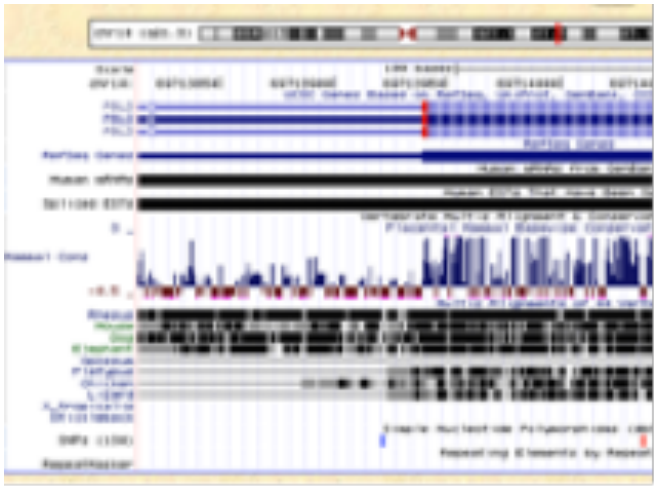
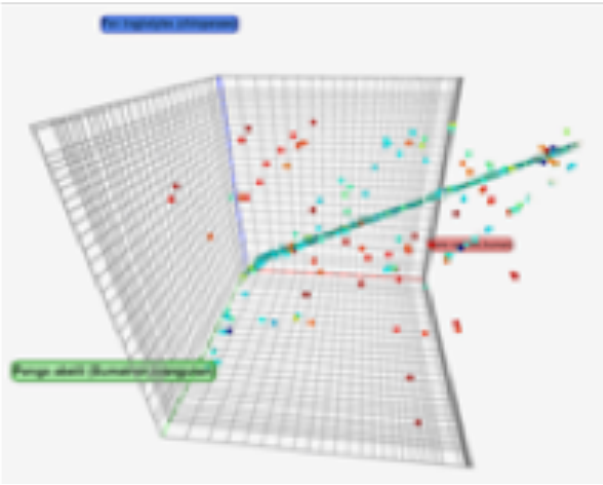
Computational Genomics

Computational Arithmetics II





Computational Arithmetics

Genomic Intervals

About JOIN INTERVALS

Dataset 1

ctg15	10	49	Feature1
ctg15	70	119	Feature2
ctg15	170	209	Feature3
ctg15	180	229	Feature4

Dataset 2

ctg15	80	109	FeatureA
ctg15	150	199	FeatureB
ctg15	250	289	FeatureC
ctg15	270	309	FeatureD

Only records that are joined (INNER JOIN)

ctg15	70	119	Feature2	ctg15	80	109	FeatureA
ctg15	170	209	Feature3	ctg15	150	199	FeatureB
ctg15	180	229	Feature4	ctg15	150	199	FeatureB

All records of first dataset

ctg15	10	49	Feature1	.	.	.	.
ctg15	70	119	Feature2	ctg15	80	109	FeatureA
ctg15	170	209	Feature3	ctg15	150	199	FeatureB
ctg15	180	229	Feature4	ctg15	150	199	FeatureB

All records of second dataset

ctg15	70	119	Feature2	ctg15	80	109	FeatureA
ctg15	170	209	Feature3	ctg15	150	199	FeatureB
ctg15	180	229	Feature4	ctg15	150	199	FeatureB
.	.	.	.	ctg15	250	289	FeatureC
.	.	.	.	ctg15	270	309	FeatureD

All records of both datasets

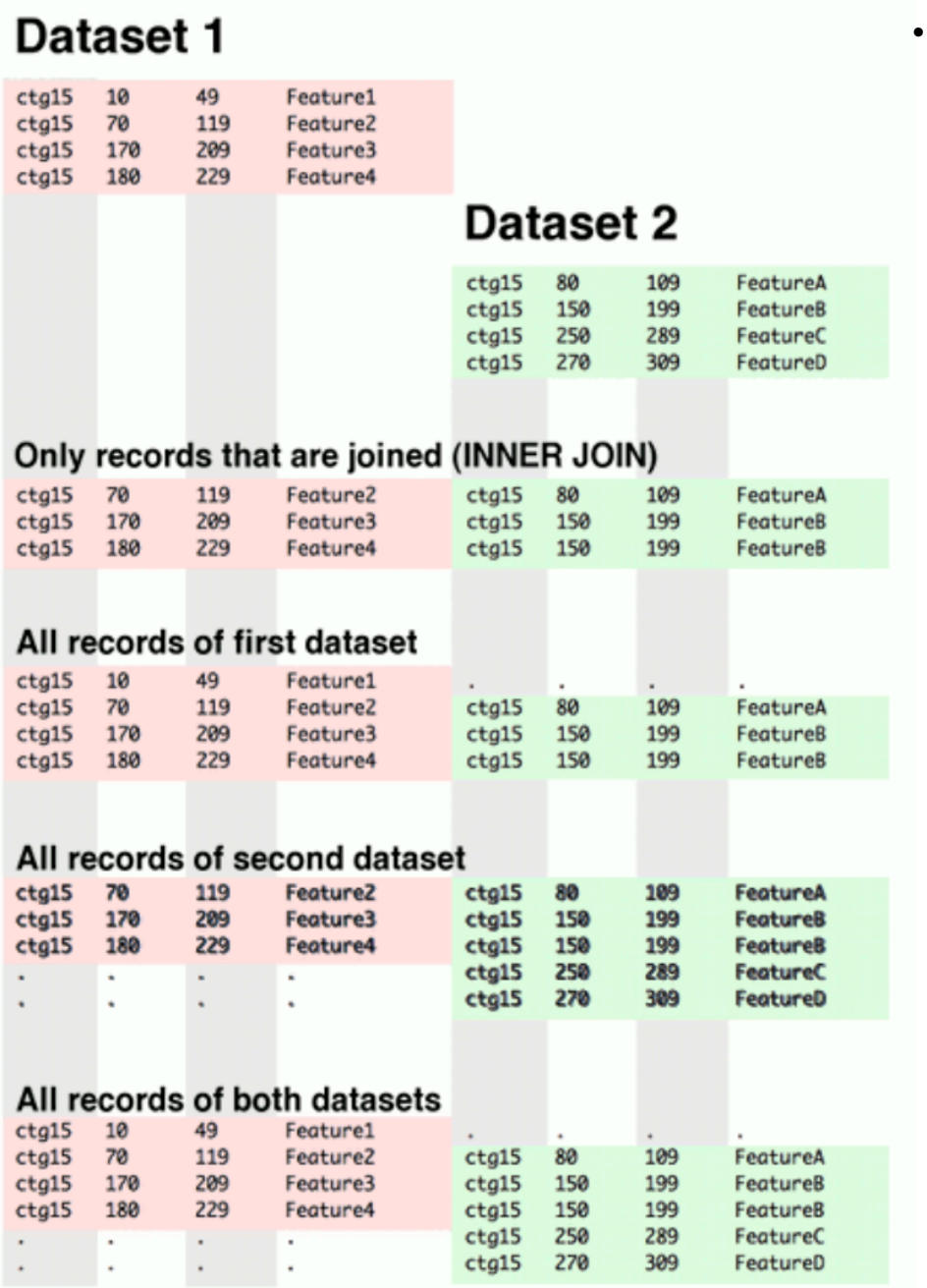
ctg15	10	49	Feature1	.	.	.	.
ctg15	70	119	Feature2	ctg15	80	109	FeatureA
ctg15	170	209	Feature3	ctg15	150	199	FeatureB
ctg15	180	229	Feature4	ctg15	150	199	FeatureB
.	.	.	.	ctg15	250	289	FeatureC
.	.	.	.	ctg15	270	309	FeatureD

- The join operation is similar to joins done by database management systems such as MySQL. Join looks at two datasets of intervals, and joins them based on interval overlap. Any interval in the second dataset that overlaps an interval in the first dataset will be appended to the line from the first dataset and output.

