Course assignment: Optimal Database Design problem

R. Tadei, D. Manerba

Dept. of Control and Computer Engineering

Politecnico of Torino, Italy

01OUV - Optimization Methods and Algorithms - 2018/2019

Design of Physical Databases

- translates the logical data model into *technical specifications* able to physically create a database (DB) on a machine;
- **definition of an appropriate set of access structures**, offering a good compromise between memory occupation and time required for data retrieval and maintenance;
- **depends upon the database management system** (DBMS) used. The designer has to take into account, among others, two basic aspects:
  - the access structures supported by the DBMS (e.g., hash functions, links, inverted indexes, clustering of data, etc.);
  - the strategies used in accessing the data (e.g., join algorithms, index intersection methods, etc).
- **complex task** due to the large number of possible choices (even if the DBMS offers only a limited set of features);
- a particularly important phase of the design consists of **choosing the indexes to be created in order to globally minimize the response time** for a given workload (queries on the DB).

Optimal Database Design problem (ODBDP)

Let Q be the set of queries composing the workload of a DB, and I be the set of possible indexes that can be built to serve the queries. Since not only single indexes, but also sets of indexes can be used for a query, let C be the set of possible indexes configurations. The set of indexes composing a certain configuration is the subset $I_c \subset I$ and a configuration c is called *active* if all its indexes $i \in I_c$ are built.

Each index $i \in I$ can be built at a fixed cost $f_i > 0$, corresponding to the time needed for its creation and maintenance, and using $m_i > 0$ units of memory. Moreover, let $g_{cq} \geq 0$ be the gain in terms of execution time of using index configuration $c \in C$ for query $q \in Q$. A null gain $g_{cq} = 0$ either means that configuration c cannot be used for query q or simply that it does not affect its execution time.

The Optimal Database Design problem (ODBDP) aims at selecting the indexes to be built for the DB (consistently with the activated configurations) ensuring that:

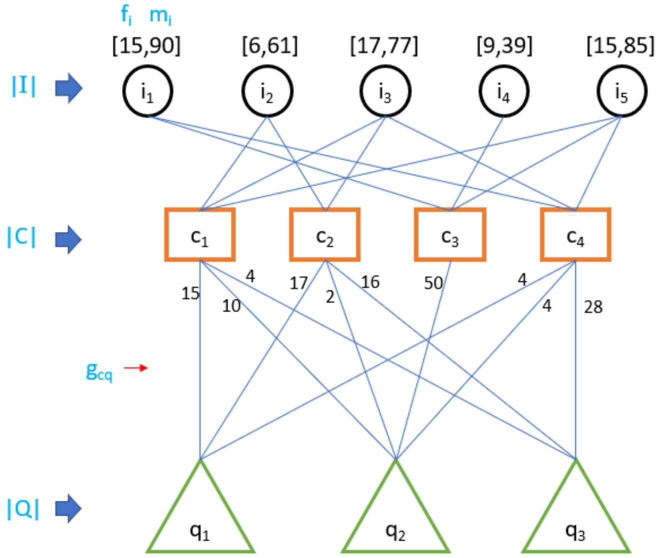
- at most one index configuration is used for each query;
- the total memory usage do not exceed a predefined capacity M.

The objective is to maximize the net gain in terms of total time (i.e., also considering creation and maintenance times) to execute all the queries on the DB.

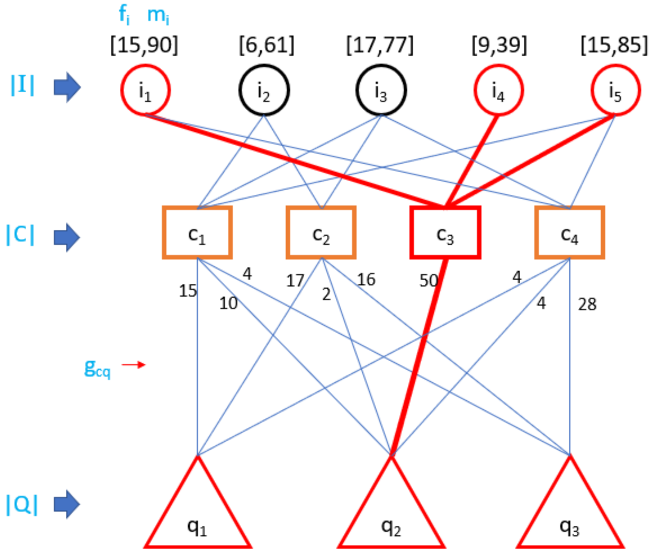
Assumptions:

- if no one of the configurations in C is used to serve a query, we will assume that the default configuration is used

Clarifying example: a *test* instance ($M = 300$)



