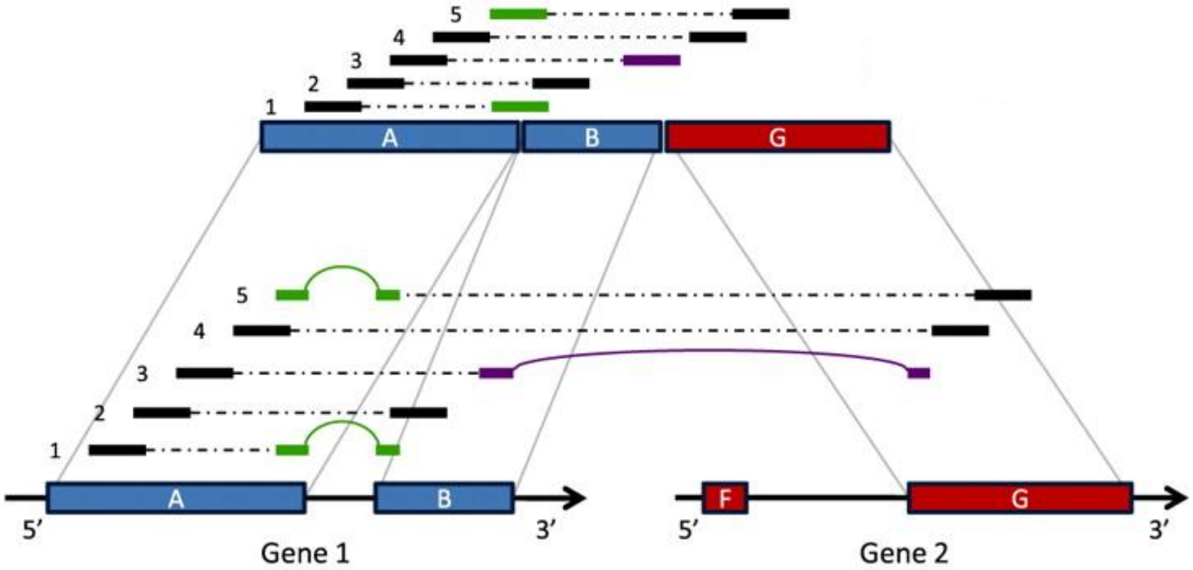
- CGH array – Detect Copy Number Variations (CNV), resolution as low as 100 Kilo bases.
- Paired Ends Reads – Detect novel adjacencies in the genome compared to the reference.

CGH Array



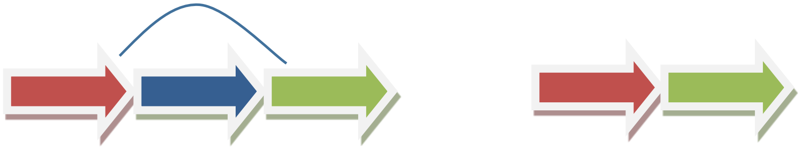
Paired-end reads

- Sample DNA sequence *S* is cut into small fragments (200-500 bp)
- Each end of the fragment (36 bp) is then aligned against a reference genome *R.*
- Concordant reads – both ends aligned to the same distance
- Discordant reads – ends are aligned to a different distance.



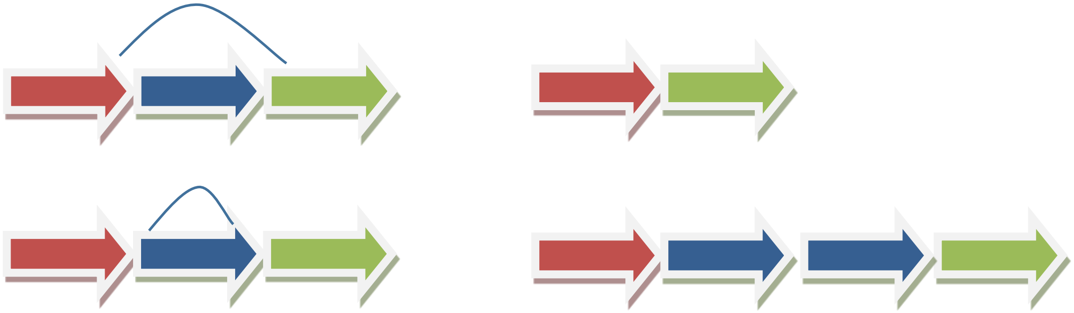
Orientations of breakpoints determine the rearrangement

Orientations of breakpoints determine the rearrangement



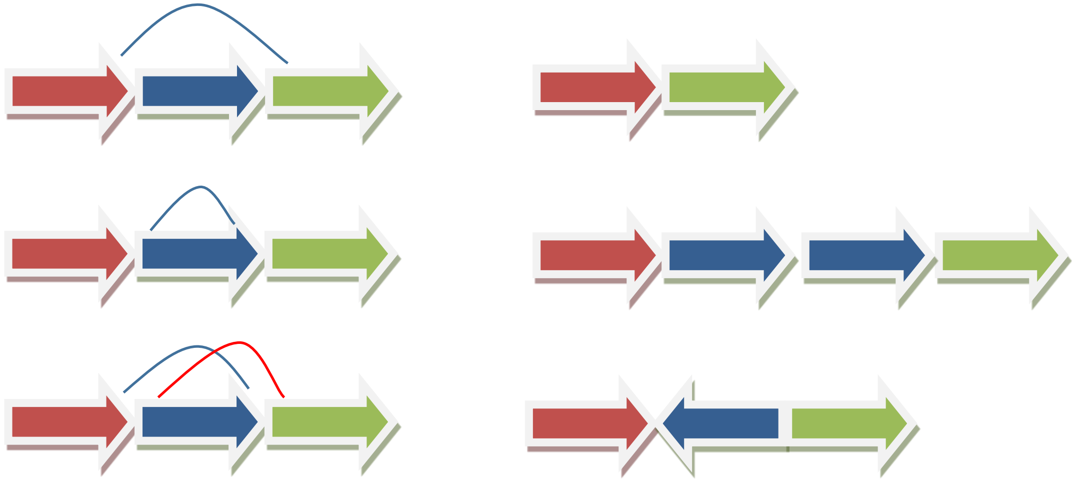
Orientation	Rearrangement
head-to-tail	Deletion