Computational Arithmetics

Genomic Intervals

About JOIN INTERVALS



- Join allows a minimum overlap to be specified. Intervals must exceed the minimum overlap to be joined. Several types of join can be done. These are specified by the drop-down list labeled Join Type:
  - Return only records that are joined will only return intervals in the first query that overlap and are joined to an interval in the second query. For users of SQL databases, this is similar to an “INNER JOIN”.
  - Return all records of first query (fill null with '.') returns all intervals from the first query. Any interval in the first query that does not join an interval in the second query will have the extra fields padded with a period (.).
  - Return all records of second query (fill null with '.') returns all intervals from the second query. Any interval in the second query that is not joined to an interval in the first query will have fields filled in with a period (.). Because the intervals are filled in with a period, this may output an invalid interval dataset. Further operations on the resulting dataset may not be possible, since chromosome, start, and end will be replaced with a period, which is not a valid value.
  - Return all records of both queries (fill nulls with a '.') returns all of the intervals from both queries. Intervals that do not join have fields filled in with a period (.). As with the previous option, this could result in an interval dataset with a period in the chromosome, start, and end fields. This would result in a dataset that cannot have any further operations performed on it.

Computational Arithmetics

Genomic Intervals

About JOIN INTERVALS

Query 1:

chr1	10	100	Query1.1
chr1	500	1000	Query1.2
chr1	1100	1250	Query1.3

Query 2:

chr1	20	80	Query2.1
chr1	2000	2204	Query2.2
chr1	2500	3000	Query2.3

Input

Output

(Return only records that are joined)

chr1	10	100	Query1.1	chr1	20	80	Query2.1

Return only records that are joined (INNER JOIN)
Return all records of first query (fill null with ".")
Return all records of second query (fill null with ".")
Return all records of both queries (fill nulls with ".")

(Return all records of first query)

chr1	10	100	Query1.1	chr1	20	80	Query2.1
chr1	500	1000	Query1.2	.	.	.	.
chr1	1100	1250	Query1.3	.	.	.	.

Return only records that are joined (INNER JOIN)
Return all records of first query (fill null with ".")
Return all records of second query (fill null with ".")
Return all records of both queries (fill nulls with ".")

(Return all records of second query)

chr1	10	100	Query1.1	chr1	20	80	Query2.1
.	.	.	.	chr1	2000	2204	Query2.2
.	.	.	.	chr1	500	3000	Query2.3

Return only records that are joined (INNER JOIN)
Return all records of first query (fill null with ".")
Return all records of second query (fill null with ".")
Return all records of both queries (fill nulls with ".")

(Return all records of both queries)

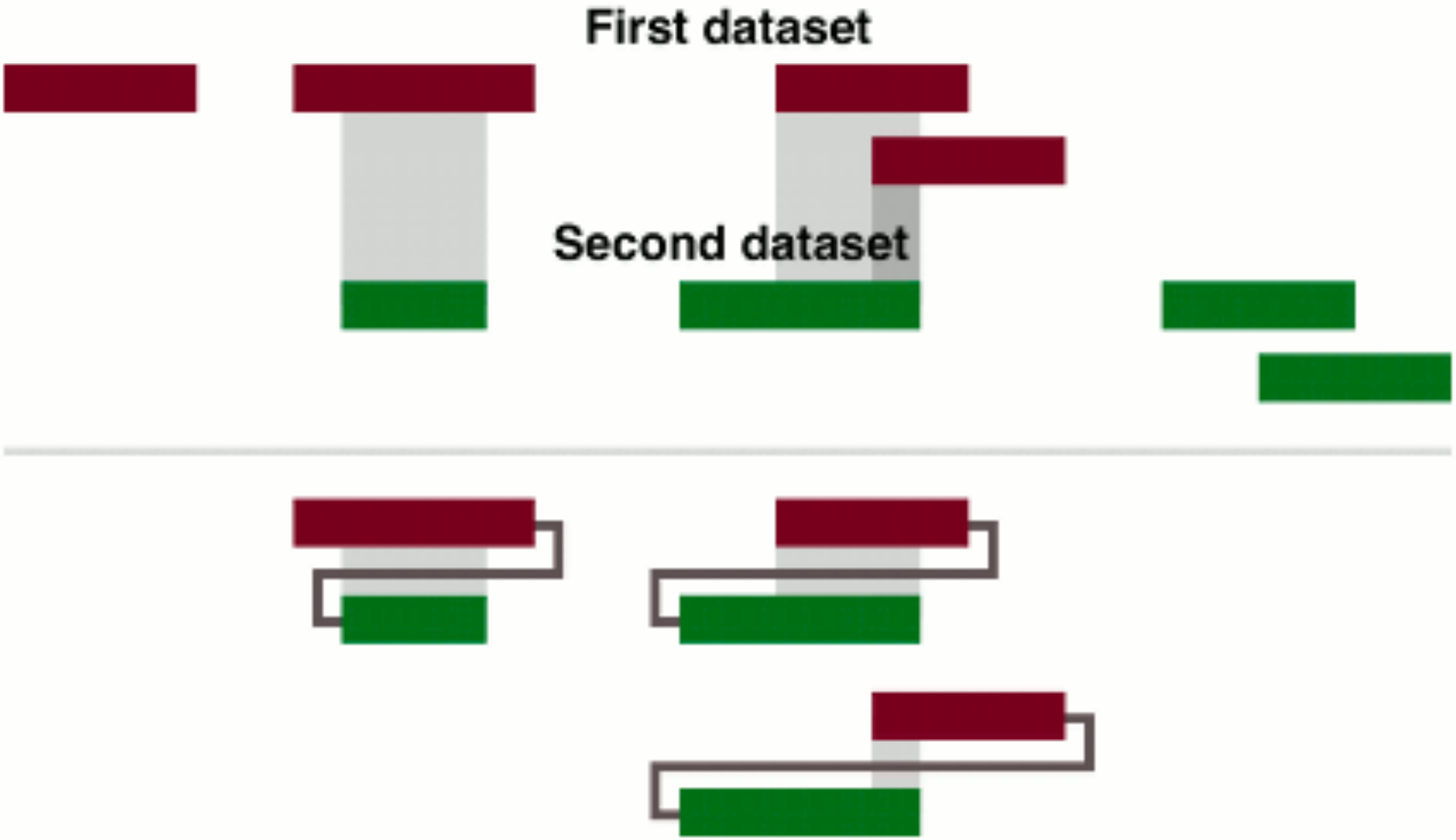
chr1	10	100	Query1.1	chr1	20	80	Query2.1
chr1	500	1000	Query1.2	.	.	.	.
chr1	1100	1250	Query1.3	.	.	.	.
.	.	.	.	chr1	2000	2200	Query2.2
.	.	.	.	chr1	2500	3000	Query2.3

Return only records that are joined (INNER JOIN)
Return all records of first query (fill null with ".")
Return all records of second query (fill null with ".")
Return all records of both queries (fill nulls with ".")

