POLITECNICO DI MILANO

Department of Electronics, Information and Bioengineering

Genomic Computing: processing heterogeneous genomic datasets for biomedical knowledge discovery

Marco Masseroli

(joint work with Stefano Ceri and several postocs and PhD students)

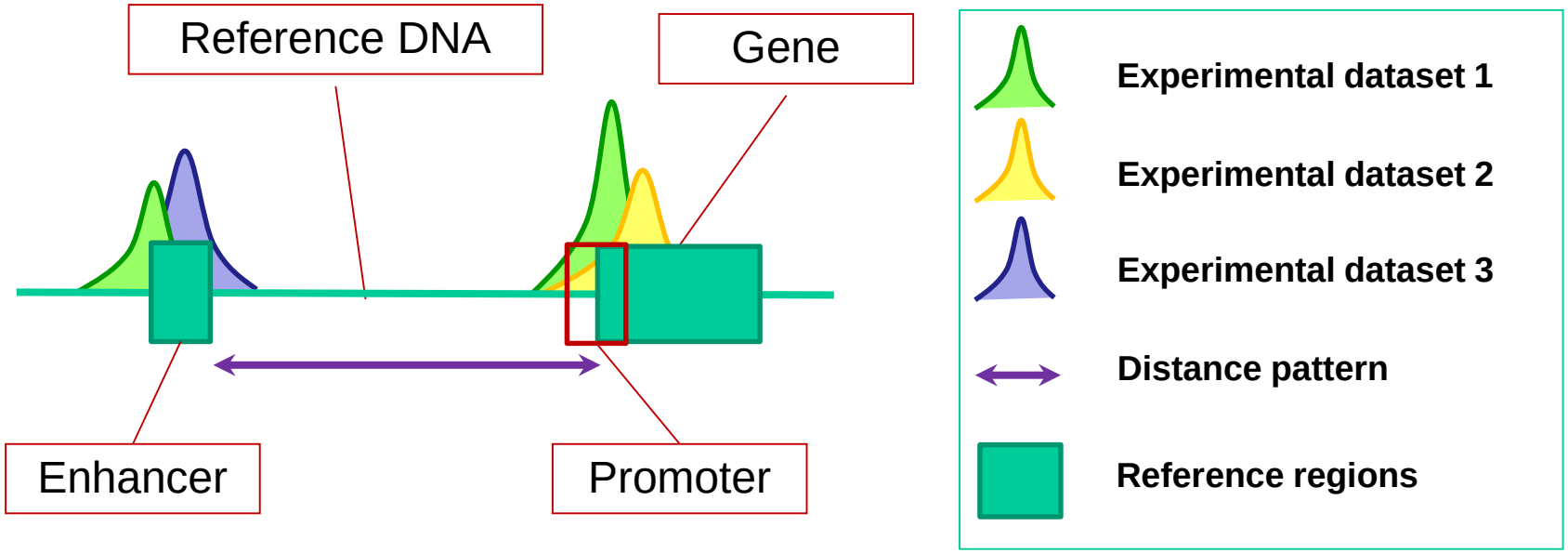
Bioinformatics Group, Politecnico di Milano, Italy

GMQL Application Examples

Genomic Computing applications
Typical GMQL applications

The language allows for queries on the genome involving large and heterogeneous datasets describing:

- Genomic signals (i.e. experiment dataset regions)
- Reference regions (e.g. TSS, genes, enhancers, promoters)
- Distance rules (e.g. the nearest enhancer that stands at least at 100 kb from the nearest gene)



Genomic Computing applications
Typical GMQL applications

- **Annotating unknown data to known annotations** (e.g. annotating transcription factors to genes)

- **Defining (epi)genomic features on the base of experimental data** (e.g. putative enhancers)

- **Searching for patterns of (epi)genomic features in experimental data and relating them to specific biological phenomena** (e.g. short-range interactions)

- **Looking for patterns of (epi)genomic features related to the tridimensional structure of the genome and relating them to specific biological phenomena** (e.g. long-range interactions)