Orientations of breakpoints determine the rearrangement

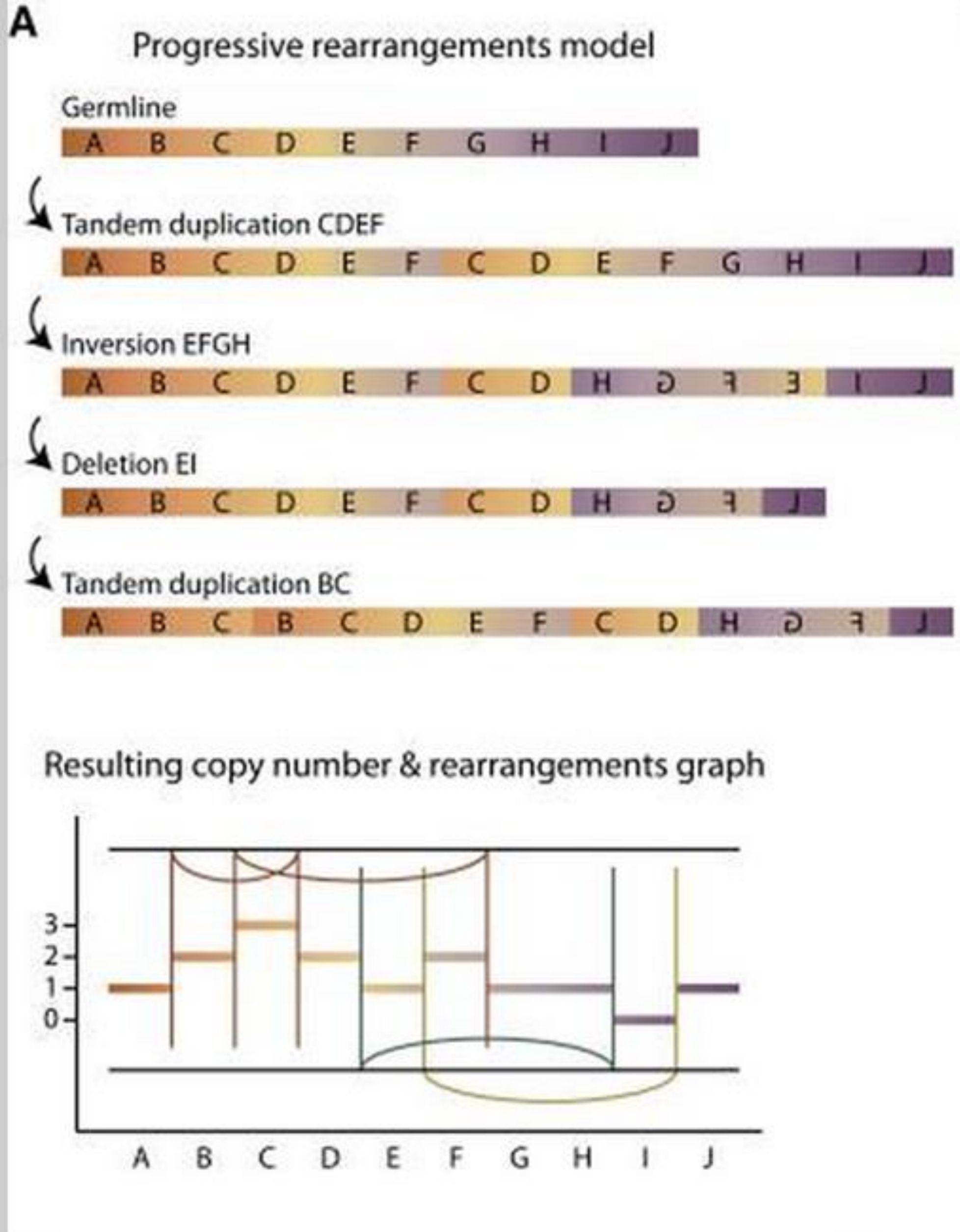


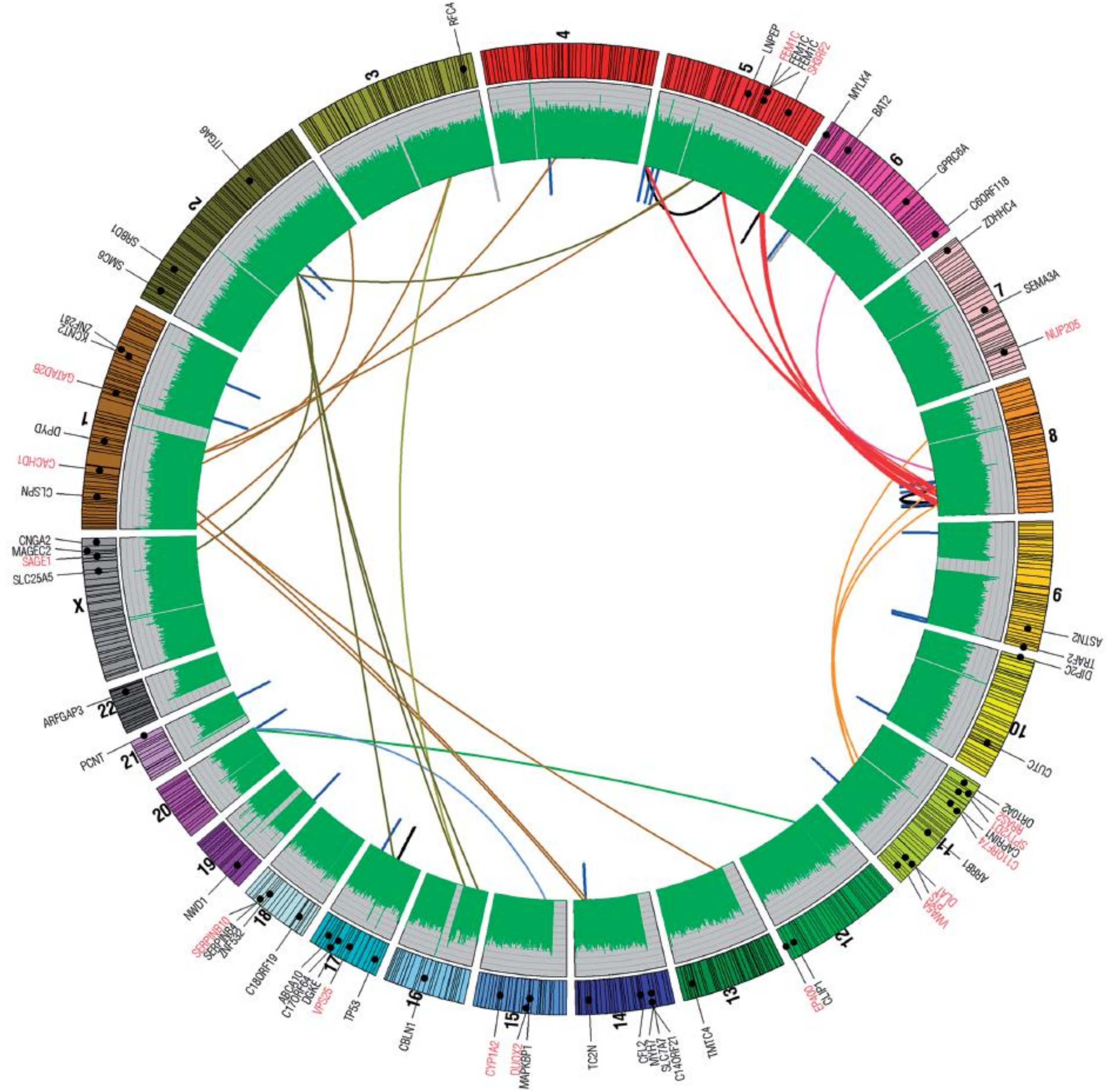
Orientation	Rearrangement
head-to-tail	Deletion
tail-to-head	Duplication

Orientations of breakpoints determine the rearrangement



Orientation	Rearrangement
head-to-tail	Deletion
tail-to-head	Duplication
tail-to-tail	Inversion
head-to-head	Inversion


A
Progressive rearrangements model
Germline
A B C D E F G H I J
Tandem duplication CDEF
A B C D E F C D E F G H I J
Inversion EFGH
A B C D E F C D H G F E I J
Deletion EI
A B C D E F C D H G F J
Tandem duplication BC
A B C B C D E F C D H G F J
Resulting copy number & rearrangements graph
3
2
1
0
A B C D E F G H I J



Breast tumor HCC1954

Galante 2011

Genome reconstruction

Genome reconstruction

From the reads we derive:

Genome reconstruction

From the reads we derive:

- A sequence of intervals (segments) $\boldsymbol{I} = (I_1, I_2, \dots, I_n)$ Each Interval $I_j = [s_j, t_j]$.