Computational Arithmetics

Genomic Intervals

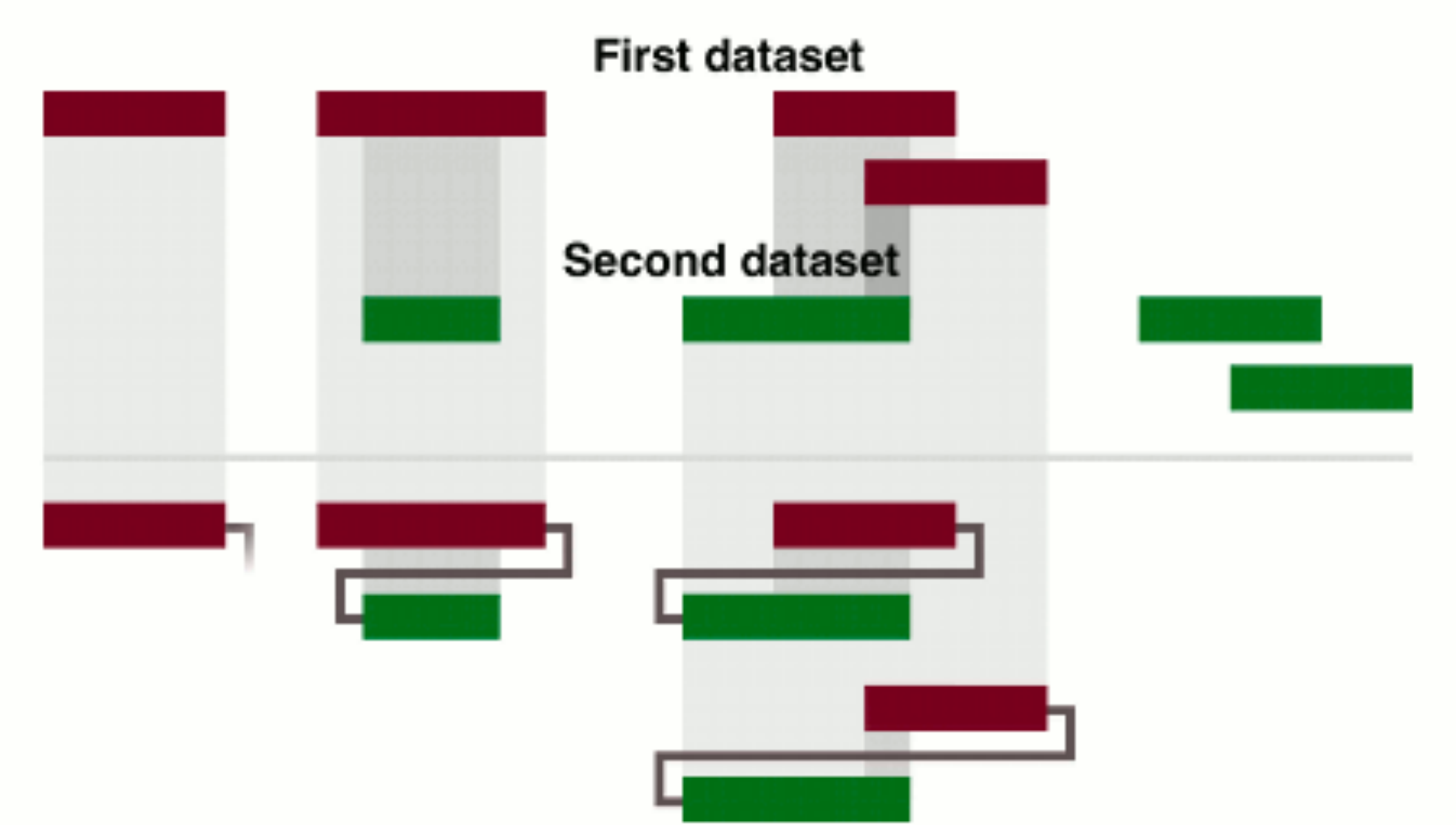
Only records that are joined (inner join):



Computational Arithmetics

Genomic Intervals

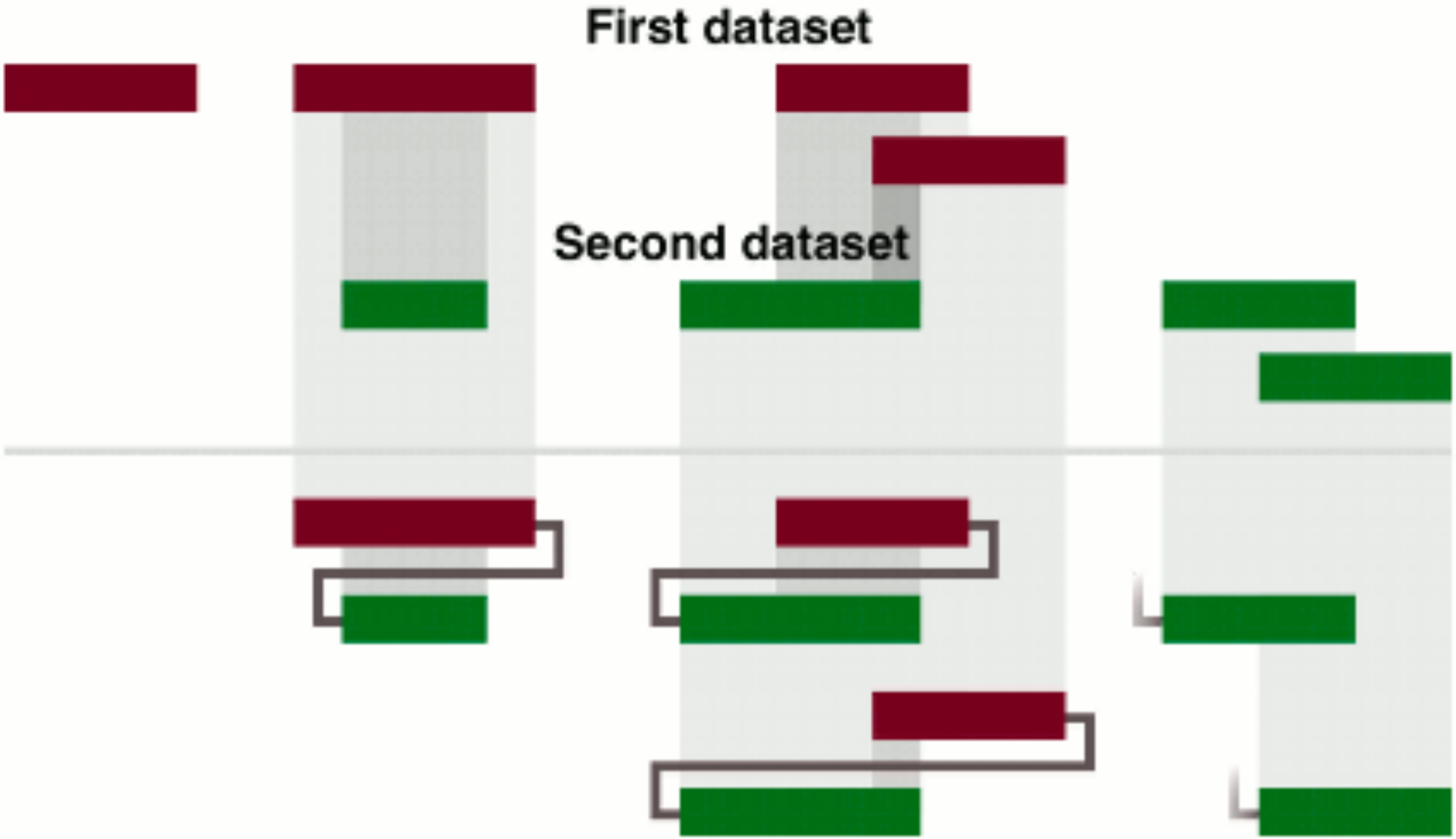
All records of first dataset:



