USING CASE-BASED REASONING FOR DATABASE QUERY OPTIMIZATION IN UBIQUITOUS ENVIRONMENTS

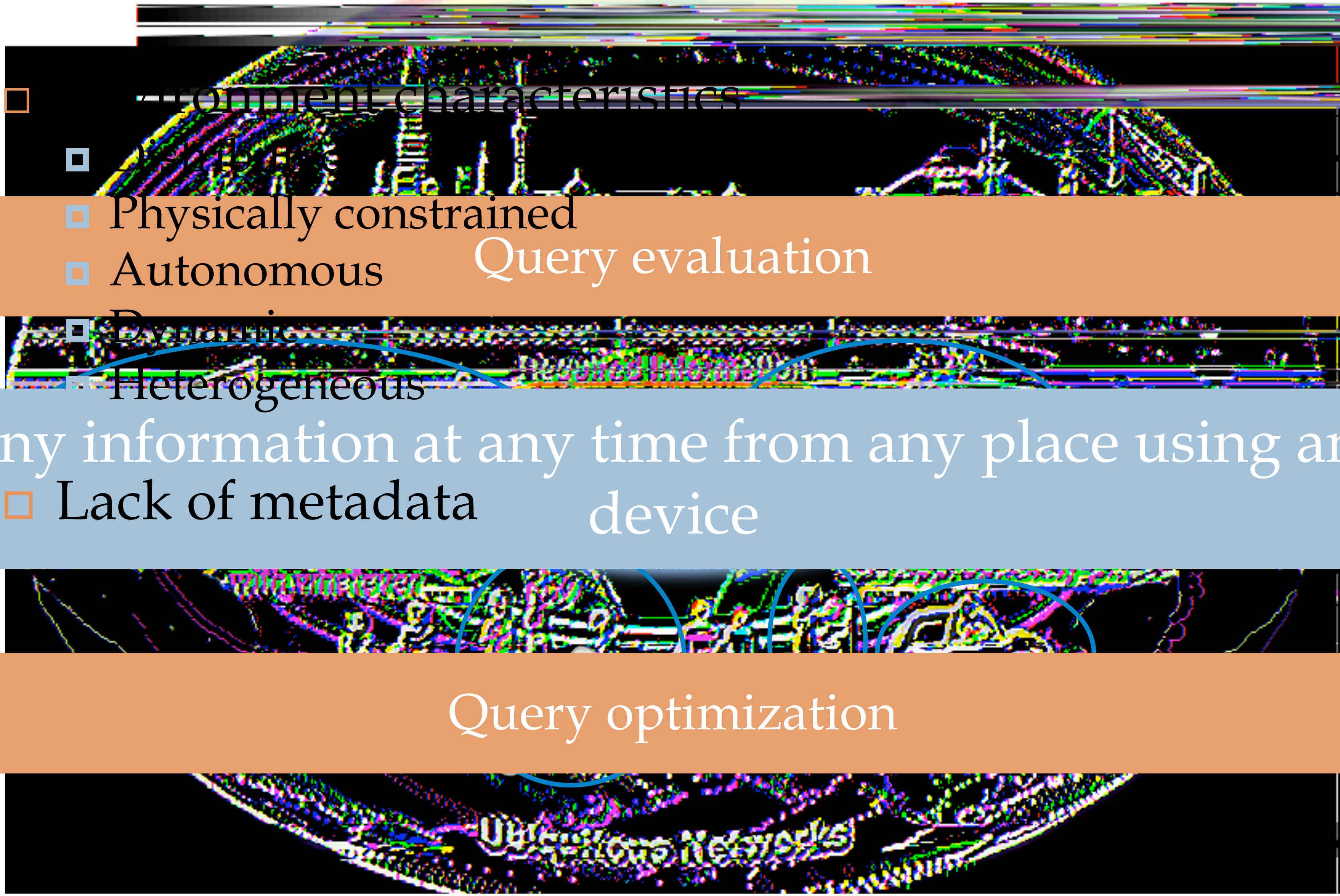
Lourdes MARTÍNEZ-MEDINA and Christophe BOBINEAU

Firstname.Lastname@imag.fr

UDLA-P / CENTIA, LAFMIA (UMI 3175)