Genomic Computing applications
GMQL examples

1. Find distal bindings in transcription regulatory regions
2. Find exons / CpG islands with somatic mutations
3. Find top ChIP-seq samples with highest number of enriched regions in promoters
4. Find promoters with highest number of ChIP-seq enriched regions
5. Combining ChIP-seq and DNase-seq data in different formats and sources
6. Counting distinct DNA mutations in patient groups
7. Combining data types: copy number variation and microRNA data
8. Combining and comprehensive processing of patients' heterogeneous omics data
9. Calling cell line-specific active enhancers
10. Mapping pairs of genes on topologically associating domains

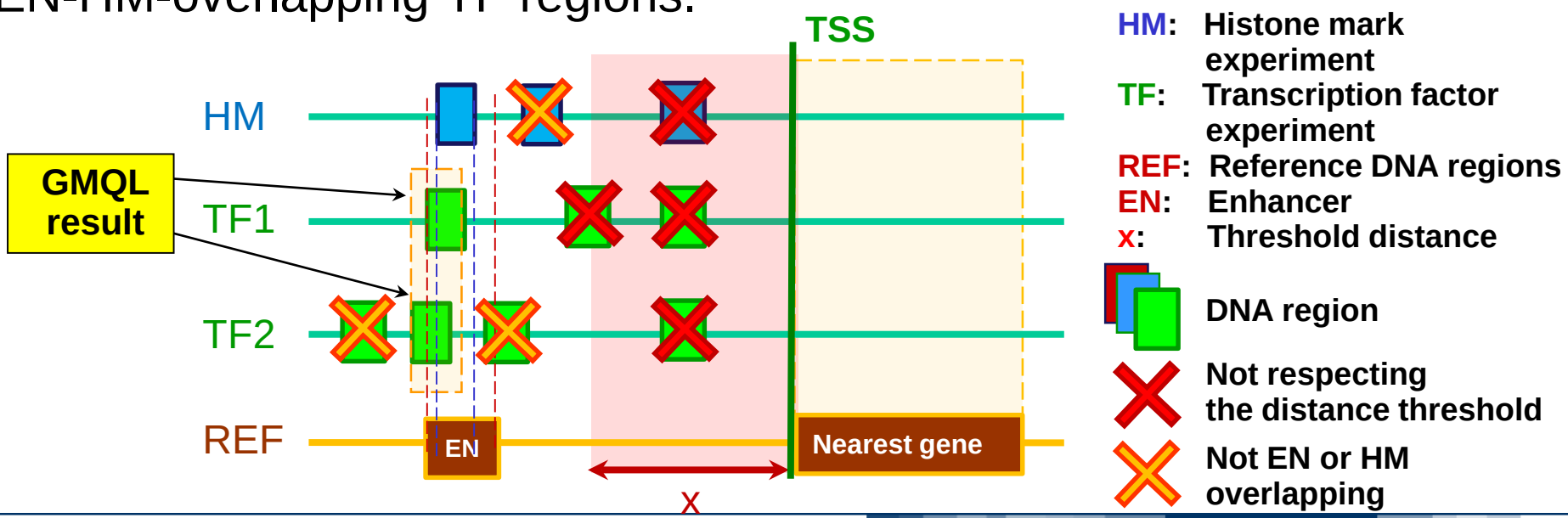
GMQL examples – 1. Distal bindings

Identification of distal bindings in transcription regulatory regions

Find all CTCF transcription factor (TF) binding regions of ChIP-seq data regarding human cancer cell line HeLa-S3, which are farther than x kb (e.g. 1000 kb) from a transcription start site (TSS) of the nearest gene.