Computational Arithmetics

Genomic Intervals

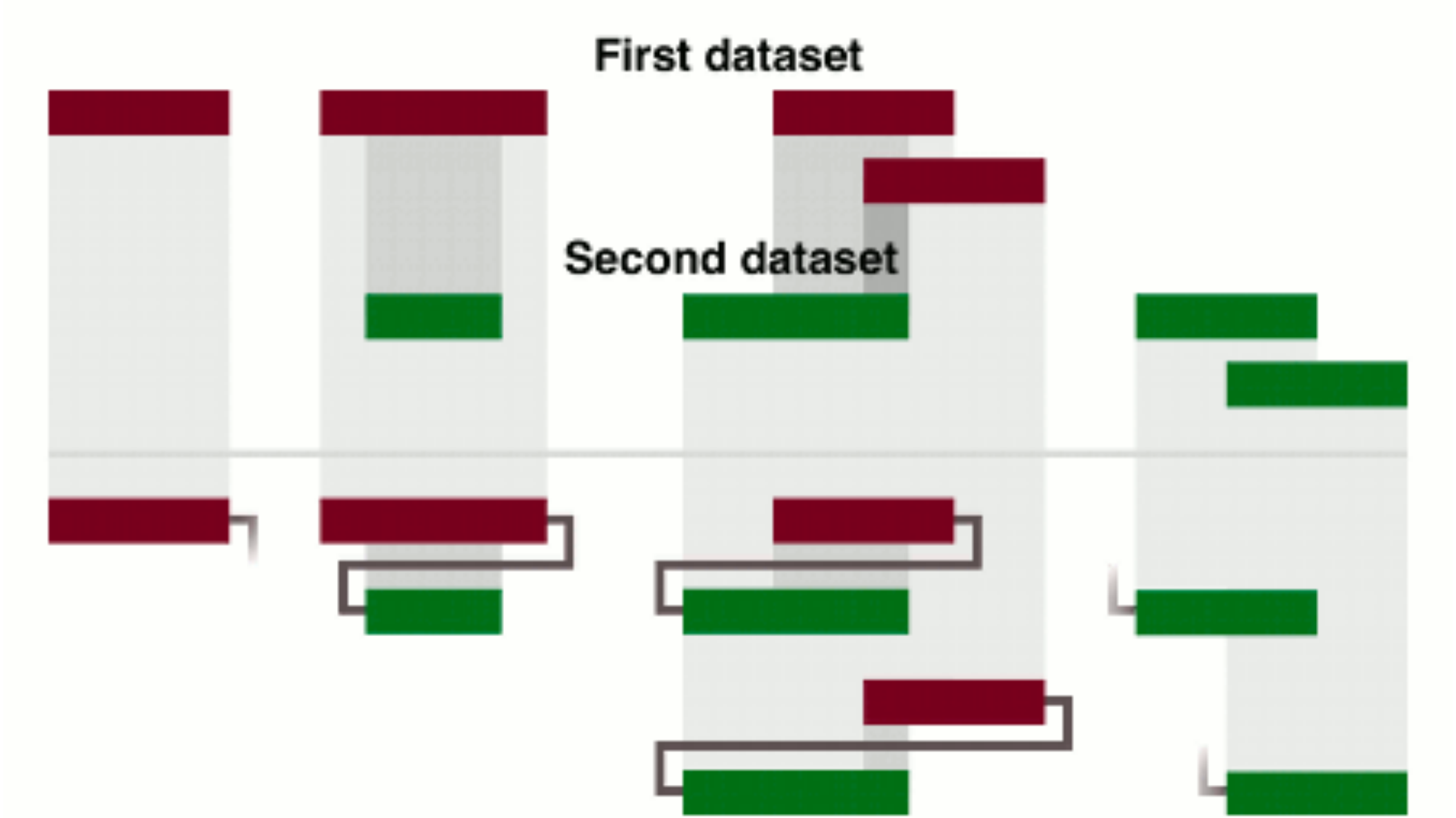
All records of second dataset:



Computational Arithmetics

Genomic Intervals

All records of both datasets:



Computational Arithmetics

Genomic Intervals

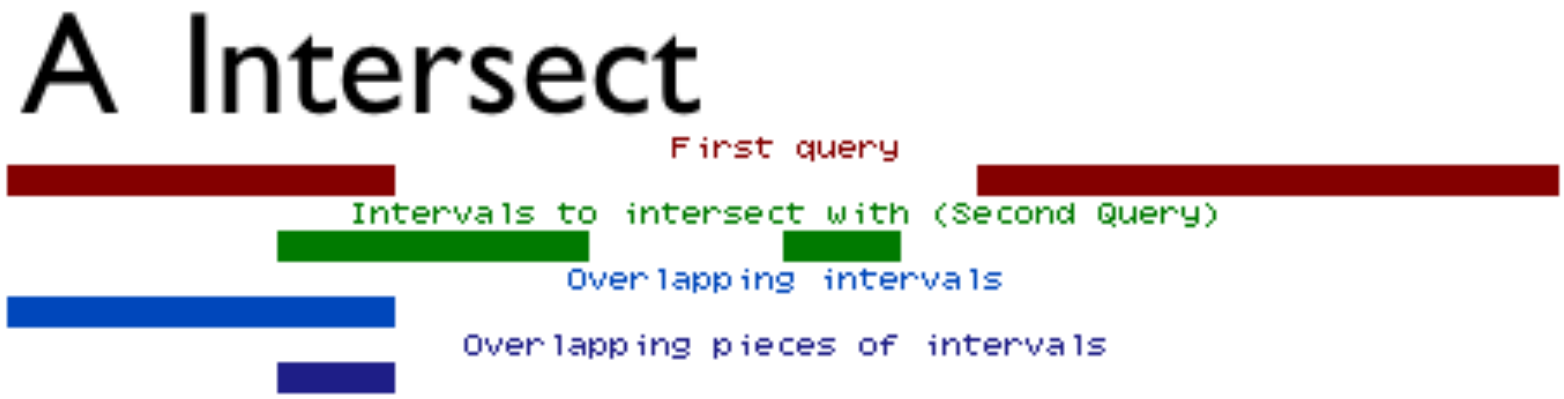
Identifying Human Coding Exons with Highest SNPs Density