Genome reconstruction

From the reads we derive:

- A sequence of intervals (segments) $\boldsymbol{I} = (I_1, I_2, \dots, I_n)$ Each Interval $I_j = [s_j, t_j]$.
- From discordant reads: a set of adjacencies (Breakpoints) $A \subseteq \{(I_j, I_k) | j, k \in \{\pm 1, \pm 2, \dots, \pm n\}\}$

Genome reconstruction

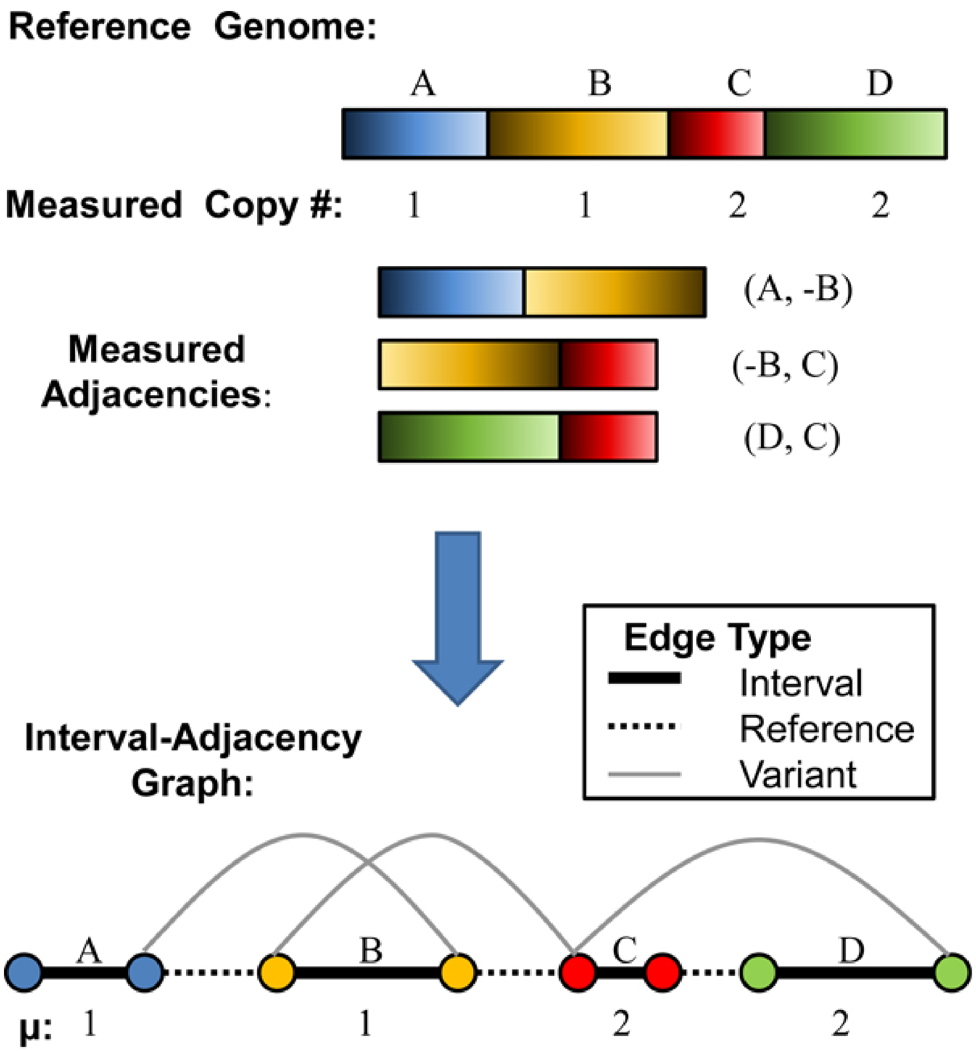
From the reads we derive:

- A sequence of intervals (segments) $\boldsymbol{I} = (I_1, I_2, \dots, I_n)$ Each Interval $I_j = [s_j, t_j]$.
- From discordant reads: a set of adjacencies (Breakpoints) $A \subseteq \{(I_j, I_k) | j, k \in \{\pm 1, \pm 2, \dots, \pm n\}\}$
- From CGH array: CNV data - $c: I \rightarrow R$

Copy number and adjacency genome reconstruction problem

Given an interval vector I*, a set* A *of cancer adjacencies, and a copy number vector derived from a cancer sample* S*, find the cancer genomes that are most consistent with the data.*

Interval-adjacency graph



Reference Genome:

A B C D

Measured Copy #: 1 1 2 2

Measured Adjacencies:

(A, -B)

(-B, C)

(D, C)

Edge Type

Interval

Reference

Variant

Interval-Adjacency Graph:

A B C D

μ: 1 1 2 2

Interval-adjacency graph

Interval-adjacency graph

- Undirected graph G(V,E)

Interval-adjacency graph

- Undirected graph G(V,E)

$$V = \{s_1, t_1, s_2, t_2, \dots, s_n, t_n\}$$

Interval-adjacency graph

- Undirected graph G(V,E)

$$V = \{s_1, t_1, s_2, t_2, \dots, s_n, t_n\}$$

$$E = E_I \cup E_R \cup E_V$$

Interval-adjacency graph

- Undirected graph G(V,E)

$$V = \{s_1, t_1, s_2, t_2, \ldots, s_n, t_n\}$$

$$E = E_I \cup E_R \cup E_V$$

- *Interval edges* $E_I = \{e_I(j) = (s_j, t_j)\}$

Interval-adjacency graph

- Undirected graph G(V,E)

$$V = \{s_1, t_1, s_2, t_2, \dots, s_n, t_n\}$$

$$E = E_I \cup E_R \cup E_V$$

- *Interval edges* $E_I = \{e_I(j) = (s_j, t_j)\}$
- *Reference Edges* $E_R = \{(t_j, s_{j+1})\}$

Interval-adjacency graph

- Undirected graph G(V,E)

$$V = \{s_1, t_1, s_2, t_2, \dots, s_n, t_n\}$$

$$E = E_I \cup E_R \cup E_V$$

- *Interval edges*
- *Reference Edges*
- *Variant edges*

$$E_I = \{e_I(j) = (s_j, t_j)\}$$

$$E_R = \{(t_j, s_{j+1})\}$$

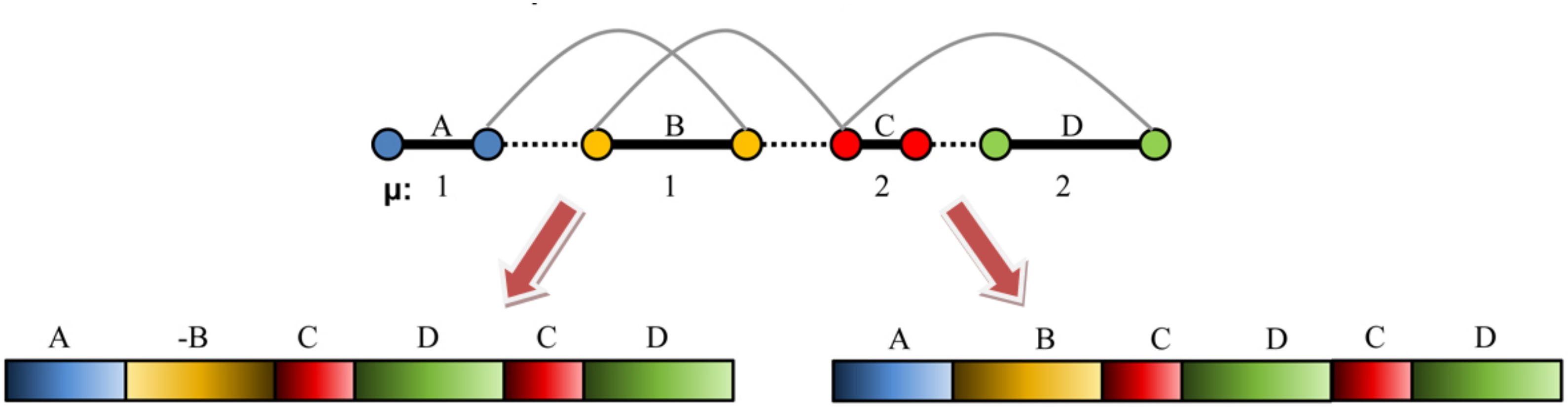
$$E_V = \{(t_i/s_i, s_j/t_j) | [I_{i/-i}, I_{j/-j}] \in A\}$$