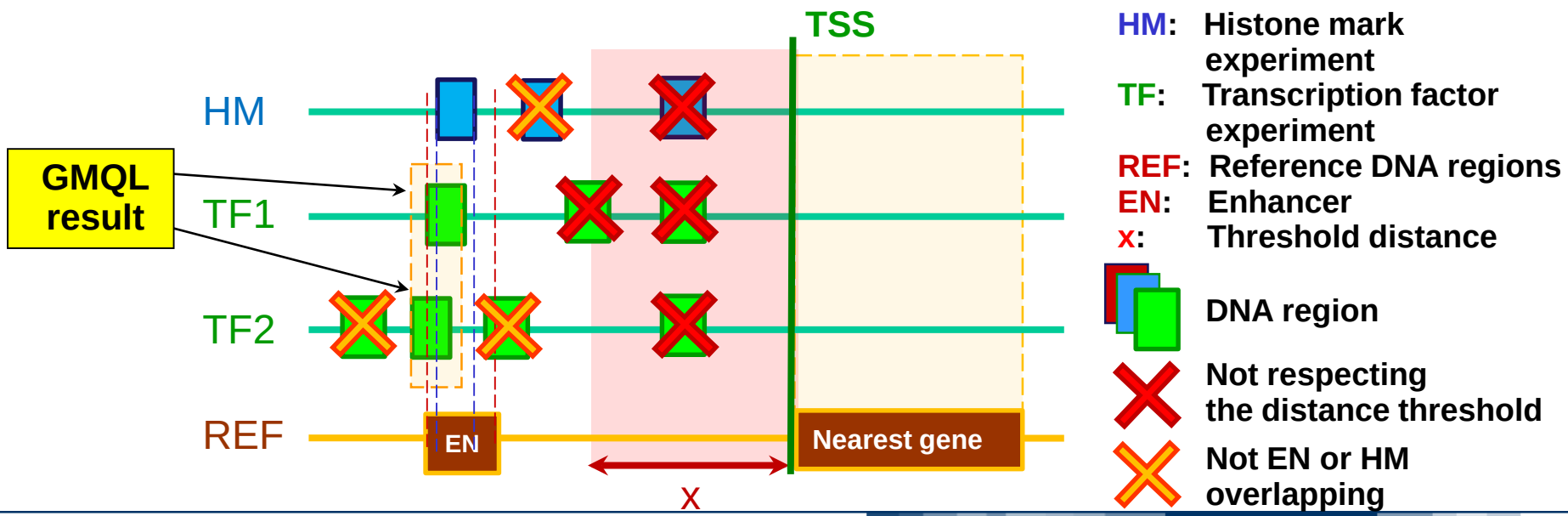
Then, for the same cell line find all H3K4me1 histone modification (HM) peaks that are the nearest regions farther than x kb from a TSS.

Finally, consider known enhancer (EN) regions and return a list of EN-HM-overlapping TF regions.



Genomic Computing applications
GMQL examples – 1. Distal bindings

```
HM = SELECT(dataType == 'ChipSeq‘ AND cell == 'HeLa-S3‘
                                        AND antibody == ‘H3K4me1') PEAK;
TF = SELECT(dataType == 'ChipSeq‘ AND cell == 'HeLa-S3‘
                                        AND antibody == ‘CTCF') PEAK;
TSS = SELECT(type == ‘TSS‘) ANNOTATION;
EN = SELECT(type == ‘enhancer‘) ANNOTATION;
TFa = JOIN(distance > 1000000, minDistance(5); output: right) TSS TF;
HMa = JOIN(distance > 1000000, minDistance(5); output: right) TSS HM;
HMb = JOIN(distance < 0; output: int) EN HMa;
HMc = MERGE() HMb;
TF_res = JOIN(distance < 0; output: right) HMc TFa;
```



POLITECNICO DI MILANO

Finding CpG islands with genetic mutations in multiple cell lines

Consider DNA-seq data of distinct human cancer cell lines; for each of them quantify the mutations in each CpG island.

Then, select the CpG islands with at least a mutation.

Return the list of cancer cell lines ordered by the number of such CpG islands.

```
MUT = SELECT(dataType == ‘DnaSeq‘ AND cell_karyotype == ‘cancer‘) EXP;
CpG = SELECT(type == ‘CpG islands') ANNOTATION;
CpG1 = MAP() CpG MUT;
CpG2 = SELECT(region: count_CpG_MUT > 0) CpG1;
CpG3 = EXTEND(CpG_Count AS COUNT()) CpG2;
CpG_res = ORDER(CpG_Count DESC) CpG3;
MATERIALIZE CpG_res INTO CpG_res;
```