1. Retrieve GENCODE Coding Exons Human Chromosome 22 (hg38) in BED format
    1. **Rename to ‘CExonsChr22hg38’**
2. Retrieve SNPs Whole Gene for Human Chromosome 22 (hg38) in BED format
    1. **Rename to ‘SNPsChr22hg38’**
3. Run ‘Operate on Genomic Intervals’ > ‘Join’
    1. CExonsChr22hg38
    2. SNPsChr22hg38
    3. Default parameters (1 bp min overlap and INNER JOIN)
4. Run ‘Join_Subtract_and_Group’ > Group
    1. Group by C04 (Name)
    2. ‘Add New Operation’
    3. Count on C04
5. Run ‘Join_Subtract_and_Group’ > ‘Join two Datasets side by side on a specified field’
    1. CExonsChr22hg38 on C04
    2. ‘4: Group on data 3’ on C01
    3. Default parameters
6. Run ‘Text_Manipulation’ > ‘Cut’ (re-order)
    1. Cut: c1,c2,c3,c4,c8,c6
    2. This will produce a valid BED format file
    3. Be Careful converting Interval to Bed…
7. Sort by c5 (Descending), to identify the Coding Exon with highest number of SNPs
8. Sort by c2 (default) and **Rename to ‘CExonsHighSNPs.bed’**

Computational Arithmetics

Genomic Intervals

Intersection



- Intersect allows for the intersection of two queries to be found. The intersect tool can output either the entire set of intervals from the first dataset that overlap the second dataset (e.g. all exons containing repeats), or just the intervals representing the overlap between the two datasets (e.g. all regions that are both exonic and repetitive; see Figure above).

- When finding entire intervals (by setting Return to Overlapping Intervals), the order of the datasets is important. The operation will output all of the intervals in the first query that overlap any interval in the second query. It can also be thought of as a filter: intervals that do not overlap any interval in the second query will be removed.

- When finding pieces of intervals, or the regions representing the overlap between the two datasets (by setting Return to Overlapping Pieces of Intervals), the output will be the intervals of the first dataset with the non-overlapping subregions removed.

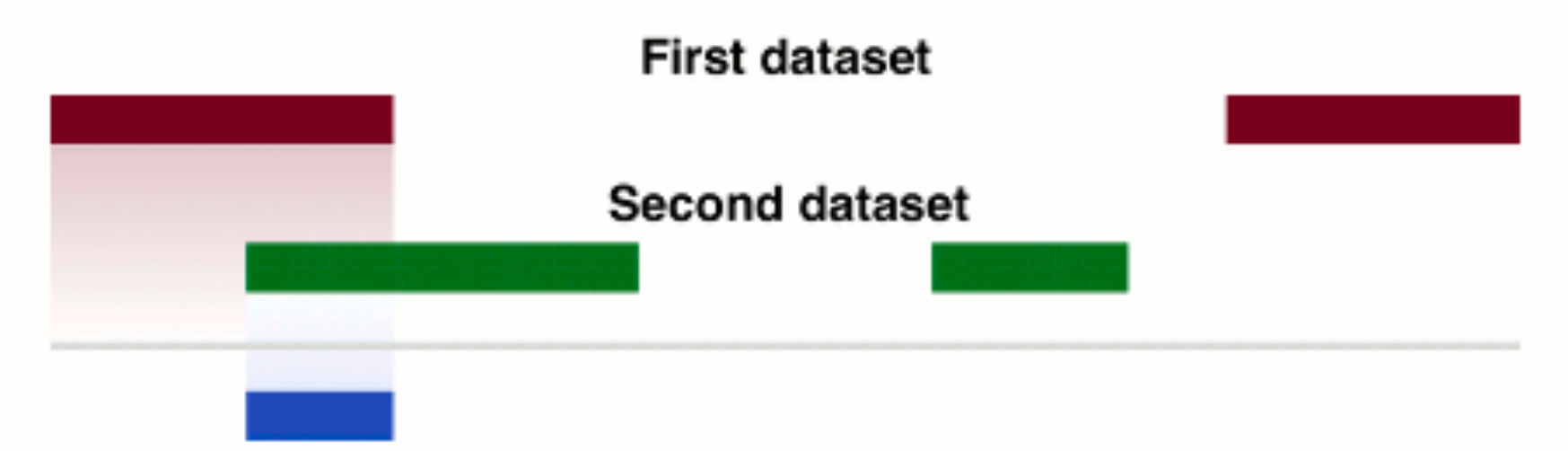
Computational Arithmetics

Genomic Intervals

Intersection

- **Overlapping Intervals:**

- **Overlapping Pieces of Intervals:**



Computational Arithmetics

Genomic Intervals

Intersection

1. Retrieve Coding Exons from Human Chromosome 22 (hg38)
    1. **Rename ‘CExonsChr22hg38’**

2. Retrieve Repeats (RepeatMasker) -Whole Gene- from Human Chromosome 22 (hg38)
    1. **Rename ‘RepeatsChr22hg38’**

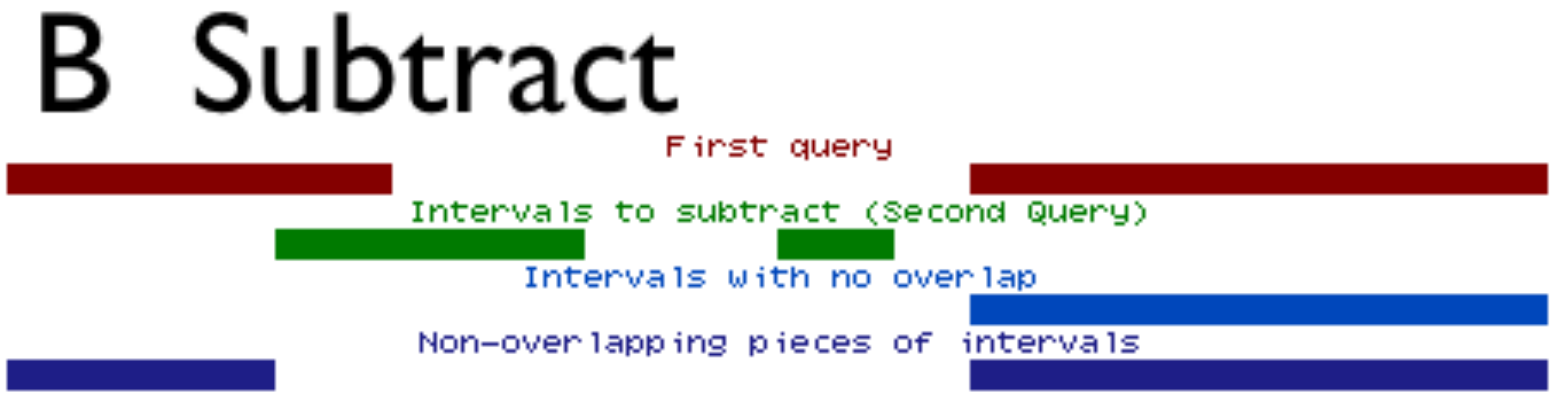
3. Intersect:
    1. Return: Overlapping Intervals
    2. First Dataset: CExonsChr22hg38
    3. Second Dataset: RepeatsChr22hg38
    4. For at least: 1 bp
    5. **Name Job: Intersect01**