Clarifying example: a *feasible* solution

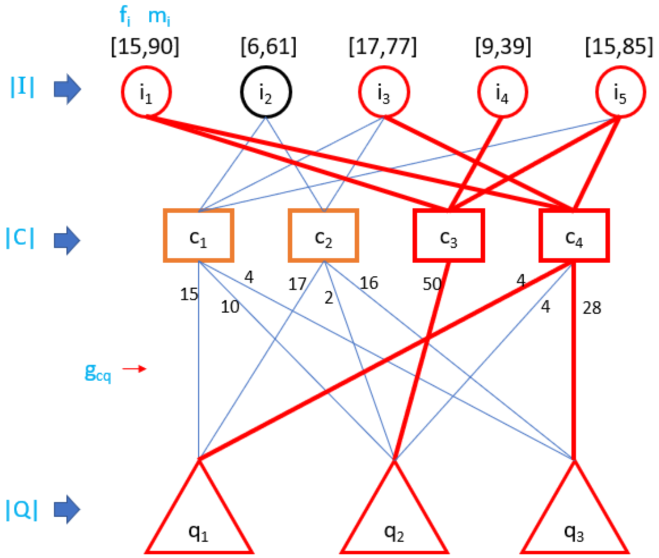


FEASIBLE!

Gain: 50
Cost: 39
Net gain: 11

Memory: 214

Clarifying example: the *optimal* solution

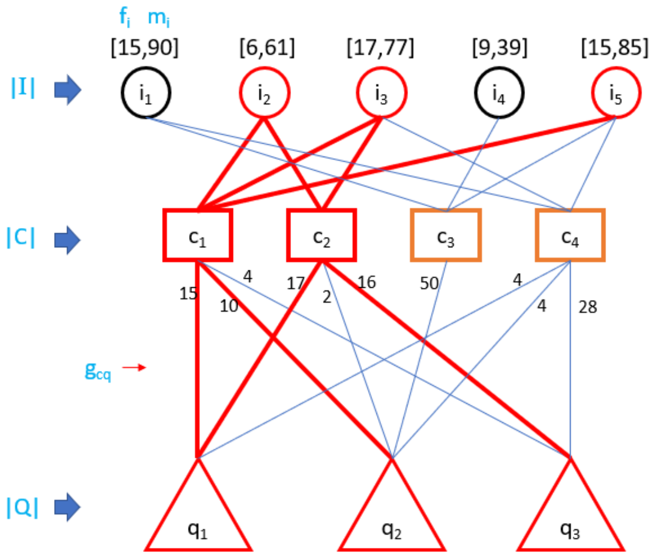


FEASIBLE!
and OPTIMAL!

Gain: 82
Cost: 56
Net gain: 26

Memory: 291

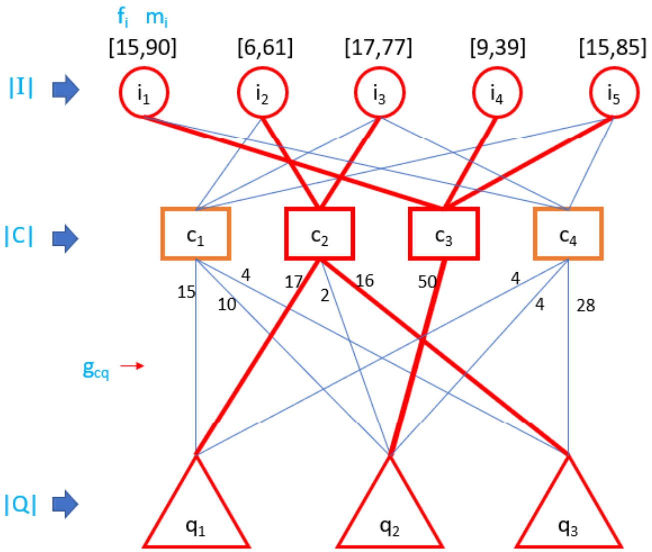
Clarifying example: an *infeasible* solution



INFEASIBLE!

Query q_1 is served by many configurations!

Clarifying example: another *infeasible* solution

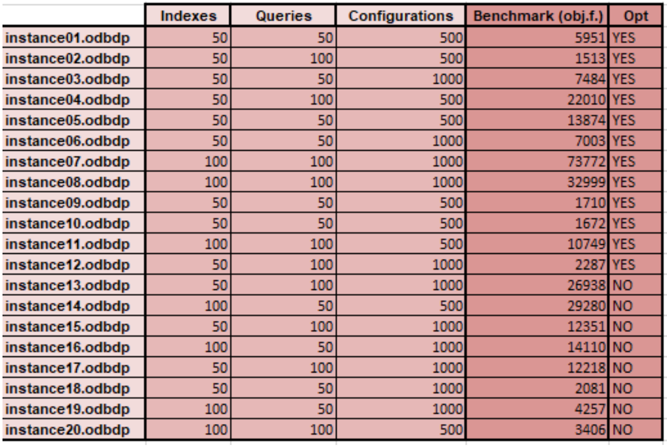


INFEASIBLE!