GMQL examples – 3. Top ChIP-seq samples

Finding top ChIP-seq samples with highest number of enriched regions in promoters

In each ENCODE narrow peak ChIP-seq sample of the K562 chronic myelogenous leukemia cell line, select ChIP-seq enriched regions that intersect at least a gene promoter and extract the top 3 samples with the highest number of such enriched regions.

```
HM_TF = SELECT(assay == ‘ChIP-seq‘ AND output_type == ‘peaks‘ AND
    biosample_term_name == ‘K562‘) HG19_ENCODE_NARROW_NOV_2017;

TRANSC = SELECT(annotation_type == ‘transcript‘ AND
    release_version == ‘19‘) HG19_ANNOTATION_GENCODE;
PROM = PROJECT(region_update: start AS start - 2000,
    stop AS start + 1000) TRANSC;

HM_TF_PROM = JOIN(DISTANCE < 0; output: left_distinct) HM_TF PROM;
HM_TF_PROM1 = EXTEND(region_count AS COUNT()) HM_TF_PROM;
RES = ORDER(region_count DESC; meta_top: 3) HM_TF_PROM1;
MATERIALIZE RES INTO RES;
```

POLITECNICO DI MILANO

Finding promoters with highest number of ChIP-seq enriched regions

After combining high quality narrow peak ChIP-seq sample replicates for each experiment target of the K562 chronic myelogenous leukemia cell line available in ENCODE and calculating the average enrichment (signal) for each obtained enriched region of each experiment target, extract the gene promoters with more than 50 experiment target enriched regions.

```
HM_TF = SELECT(assay == ‘ChIP-seq‘ AND output_type == ‘peaks‘ AND
    biosample_term_name == ‘K562‘) HG19_ENCODE_NARROW_NOV_2017;
TRANSC = SELECT(annotation_type == ‘transcript‘ AND
    release_version == ‘19‘) HG19_ANNOTATION_GENCODE;
PROM = PROJECT(region_update: start AS start - 2000,
    stop AS start + 1000) TRANSC;
HM_TF1 = COVER(1, ANY; groupby: experiment_target;
    aggregate: AVG_signal AS AVG(signal)) HM_TF;
HM_TF2 = MERGE() HM_TF1;
PROM_HM_TF = MAP(count_name: region_count) PROM HM_TF2;
RES = SELECT(region: region_count > 50) PROM_HM_TF;
MATERIALIZE RES INTO RES;
```

Combining ChIP-seq and DNase-seq data in different formats and sources

Extract broad peaks of ChIP-seq transcription factor binding sites and histone modifications from ENCODE samples that intersect DNase-seq open chromatin regions from Roadmap Epigenomics in normal H1 embryonic stem cells.

```
CHIPSEQ = SELECT(dataType == 'ChipSeq' AND view == 'Peaks'
     AND setType == 'exp' AND cell == 'H1-hESC') HG19_ENCODE_BROAD;
DNASESEQ = SELECT(assay == 'DNase.hotspot.broad'
     AND Standardized_Epigenome_name == 'H1 Cells')
     HG19_ROADMAP_EPIGENOMICS_BED;
DNASESEQ1 = COVER(1, ANY) DNASESEQ;
CHIPSEQ_IN_DNASESEQ = JOIN(distance < 0; output: right_distinct)
     DNASESEQ1 CHIPSEQ;
MATERIALIZE CHIPSEQ_IN_DNASESEQ INTO CHIPSEQ_IN_DNASESEQ;
```

POLITECNICO DI MILANO

Counting distinct DNA mutations in patient groups

Group patients by tumor type and ethnicity, and count the distinct DNA somatic mutations in each group.

```
MUTATION = SELECT(dataType == ‘dnaseq’) TCGA_dnaseq;
MUTATION_BY_RACE = COVER(1, ANY; GROUPBY tumor_tag, race;
   AGGREGATE: overlap_count AS COUNT(),
   barcodes AS BAG(tumor_sample_barcode)) MUTATION;
MUTATION_COUNT = EXTEND(mutation_count AS COUNT())
   MUTATION_BY_RACE;
MATERIALIZE MUTATION_COUNT INTO MUTATION_COUNT;
```