4. Intersect:
    1. Return: Overlapping Pieces of Intervals
    2. First Dataset: CExonsChr22hg38
    3. Second Dataset: RepeatsChr22hg38
    4. For at least: 1 bp
    5. **Name Job: Intersect02**

Computational Arithmetics

Genomic Intervals

Subtraction



- **Subtract does the opposite of intersect. It removes the intervals or parts of intervals in the first dataset that are found in the second dataset (Figure above). Like Intersect, Subtract can treat intervals as a whole, removing or keeping entire intervals, or it can break them apart, removing overlapping subregions.**

- **As with arithmetic subtraction, the order of the datasets is important. The second dataset is subtracted from the first dataset. The output is a variation of the first dataset and all of its columns. When subtracting whole intervals (by setting Return to Intervals with no overlap), the output will be the intervals of the first dataset that do not overlap any part of intervals of the second dataset.**

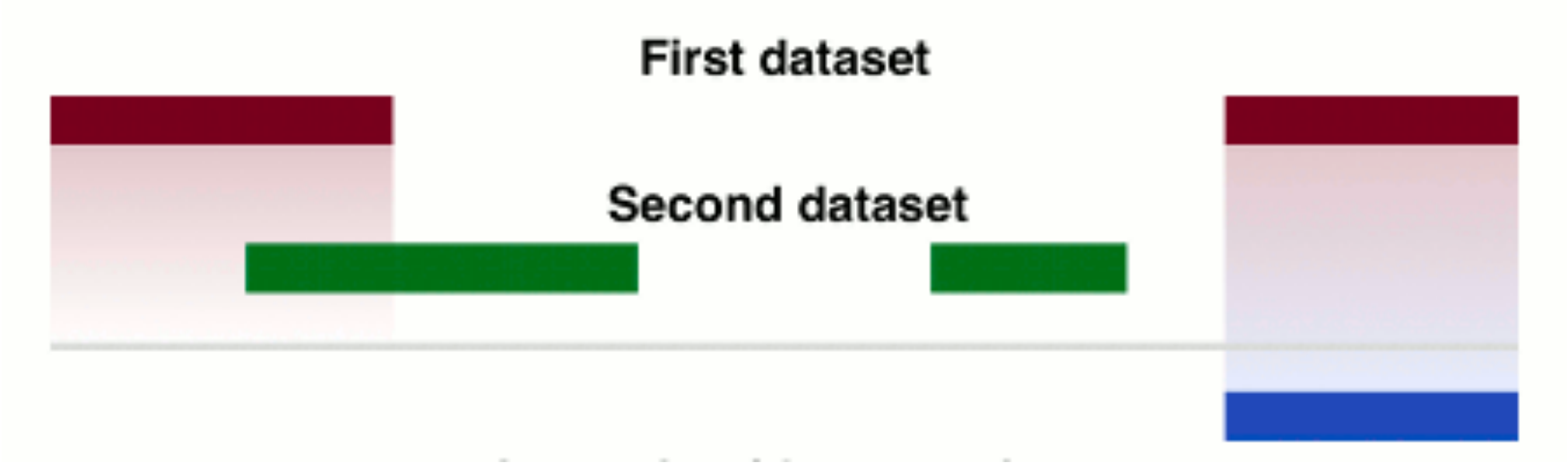
- **When subtracting overlapping subregions (by setting Return to Non-overlapping pieces of intervals), the output will be the intervals of the first dataset with the overlapping subregions removed.**

Computational Arithmetics

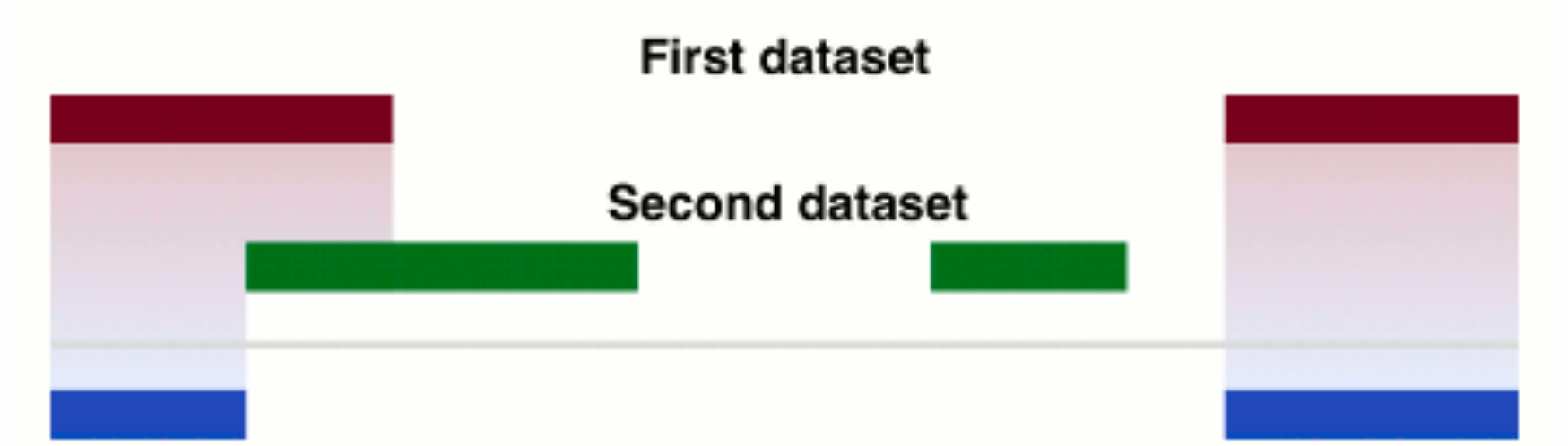
Genomic Intervals

Subtraction

- Intervals with no overlap:



- Non-overlapping pieces of intervals:



Computational Arithmetics

Genomic Intervals

Subtraction
Coding Exons From Repeats

1. Subtract01
    1. Subtract: CExonsChr22hg38
    2. From: RepeatsChr22hg38
    3. Return (of the first dataset): Intervals with no overlap
    4. Where minimal overlap is:: 1 bp
    5. **Name Job: Subtract01**