A block organization of the cancer genome corresponds to a path along the graph that:

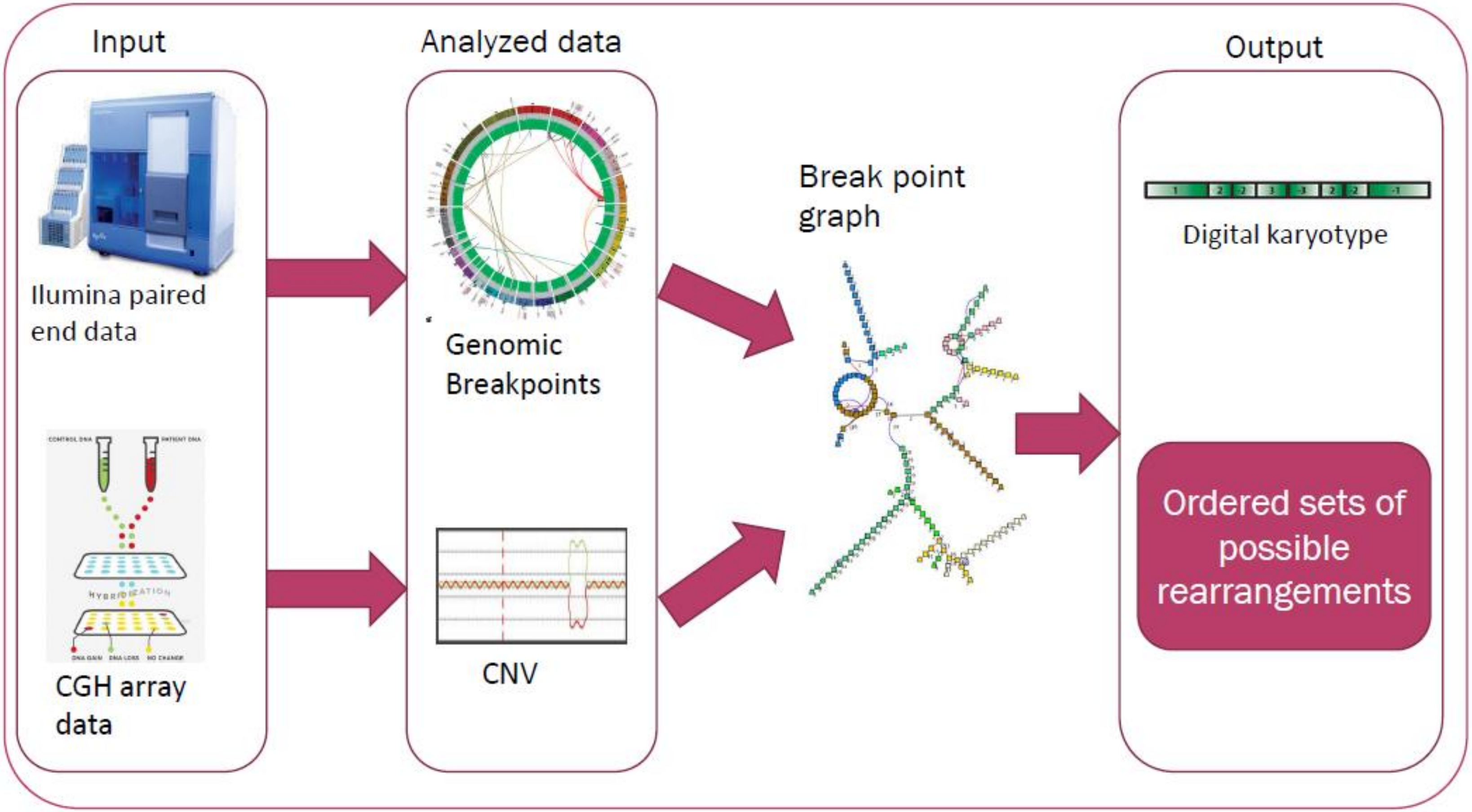
1. *Starts at* s_1 *and ends at* t_n
2. *Alternates between interval edges and non-interval edges.*
3. *The number of times each interval edge is traversed is equal to* c_j

A block organization of the cancer genome corresponds to a path along the graph that:

1. *Starts at s_1 and ends at t_n*
2. *Alternates between interval edges and non-interval edges.*
3. *The number of times each interval edge is traversed is equal to c_j*



Reconstruction pipeline



Input

Ilumina paired end data

CGH array data

Analyzed data

Genomic Breakpoints

CNV

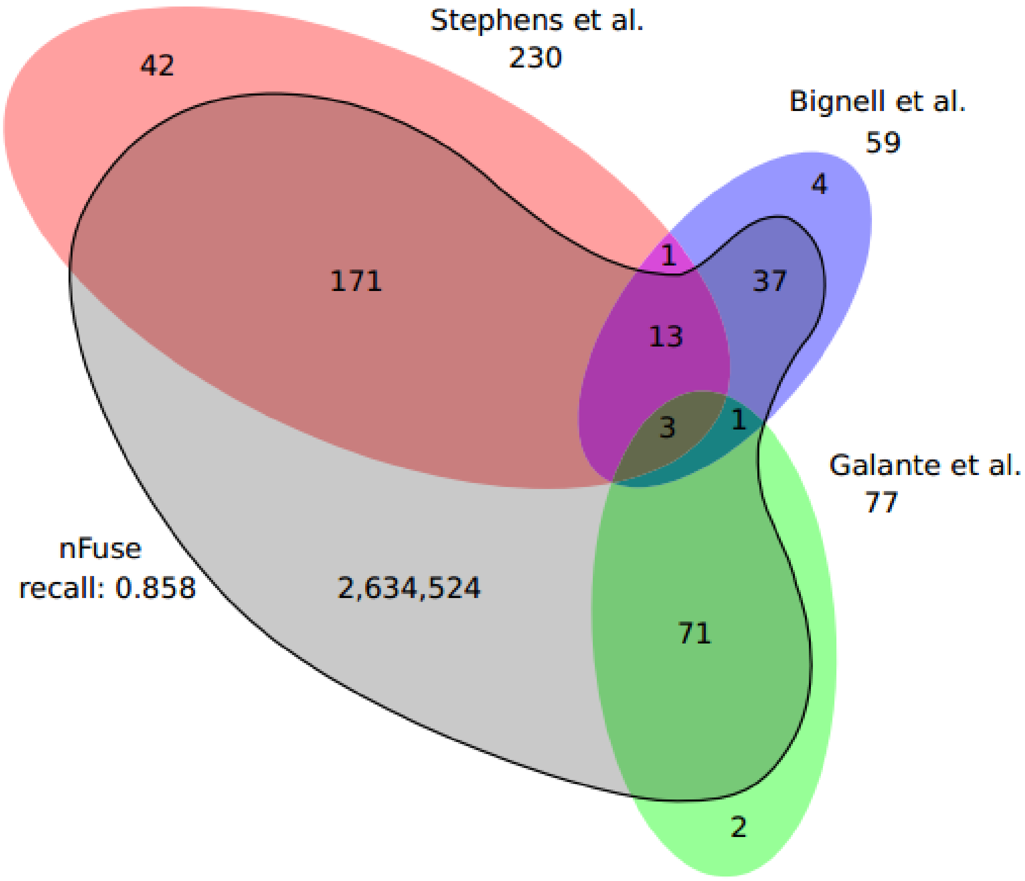
Break point graph

Output

Digital karyotype

Ordered sets of possible rearrangements

Intersection of discovered BPs



Other methods for reconstruction

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)
  - use a statistical model to determine the likelihood of each breakpoint.