The needed memory to build the indexes is 352, while the available memory is 300!

Benchmark instances: properties

$\longrightarrow$ **20 public instances**: to test and assess the results of your algorithms with respect to the benchmarks

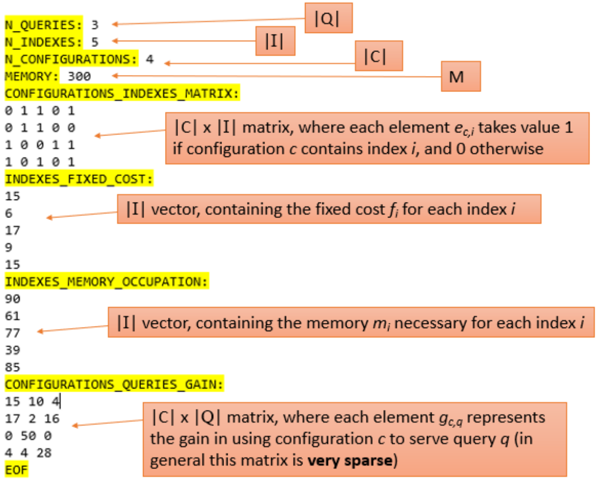


	Indexes	Queries	Configurations	Benchmark (obj.f.)	Opt
instance01.odbdp	50	50	500	5951	YES
instance02.odbdp	50	100	500	1513	YES
instance03.odbdp	50	50	1000	7484	YES
instance04.odbdp	50	100	500	22010	YES
instance05.odbdp	50	50	500	13874	YES
instance06.odbdp	50	50	1000	7003	YES
instance07.odbdp	100	100	1000	73772	YES
instance08.odbdp	100	100	1000	32999	YES
instance09.odbdp	50	50	500	1710	YES
instance10.odbdp	50	50	500	1672	YES
instance11.odbdp	100	100	500	10749	YES
instance12.odbdp	50	100	1000	2287	YES
instance13.odbdp	50	100	1000	26938	NO
instance14.odbdp	100	50	500	29280	NO
instance15.odbdp	50	100	1000	12351	NO
instance16.odbdp	100	50	1000	14110	NO
instance17.odbdp	50	100	1000	12218	NO
instance18.odbdp	50	50	1000	2081	NO
instance19.odbdp	100	50	1000	4257	NO
instance20.odbdp	100	100	500	3406	NO

$\longrightarrow$ **5 private instances**: to evaluate how the same algorithms are robust when solving instances unknown during the development

Benchmark instances: format

Each instance is defined by a plain text file, with the following format:



```
N_QUERIES: 3
N_INDEXES: 5
N_CONFIGURATIONS: 4
MEMORY: 300
CONFIGURATIONS_INDEXES_MATRIX:
0 1 1 0 1
0 1 1 0 0
1 0 0 1 1
1 0 1 0 1
INDEXES_FIXED_COST:
15
6
17
9
15
INDEXES_MEMORY_OCCUPATION:
90
61
77
39
85
CONFIGURATIONS_QUERIES_GAIN:
15 10 4
17 2 16
0 50 0
4 4 28
EOF
```

|C| x |I| matrix, where each element $e_{c,i}$ takes value 1 if configuration c contains index i, and 0 otherwise

|I| vector, containing the fixed cost f_i for each index i

|I| vector, containing the memory m_i necessary for each index i

|C| x |Q| matrix, where each element $g_{c,q}$ represents the gain in using configuration c to serve query q (in general this matrix is **very sparse**)

P.S.: Keywords have been highlighted in yellow

Assignment: tasks and deadlines organization

Required tasks:

- provide a Linear Programming formulation for the ODBDP (deadline 1)
- propose a solution approach for the ODBDP by exploiting one (or more) heuristic and meta-heuristic algorithms presented during the course
- develop/implement such a solution algorithm through a known programming language (C/C++ or Java)
- solve the benchmark instances through the implemented algorithm
- deliver the project code, the results, and a report of the work (deadline 2)
- present the solution method adopted and the results obtained (to be defined within Jan 8–18, 2019, during the last lessons of the course)

Deadlines:

- deadline 1 (problem formulation): 25/11/2018, 11:59 p.m.
- deadline 2 (project code, results, and report): 07/01/2019, 11:59 p.m.