3. Subtract02
    1. Subtract: CExonsChr22hg38
    2. From: RepeatsChr22hg38
    3. Return (of the first dataset): Non-overlapping pieces of Intervals
    4. Where minimal overlap is:: 1 bp
    5. **Name Job: Subtract02**

Computational Arithmetics

Genomic Intervals

Subtraction
Repeats From Coding Exons

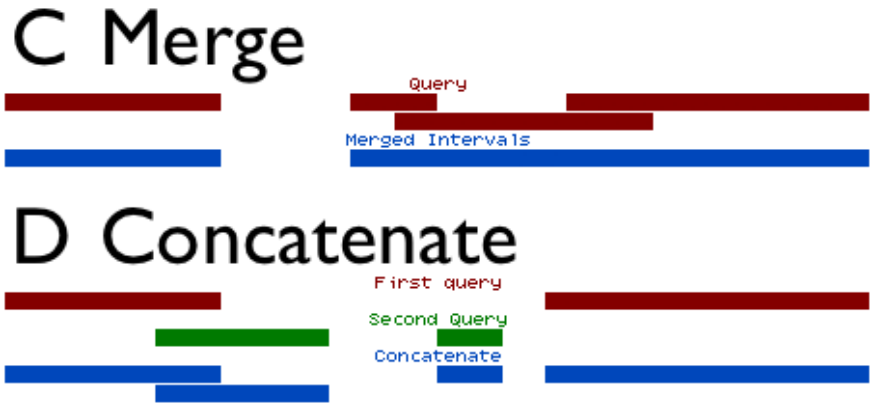
1. Subtract03
    1. Subtract: RepeatsChr22hg38
    2. From: CExonsChr22hg38
    3. Return (of the first dataset): Intervals with no overlap
    4. Where minimal overlap is:: 1 bp
    5. **Name Job: Subtract03**

3. Subtract04
    1. Subtract: RepeatsChr22hg38
    2. From: CExonsChr22hg38
    3. Return (of the first dataset): Non-overlapping pieces of Intervals
    4. Where minimal overlap is:: 1 bp
    5. **Name Job: Subtract04**

Computational Arithmetics

Genomic Intervals

Concatenation and Merging



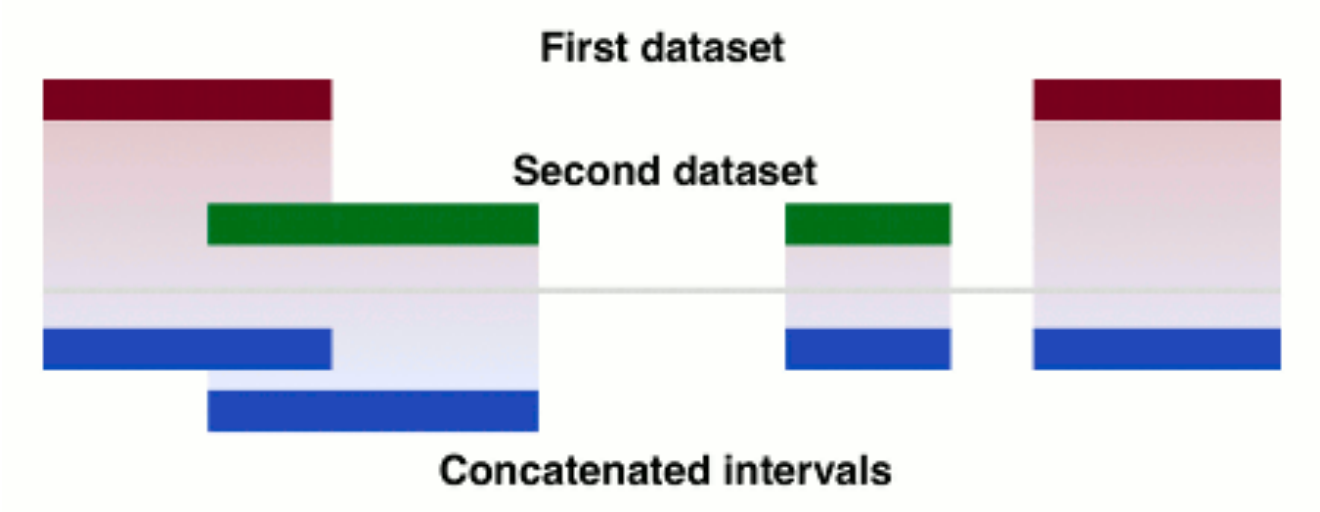
- Concatenate and Merge are analogous to addition and union (figure above). They can be used together to combine datasets and merge (or flatten) the intervals.
- Concatenate simply combines two interval datasets. The option Both queries are exactly the same filetype indicates that columns in both datasets are the same. If this option is unchecked, then the second dataset is adjusted to match the column assignments of the first. However, since the columns chromosome, start, end, and strand are the only columns used by the operations, all other columns will be replaced in the second dataset with a period(.). Typically this option is left checked, as BED files are the typical interval format used within Galaxy.
- Merge reads a dataset, and combines all overlapping intervals into single intervals. When merging intervals, all columns besides chromosome, start, and end are lost. When two intervals are combined into one, it is ambiguous what the other columns represent or which field should be carried over to the resulting interval. For this reason, all columns except for chromosome, start and end are omitted from the output.
- Concatenate combines datasets, and has the ability to combine interval datasets of different types. Merge combines overlapping intervals into single intervals. Together, the two operations can be used to combine intervals from different datasets into simple regions.