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)
    - use a statistical model to determine the likelihood of each breakpoint.
    - Use the likelihood to define a CGR score (*Complex Genomic Rearrangements)* on the graph.

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)
    - use a statistical model to determine the likelihood of each breakpoint.
    - Use the likelihood to define a CGR score (*Complex Genomic Rearrangements)* on the graph.
    - Find a path that maximize the CGR score.

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)
  - use a statistical model to determine the likelihood of each breakpoint.
  - Use the likelihood to define a CGR score (*Complex Genomic Rearrangements)* on the graph.
  - Find a path that maximize the CGR score.
- PREGO (Rephael, Oseper et al. 2012)

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)
  - use a statistical model to determine the likelihood of each breakpoint.
  - Use the likelihood to define a CGR score (*Complex Genomic Rearrangements)* on the graph.
  - Find a path that maximize the CGR score.
- PREGO (Rephael, Oseper et al. 2012)
  - Derive breakpoints and intervals from the data

Other methods for reconstruction

- nFuse (Sainhalp, McPherson et al. 2013)
    - use a statistical model to determine the likelihood of each breakpoint.
    - Use the likelihood to define a CGR score (*Complex Genomic Rearrangements)* on the graph.
    - Find a path that maximize the CGR score.
- PREGO (Rephael, Oseper et al. 2012)
    - Derive breakpoints and intervals from the data
    - Use a maximum likelihood function to estimate the copy number from the concordant reads.

Our attempt

Our attempt

- Use available breakpoint and CNV data

Our attempt

- Use available breakpoint and CNV data
- Construct a weighted breakpoint graph

Our attempt

- Use available breakpoint and CNV data
- Construct a weighted breakpoint graph
- Consider multi ploidity – find the set of paths that are closest to the observed data

Malhorta et al (2013) data

Malhorta et al (2013) data

- Data of 64 tumor and matched normal samples taken from TCGA of different cancer types.

Malhorta et al (2013) data

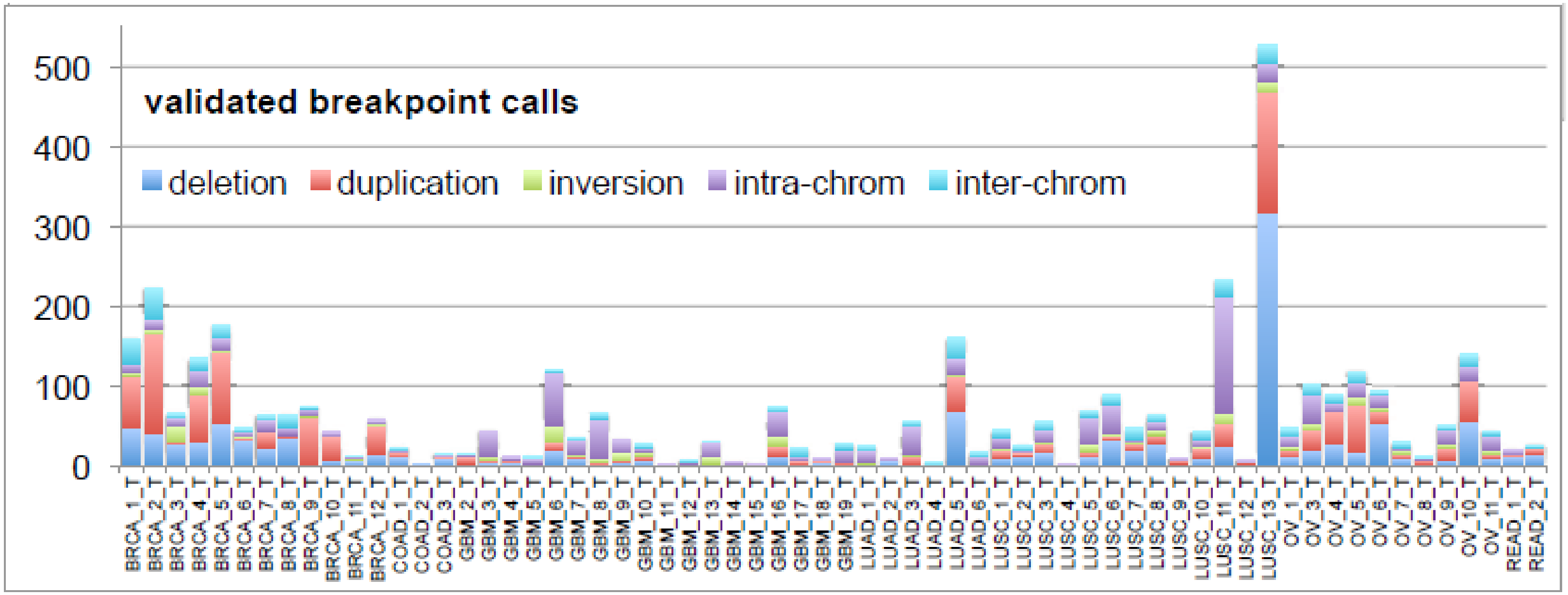
- Data of 64 tumor and matched normal samples taken from TCGA of different cancer types.
- Total of 6179 breakpoints identified using HYDRA

Malhorta et al (2013) data

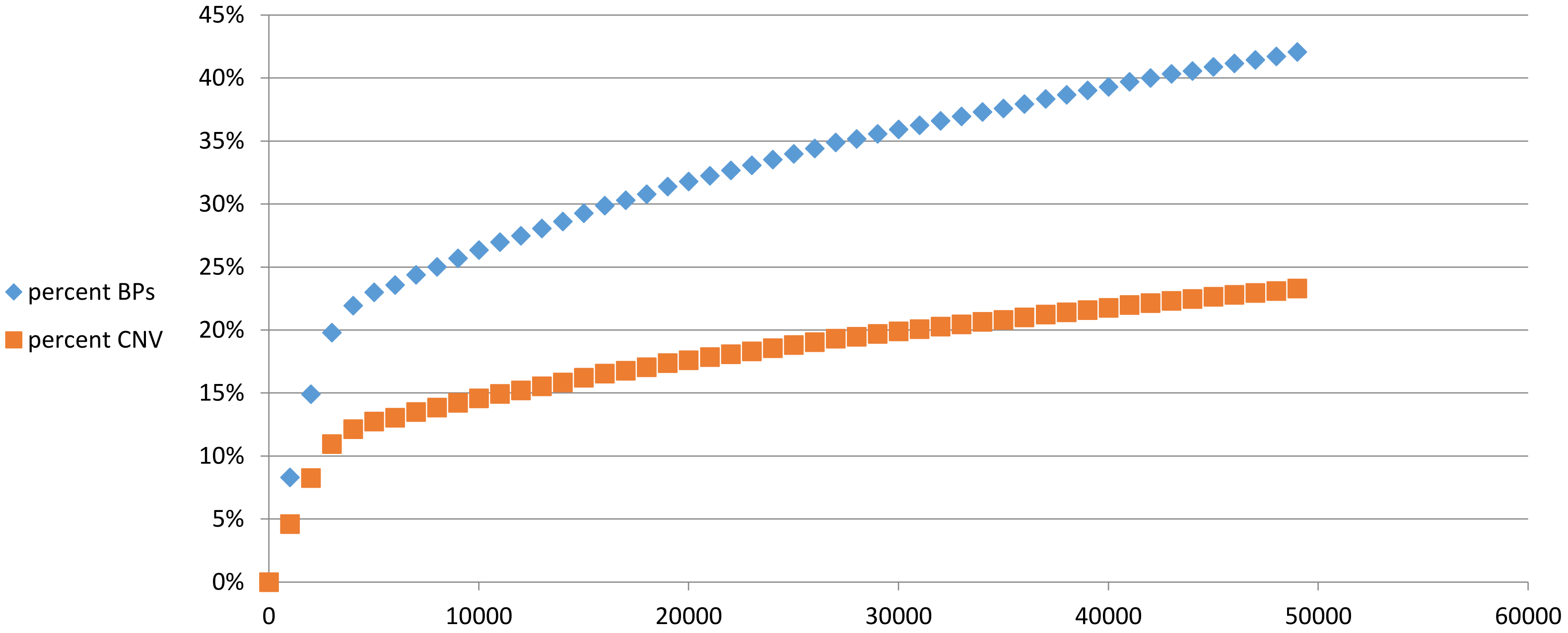
- Data of 64 tumor and matched normal samples taken from TCGA of different cancer types.
- Total of 6179 breakpoints identified using HYDRA
- 22321 CNV’s from CGH array data