GMQL examples – 7. Patient datasets (2)

Combining different data types of multiple patients: CNV & miRNA

Match DNA copy number variation (CNV) and microRNA (miRNA) data samples regarding the same biospecimen and extract the CNVs occurring within expressed miRNA genes in the paired samples.

```
CNV = SELECT(dataType == 'cnv') TCGA_cnv;
MIRNA_GENE = SELECT(dataType == 'mirnaseq') TCGA_mirnaseq_mirna;
CNV_GENE_0 = MAP(joinby: bcr_sample_barcode; count_name: gene_count;
      mirna_genes AS BAG(mirna_id)) CNV MIRNA_GENE;
CNV_GENE = SELECT(gene_count > 0) CNV_GENE_0;
MATERIALIZE CNV_GENE INTO CNV_GENE;
```

POLITECNICO DI MILANO

Combining and processing heterogeneous omics data of patients

In TCGA data of breast cancer (BRCA) patients, find the DNA somatic mutations within the first 2000 bp outside of the genes that are both expressed (with normalized_count > 3.0) and have at least a methylation in the same patient biospecimen, and extract these mutations of the top 5% patients with the highest number of mutations.

```
EXPRESSED_GENE = SELECT(dataType == ‘rnaseqv2’ AND tumor_tag == 'brca‘;
            region: normalized_count > 3.0) HG19_TCGA_RnaSeqV2_Gene;
METHYLATION = SELECT(dataType == ‘dnamethylation’ AND tumor_tag == 'brca')
            HG19_TCGA_Dnamethylation;
MUTATION = SELECT(data_type == ‘dnaseq’ AND tumor_tag == 'brca')
            HG19_TCGA_DnaSeq;
GENE_METHYL = JOIN(distance < 0; output: left_distinct;
            joinby: bcr_sample_barcode) EXPRESSED_GENE METHYLATION;
MUTATION_GENE = JOIN(distance <= 2000, distance >= 0; output: left_distinct;
            joinby: bcr_sample_barcode) MUTATION GENE_METHYL1;
MUTATION_GENE_count = EXTEND(mut_count AS COUNT()) MUTATION_GENE;
MUTATION_GENE_top = ORDER(mut_count DESC; meta_topp 5) MUTATION_GENE_count;
MATERIALIZE MUTATION_GENE_top INTO MUTATION_GENE_top;
```

Calling cell line-specific active enhancers

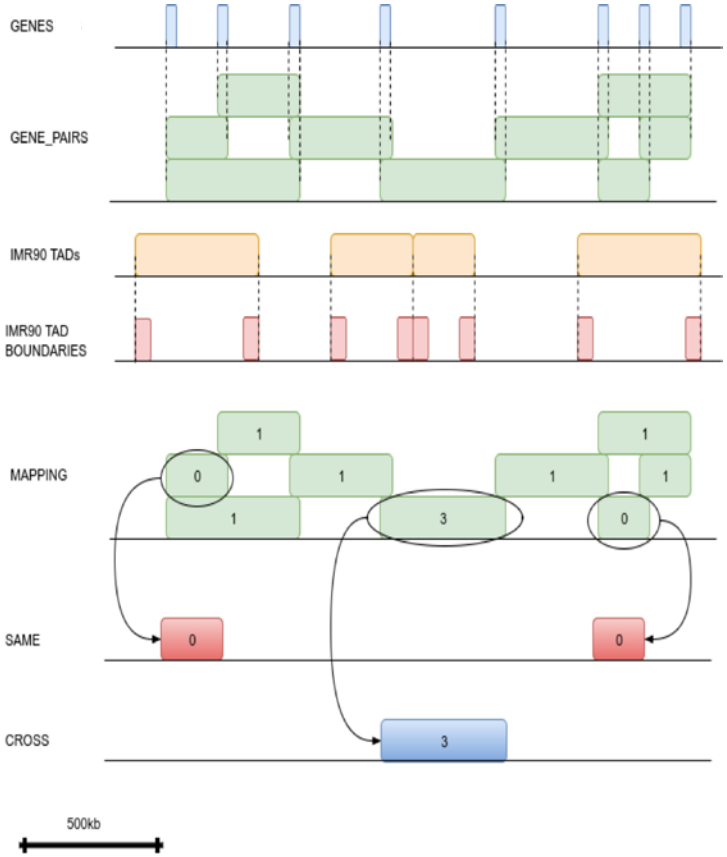
From the entire ENCODE ChIP-seq narrow peak dataset, combine all available H3K4me1 and H3k27ac histone modifications to identify, for each cell line in ENCODE, putative active enhancers, i.e., regions of the genome where both a peak of H3K4me1 and a peak of H3k27ac exist. Next, combine the results for the various cell lines to identify those enhancers that are specifically present in only one of them.

```
me1 = SELECT(assay == "ChIP-seq" AND output_type == "peaks" AND
    experiment_target == "H3K4me1-human") HG19_ENCODE_NARROW_NOV_2017;
me1_c = COVER(1,ANY; groupby: biosample_term_name) me1;
ac = SELECT(assay == "ChIP-seq" AND output_type == "peaks" AND
    experiment_target == "H3K27ac-human") HG19_ENCODE_NARROW_NOV_2017;
ac_c = COVER(1,ANY; groupby: biosample_term_name) ac;
active = JOIN(DISTANCE < 0; output: INT; joinby: biosample_term_name)
    me1_c ac_c;
labeled = PROJECT(region_update: cell AS META(ac_c.biosample_term_name,
    STRING)) active;
cell_specific = COVER(1,1; aggregate: cell_line AS BAG(cell)) labeled;
MATERIALIZE cell_specific INTO cell_specific;
```