Computational Arithmetics

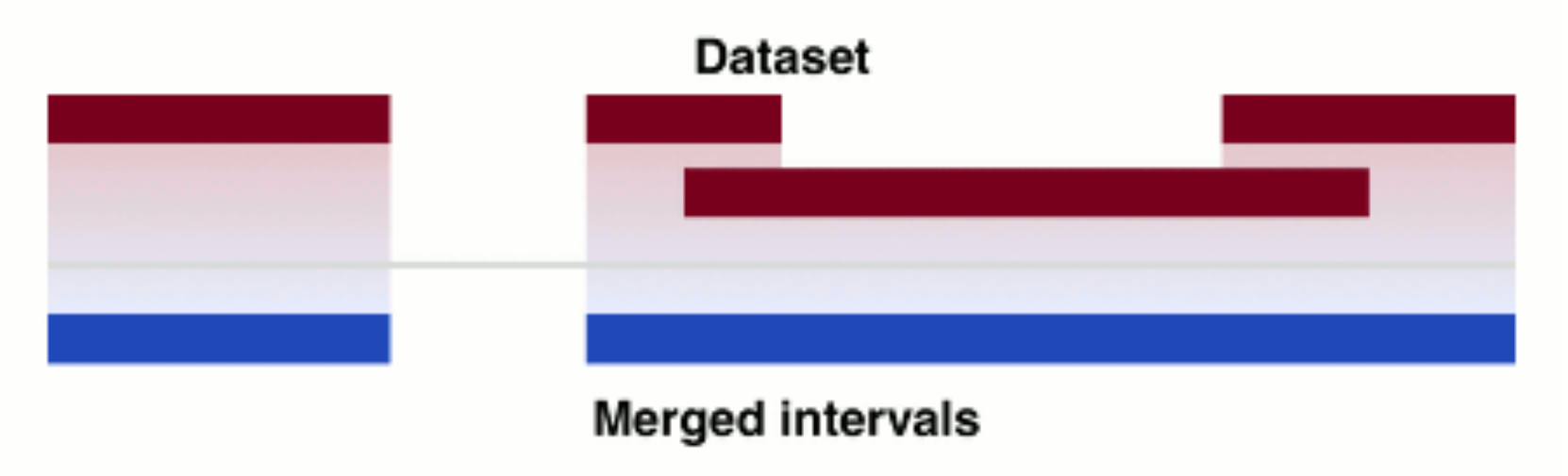
Genomic Intervals

Concatenation and Merging

- Concatenate



- Merge



Computational Arithmetics

Genomic Intervals

Concatenation and Merging

1. Concatenate
    1. First Dataset: CExonsChr22hg38
    2. With (Second Dataset): RepeatsChr22hg38
    3. **Name Job: ConcatCExonsRepeats**

2. Merge
    1. Merge overlaping regions of: ConcatCExonsRepeats
    2. BED format
    3. **Name Job: MergeConcatCExonsRepeats**

Computational Arithmetics

Genomic Intervals

Complementation

- Complement inverts a dataset (Figure above). Complement reads in all of the regions of a dataset, and outputs the regions not covered by any intervals in that dataset. The option Genome-wide complement allows for the entire genome to be complemented, regardless of whether a chromosome, contig, scaffold, etc. is represented in the query dataset. In a genome-wide complement of a dataset, any chromosome that has no intervals in the query dataset will be output in the result as the entire chromosome. In a normal complement, only the chromosomes, contigs, scaffolds, etc. that are referenced in the query dataset will be represented in the output.

- When complementing a chromosome, the length of the chromosome is needed. Galaxy uses the chromInfo tables available through the UCSC Table Browser for this information. For complements on builds not available through UCSC, a default chromosome length of 512 megabases is assumed.

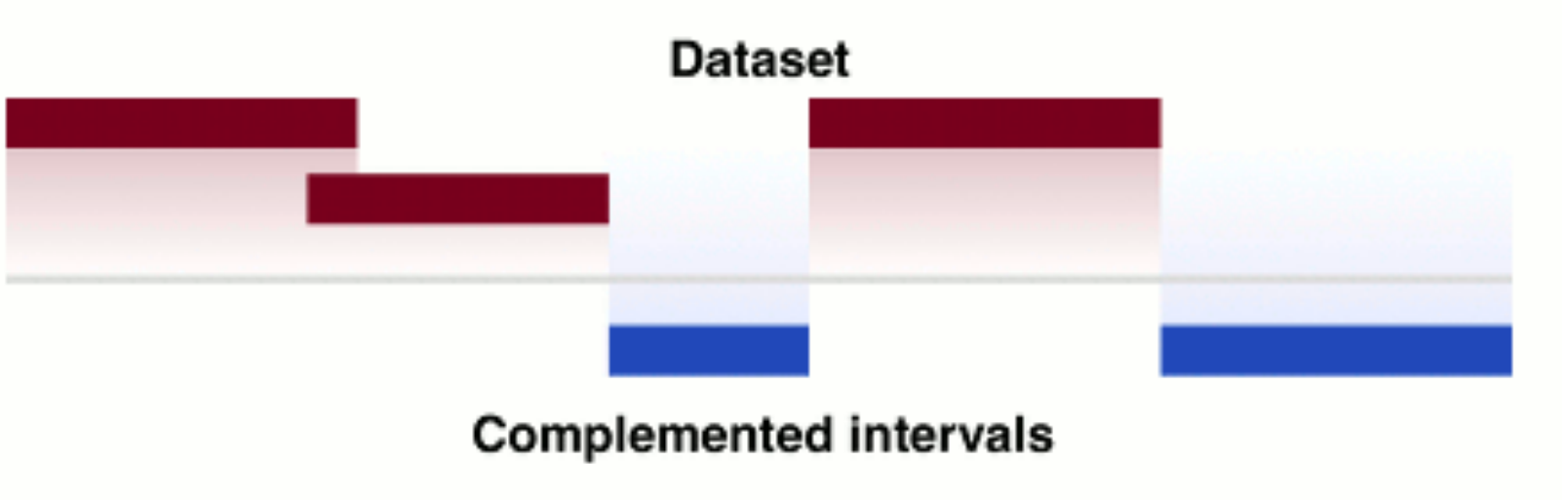
- The resulting dataset will contain intervals representing regions that are NOT transposable elements. Also, a normal complement is done in contrast to a genome-wide complement because when obtaining the simple repeats, chromosome 22 was explicitly specified, and the other chromosomes were explicitly omitted.

Computational Arithmetics

Genomic Intervals

Complementation

- **Complemention**



1. Complement
    1. First Dataset: CExonsChr22hg38
    2. **Name Job: ComplemCExonsChr22hg38**

Computational Arithmetics

Genomic Intervals

Clustering

- Cluster is one of the most versatile and powerful interval operations (figure above). Cluster finds clusters of intervals, and has a wide range of behavior depending on the options specified. The Maximum distance parameter specifies the maximum distance allowed between regions for those regions to be considered a cluster. Maximum distance can be a positive number, zero, or a negative number:
  - When maximum distance is a positive number, regions that are at most that distance from each other are considered to be a cluster.
  - When maximum distance is zero, cluster considers intervals that are touching to be a cluster. This is similar to the behavior of the merge tool, but is more flexible and specific.
- When maximum distance is a negative number, intervals that have that amount of overlap are considered to be a cluster.
- A cluster will be ignored unless it has at least as many intervals within it as specified by the parameter Minimum intervals per cluster. If this is set to 1 or lower, then all intervals, even single intervals that do not cluster with any surrounding intervals, are included in the output.

Computational Arithmetics

Genomic Intervals

Clustering

- **Find Clusters:**

- **Merge Clusters:**

Computational Arithmetics

Genomic Intervals

Clustering

1. Cluster
    1. Cluster intervals of: CExonsChr22hg38
    2. max distance between intervals: 1 (default)
    3. min number of intervals per cluster: 2 (default)
    4. Return type: Merge clusters into single intervals
    5. Name Job: **ClusterCExonsRepeatsChr22hg38_01**
    6. Merge overlaping regions of: ClusterCExonsRepeatsChr22hg38_01
    7. Name Job: **MergeClusterCExonsRepeatsChr22hg38_01**

2. Cluster
    1. Cluster intervals of: CExonsChr22hg38
    2. max distance between intervals: 1 (default)
    3. min number of intervals per cluster: 2 (default)
    4. Return type: Find cluster intervals; preserve comments and order
    5. **Name Job: ClusterCExonsRepeatsChr22hg38_02**