Malhorta et al (2013) data

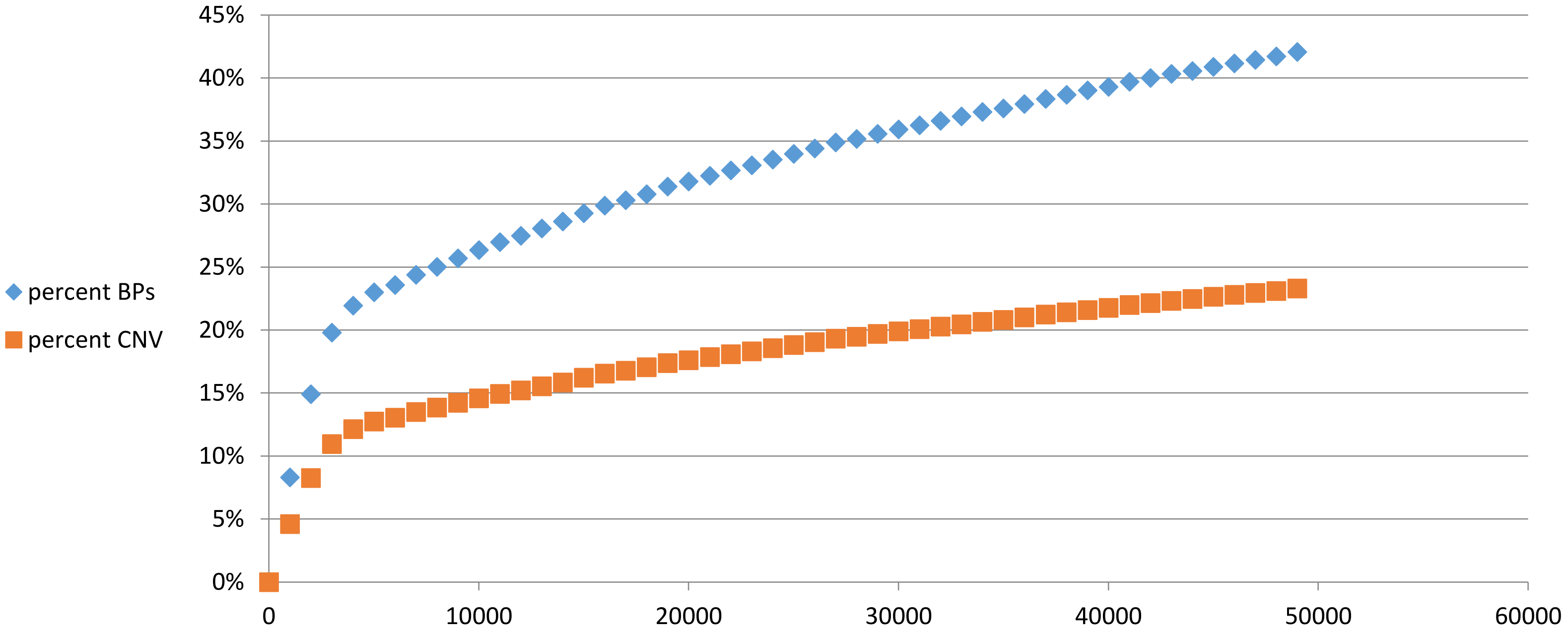
- Data of 64 tumor and matched normal samples taken from TCGA of different cancer types.
- Total of 6179 breakpoints identified using HYDRA
- 22321 CNV’s from CGH array data



CNV to BP agreement rate



CNV to BP agreement rate



With a window of 3000 bases – 20% of breakpoints and 11% of CNV agree.

Why is the support rate so low?

- Data is noisy
- Diploidity
- Not all breakpoints cause changes in CN
- Complex rearrangements (eg chromotripsis)

Some samples have higher agreement rate and are non trivial.

Example: COAD_1, chr 3

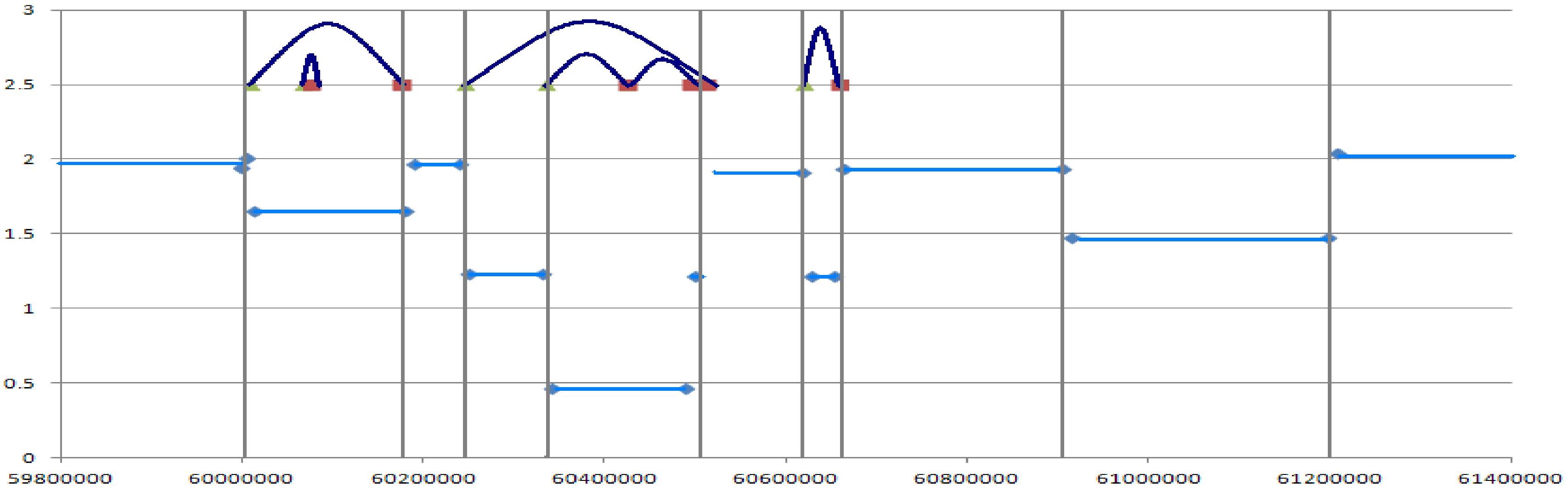
Breakpoints:

chrom	pos1	pos2	chrom2	pos3	pos4	support	strand1	strand2	variantClass
3	60009834	60009999	3	60174480	60174642	4	+	-	DEL
3	60067795	60067974	3	60076102	60076309	4	+	-	DEL
3	60246396	60246538	3	60510688	60510817	4	+	-	DEL
3	60335087	60335302	3	60423401	60423651	7	+	-	DEL
3	60424798	60424924	3	60494986	60495132	6	+	-	DEL
3	60619732	60619892	3	60658637	60658840	4	+	-	DEL