Assignment: groups and evaluation

Groups:

- 5/7 students per group (one leader must be chosen for corresponding)
- groups composition: up to you! must be decided and submitted before 21/10/2018 by compiling
  - OMA A-L: https://tinyurl.com/OMAAL
  - OMA M-Z: https://tinyurl.com/OMAMZ

Evaluation:

- one single grade [0,10] per group (no individual grades!), based on:
  - correctness/completeness of your formulation
  - soundness/rationality/efficiency of your solution algorithm
  - quality of your solutions (closeness to the benchmark) on public and private instances (3, 2, and 1 additional points to the first, second, and third group, respectively)
  - quality/clarity of the presentation

Support:

- 3 lessons (1h:30m each, approximately two weeks apart) during the course will be dedicated to assistance in assignment development
- the *Forum* section of the course's site: a main thread with FAQs and, obviously, other Q/A that may interest all the groups

Assignment: given materials

- **instances.zip**: set of the 20 public instances (format already explained) plus a test instance (with solutions) corresponding to the example shown before.
- **ODBDPchecker.exe**: a feasibility checker and obj.f. calculator.
  It is a black-box tool able to read an ODBDP instance and a relative specified solution. If the provided solution is feasible, it returns the value of the objective value (total penalty) of that solution, otherwise it returns an error relative to some infeasible characteristic of the solution.

  How to use it from a command line:
  ```
  $ODBDPchecker.exe instancefilename -check solutionfilename
  ```
  PS: only 'logical' feasibility is checked! no guarantees if a non-correctly formatted solution is provided.
- **benchmarks.xlsx**: an Excel file containing the properties and the benchmark values of the public instances. Given the solution values generated by your algorithm, it automatically calculates the percentage gap with respect to the benchmarks.

Assignment: deliveries specifications (1)

Model formulation:

- upload a pdf file (max 2 pp.) named **ODBDPmodel_OMAXX_groupYY**, where 'XX' is 'AL' or 'MZ' (depending on your OMA module) and 'YY' is the number (two digits!) of your group, in *'Elaborati'* section of the course's site
- the file must contain:
  - the definition of the notation used for the parameters and for the variables (notation already introduced here must be maintained)
  - the LP formulation
  - a brief explanation of objective function and constraints.

Assignment: deliveries specifications (2)

Project code, results, and report:

- upload a .zip archive file named **ODBDPproject_OMAXX_groupYY** in *'Elaborati'* section of the course's site
- the archive must contain:
  - 20 text files containing your best solutions of the 20 given public instances (format specifications follow)
  - a .xlsx file named **benchmarks_OMAXX_groupYY**: it is the given *benchmarks.xlsx* file duly completed with your results
  - an executable file (both .exe and .jar are allowed) named **ODBDPsolver_OMAXX_groupYY** (specifications follow)
  - a folder containing the complete development of the software (source code, project files, ...)
  - a .pdf file named **ODBDPreport_OMAXX_groupYY** containing the report of the work. Max. 3 pages, focused on algorithms and results.

Assignment: solver and solutions specifications

The solver:

- a command line application that, inputed with an instance name and a time limit t_{lim} (in seconds), solves the ODBDP problem on the specified instance within the time limit and generates a solution text file
- the solver must respond to the following syntax:
  `$ODBDPsolver_OMAXX_groupYY.exe instancefilename -t timelimit`
- must be working stand-alone, i.e. without the need of installing any additional software
- must write/overwrite a text file any time a new best solution is found! [ask me why...]

A solution:

- a text file named **instancefilename_OMAXX_groupYY.sol** containing a $|C| \times |Q|$ matrix where each element e_{cq} takes value 1 if configuration c serves query q, and 0 otherwise
- must be feasible!

Assignment: deliveries specifications (3)

Final presentation:

- in English, to the whole class, using your laptop linked to the projector
- 15 minutes max. per group (including set-up time and 2-3 minutes for possible questions)
- at least 4 elements of the group must present
- no need to present the problem, the instances, or whatever is common to all groups
- focus on your solution ideas and algorithms, on their tuning, on results and performances
- include some conclusive considerations
- upload the presented slides as a .pdf file named **ODBDPslides_OMAXX_groupYY** in *'Elaborati'* section of the course's site after the presentation

References for more insights on DB physical design

Available in the portal:

- S. Finkelstein, M. Schkolnick, P. Tiberio. Physical Database Design for Relational Databases. ACM Transactions on Database Systems, Vol. 13, No. 1, March 1988, Pages 91-128.
- P. S. Yu, M.-S. Chen, H.-U. Heiss, S. Lee. Workload characterization of relation database environments. IEEE Transactions on Software Engineering, May 1992.
- L. T. Wang. Physical Database Design. National University of Singapore. Course slides available at: https://www.comp.nus.edu.sg/∼lingtw/cs4221/physical.db.design.pdf