Information Integration

Assignment 2: Working with the GO

Ulf Leser
Overview

• Reuse all data you have imported in Assignment 1

• In this task, we will
  – Clean it up a bit (more to come)
  – Integrate the “real” GO
  – Answer queries that need to traverse a taxonomy
Task 1: Data Cleansing

• Duplicate elimination
  – Gene2refseq has duplicates (multiple entries with the same taxa|geneid). These may have different status, protein accession, start/ end positions, etc.
  – Check for each attribute whether it functionally depends on the tax|geneid columns
  – All values which have 1:n relationship to a gene should be moved to a separate table
    • One table for the “payload” suffices
• Define a primary key on TAXID,GENEID on your gene table
Task 2: Integrate the Gene Ontology

• The Gene Ontology is a collection of 3 ontologies
  - Each forms a DAG
  - Semantics: If a gene G is assigned a term X and X IS_A Y (directly or indirectly), then Y is implicitly assigned to G

• Model such an ontology in your schema
  - We want: term_ID, term_name, ontology, ISA-relationships

• Import the Gene Ontology
  - Link to GO-OBO is on the web page
  - Parse the file and find all terms
  - We only need ID, NAME, NAMESPACE and IS_A
  - IS_A contains links to parents
  - Disregard all terms with “is_obsolete: true”
  - “Connect” the data to the GENE_GO data from ass1 via ID
Task 3: Queries

- Formulate and execute the following queries
  - Task 1
    1. Number of distinct genes (9606) after deduplication
    2. Maximal number of different map_locations assigned to a gene
    3. How many genes have only one map_location?
  - Task 2
    4. How many GO terms are there in total and for each ontology?
    5. Compute a frequency histogram over the number of terms assigned to a gene (how many have 0 terms, 1 term, 2 terms, ...)
    6. Write a program (or query) which computes for each term to how many genes it is assigned
       - Store the result in a table ASSIGNMENTS( TERM_ID, COUNT)
       - This is not trivial - you need to traverse a large DAG
       - Use JAVA program, PL/SQL, recursive queries, ...
       - Traverse the tree or use some clever materialization
Competition

• Compute the last query as fast as possible
  – I.e., fill the ASSIGNMENT table as fast as possible

• To take part, you need to send me your program
  – Host language
    • Must be executable “as is” on gruenau2
    • You may use whatever programming language is available there
      – E.g. JAVA 1.4.2
  – Oracle
    • You can send me a query
    • You can send me the name of a PL/SQL procedure in your schema

• I shall measure the SYSTEM time at Unix or the procedure execution time in Oracle
Deliverables

- By Monday 17.11., or Wednesday 19.11, 23:59 o’clock
  - Two weeks
- Send by mail as ASCII
  - An updated schema graph with the new table(s) for GO
  - For queries 1-5: Query + result in a text file
  - For query 6: Report the value for “biological_process” and leave the full table ASSIGNMENT in your schema
  - For the competition: Executable code as specified on last slide