CNV:

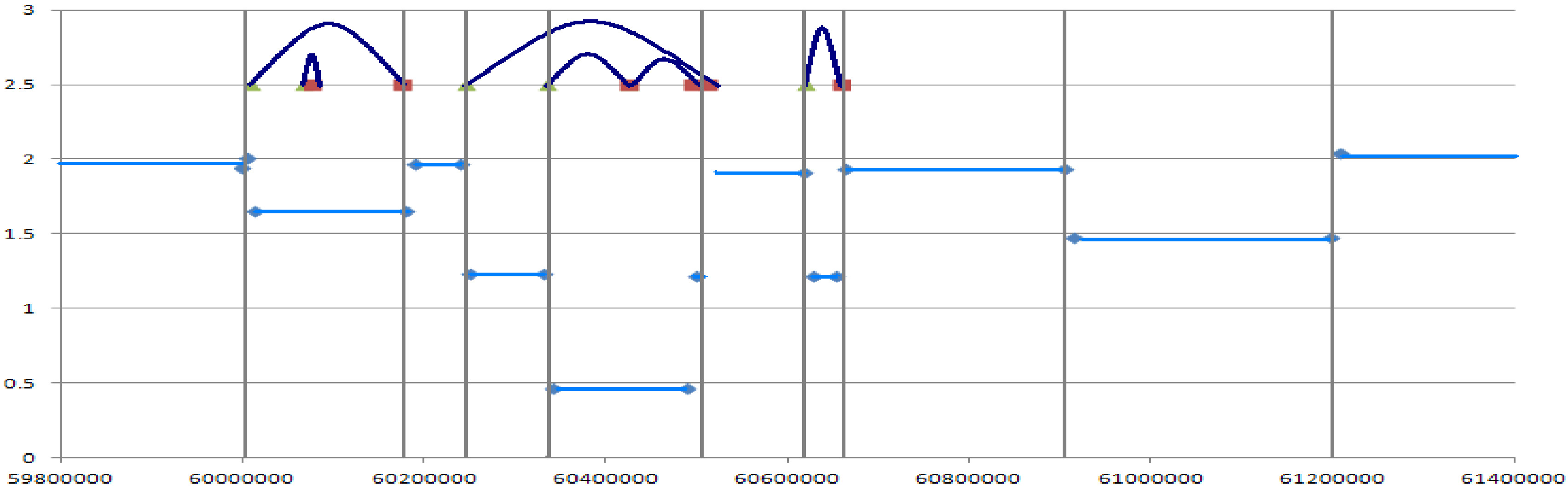
Chromosome	Start	End	Left Segment Copy Number	Right Segment Copy Number
3	4406420	4416420	1.95265	2.49525
3	60004169	60014169	2.01003	1.65659
3	60180509	60190509	1.65659	1.96985
3	60240613	60250613	1.96985	1.23601
3	60330752	60340752	1.23601	0.467678
3	60489487	60499487	0.467678	1.21906
3	60618204	60628204	1.91301	1.225
3	60653678	60663678	1.225	1.94135
3	60904531	60914531	1.94135	1.47669
3	61198109	61208109	1.47669	2.04196
3	100439625	100449625	2.98631	2.01899

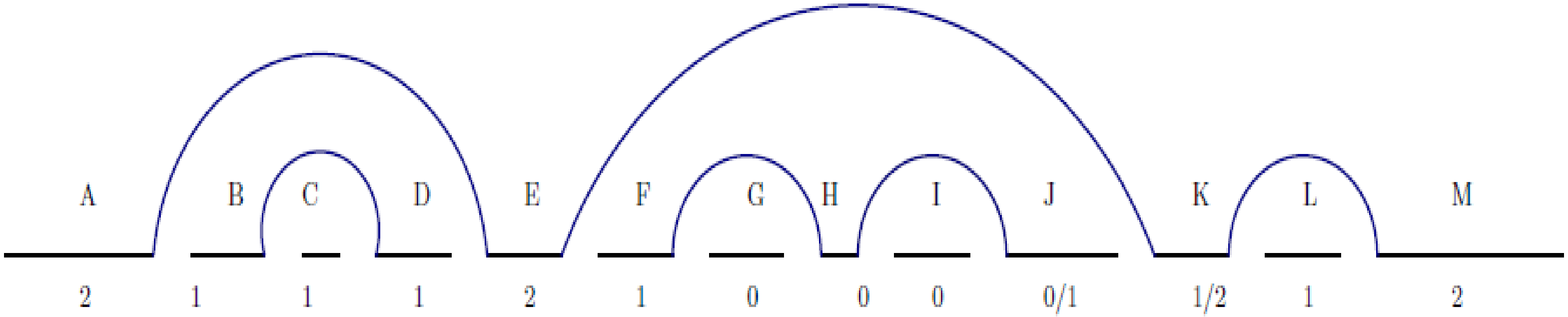
COAD_1, Chr 3

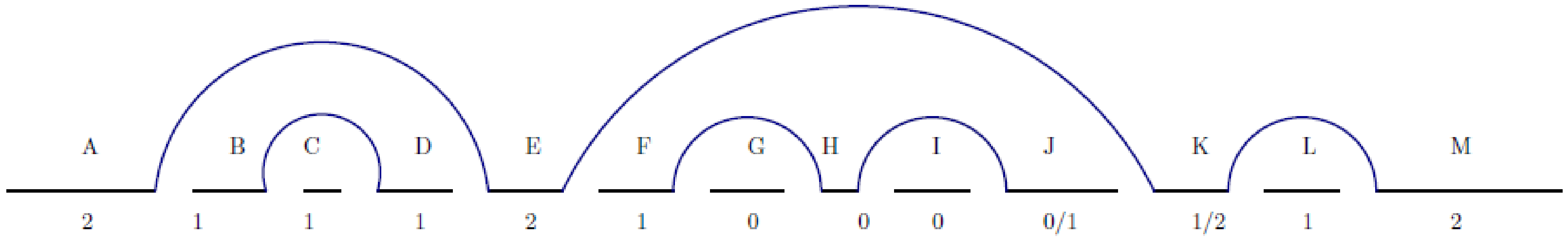


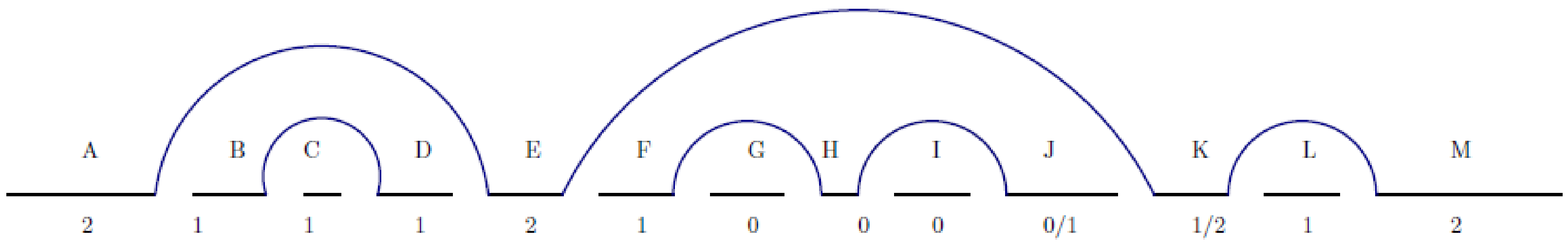
3
2.5
2
1.5
1
0.5
0
59800000
60000000
60200000
60400000
60600000
60800000
61000000
61200000
61400000