Mapping pairs of genes on topologically associating domains

Giving a dataset of topologically associating domains specific of a cell line, find pairs of genes which are inside the same topological domain or across different domains.

```
GENE = SELECT(annotation_type == 'gene')
   HG19_BED_ANNOTATION;
GENE_PAIRS = JOIN(DLE(180000); output: CAT)
   GENE GENE;
TADs = SELECT(cell == 'imr90') TADs_Dixon;
BOUNDARIES_1 = PROJECT(region_update: stop AS
   start + 1) TADs;
BOUNDARIES_2 = PROJECT(region_update: start AS
   stop - 1) TADs;
BOUNDARIES = UNION() BOUNDARIES_1 BOUNDARIES_2;
BOUNDARY = MERGE() BOUNDARIES;
MAPPING = MAP() GENE_PAIRS BOUNDARY;
SAME = SELECT(region: count_GENE_PAIRS_BOUNDARY
   == 0) MAPPING;
MATERIALIZE SAME INTO same;
CROSS = SELECT(region:
   count_GENE_PAIRS_BOUNDARY > 1) MAPPING;
MATERIALIZE CROSS INTO cross;
```



GenoMetric Query Language System
http://www.bioinformatics.deib.polimi.it/GMQLsystem/

Documentation

- GMQL introduction to the language
- Videos: Step-by-step use of GMQL Web interface or Python library
- GMQL example queries
- Biological examples

Accesses

- Web interface & User manual

Download

- PyGMQL: Python library & documentation
- RGMQL: R/Bioconductor package & documentation

Genomic Computing applications
Biological application examples

Gene regulatory networks inference (M. Negri, Maddalena Fratelli)

- GDM integrated ENCODE TF and TCGA gene expression data
- GMQL to extract genes encoding TFs binding a gene promoter
- Regression models to identify candidate regulatory genes & networks

Data-driven modeling of epigenetic transcriptional regulation (IIT)

- GMQL extraction and preprocessing of Roadmap Epigenomics and ENCODE ChIP-seq and RNA-seq data
- Regression and classification models to identify relevant TFs & HMs involved in genome-wide and specific gene expression regulation

Gene expression based comparative evaluation of breast cancer prognostic classifiers (Candiolo Cancer Institute)

- TCGA RNA-seq data and in house produced data
- Machine learning for identification and comparison of classifiers

http://www.bioinformatics.deib.polimi.it/genomic_computing/

http://www.bioinformatics.deib.polimi.it/GMQLsystem/

Thank you for your attention!

Any question?