COAD_1, Chr 3

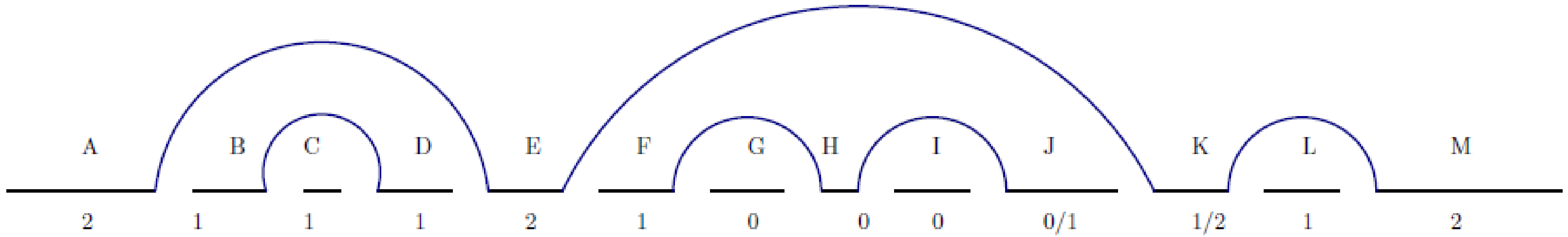

3
2.5
2
1.5
1
0.5
0
59800000
60000000
60200000
60400000
60600000
60800000
61000000
61200000
61400000


A
B
C
D
E
F
G
H
I
J
K
L
M
2
1
1
1
2
1
0
0
0
0/1
1/2
1
2

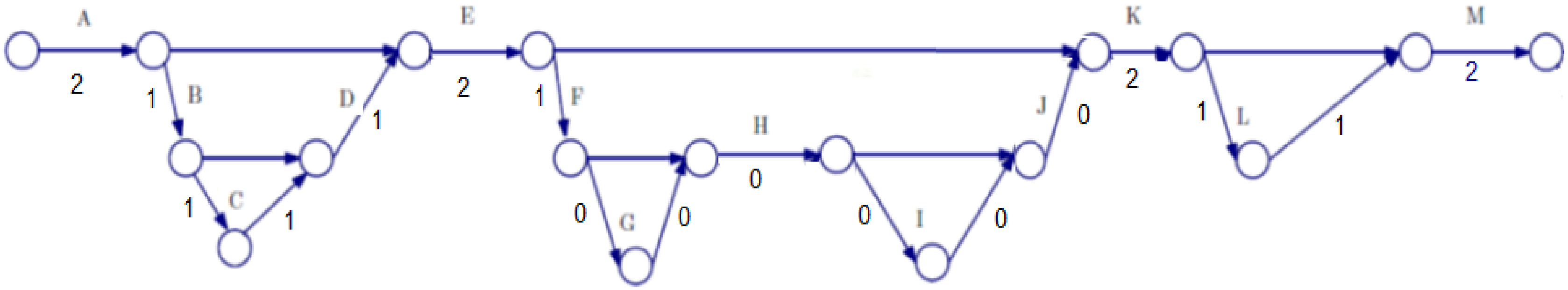

A
B
C
D
E
F
G
H
I
J
K
L
M
2
1
1
1
2
1
0
0
0
0/1
1/2
1
2



- This can be represented by the following directed graph:



- This can be represented by the following directed graph:



Formally

Let G=(V,E) be a directed graph constructed from the data as follows:

- Each adjoining of two reference segments on the breakpoint graph is represented by a node in V.
- $E = E_I \cup E_v$ is the union of both interval (segments) and variant (breakpoint) edges on the breakpoint graph.

Let $f: E \rightarrow R$ be a copy number function derived from the data.

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1 \le i \le k} = (e_1, e_2 \dots)$
   $e_i = (u, v) \Leftrightarrow e_{i+1} = (v, w)$

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1 \le i \le k} = (e_1, e_2 \dots)$
   $e_i = (u, v) \Leftrightarrow e_{i+1} = (v, w)$
2. Minimize:
   $$\sum_{e} \left(f(e) - \sum_{p} C_p(e) \right)^2$$

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1 \le i \le k} = (e_1, e_2 \ldots)$
   $e_i = (u, v) \Leftrightarrow e_{i+1} = (v, w)$
2. Minimize:
   $$\sum_e \left(f(e) - \sum_p C_p(e) \right)^2$$

The constraint can be formalized as a flow constraint on a directed graph.

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1\leq i\leq k} = (e_1, e_2 \dots)$
   $e_i = (u,v) \Leftrightarrow e_{i+1} = (v,w)$
2. Minimize:
$$\sum_e \left(f(e) - \sum_p C_p(e) \right)^2$$

The constraint can be formalized as a flow constraint on a directed graph.

For the target function:

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1\leq i\leq k} = (e_1, e_2 \ldots)$
   $e_i = (u,v) \Leftrightarrow e_{i+1} = (v,w)$
2. Minimize:
   $$\sum_e \left(f(e) - \sum_p C_p(e) \right)^2$$

The constraint can be formalized as a flow constraint on a directed graph.

For the target function:

- Quadratic programming (no guaranteed feasible solution)

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1 \le i \le k} = (e_1, e_2 \ldots)$
   $e_i = (u, v) \Leftrightarrow e_{i+1} = (v, w)$
2. Minimize:
   $$\sum_e \left(f(e) - \sum_p C_p(e) \right)^2$$

The constraint can be formalized as a flow constraint on a directed graph.

For the target function:

- Quadratic programming (no guaranteed feasible solution)
- Use absolute value (implemented in CPLEX)

We want to find a collection of $k \in \{1,2,3\}$ paths on G s.t. the number of times we traverse each edge e is as close as possible to f(e).

1. $P_{1 \le i \le k} = (e_1, e_2 \dots)$
   $e_i = (u, v) \Leftrightarrow e_{i+1} = (v, w)$
2. Minimize:
   $$\sum_e \left(f(e) - \sum_p C_p(e) \right)^2$$

The constraint can be formalized as a flow constraint on a directed graph.

For the target function:

- Quadratic programming (no guaranteed feasible solution)
- Use absolute value (implemented in CPLEX)
- Linearize the target function using discretization and a truth table.

Since interval edges are longer than breakpoint edges and differ in size, we would like to add a constant weight function to act as a penalty for “skipping” longer segments.

Since interval edges are longer than breakpoint edges and differ in size, we would like to add a constant weight function to act as a penalty for “skipping” longer segments.

1. $P_{1 \leq i \leq k} = (e_1, e_2 \ldots)$
   $e_i = (u, v) \Leftrightarrow e_{i+1} = (v, w)$
2. Minimize:
   $$\sum_e \left| f(e) - \sum_p C_p(e) \right|^2 \cdot w(e)$$

Future work:

Future work:

- The support score for breakpoint edges should somehow be normalized together with the CN of interval edges.

Future work:

- The support score for breakpoint edges should somehow be normalized together with the CN of interval edges.
- Statistical model for the weight function

Future work:

- The support score for breakpoint edges should somehow be normalized together with the CN of interval edges.
- Statistical model for the weight function
- Copy number for reference edges as well (nodes in the graph)