Laboratorie d'Informatique de Grenoble LIG – HADAS

Ubiquitous computing environment



- ronment characteristics
  - Physically constrained
  - Autonomous
  - Dynamic
  - Heterogeneous
- Lack of metadata

Query evaluation

Any information at any time from any place using any device

Query optimization

Content

- ✓ Ubiquitous computing environments
- Classical query optimization techniques and ubiquitous environments
- Query optimization using case-based reasoning
- Conclusions and future works

Classical query evaluation

Declarative query Q

Query parser

Algebraic query tr

Query optimizer

Execution plan

Code generator

Execution plan coc

Query executor

Data

Declarative query Q

Restaurants

Code

Query parser

Quer

Cod

Query executor

Data

esse	Ville

```
/*First join execution*/
        if(Rnew.a = R2.c){
            combine(Rnew, R2);
            for (int r1=0; r1<R1.data.size; r1++)
                if  (touple1 = R1.get(r1) = "Italianne")
                    recip1.add (touple1);
        }

 /*Second join execution*/
        if(Rnew.a = R2.c){
            combine(Rnew, R2);
            Rnew = combine(recip1, R2);
            for (int r3=0; r3< R1.data.size; r3++)
                touple3 = toumple3.a, toumple3.b;
                    if  (touple3 = R3.get(r3) => 50)
                        recip3.add (touple3);
        }
```

rant.ville = Ville.nom and

Ville.numDept = Region.numDept

R_1

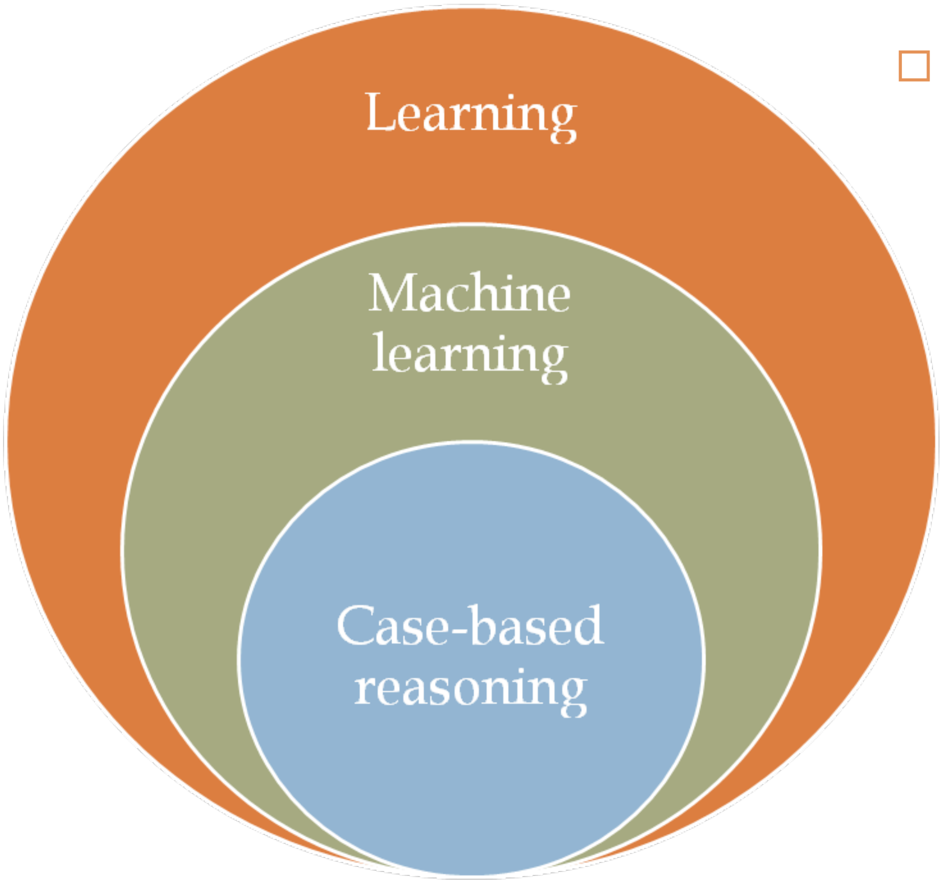
Optimization process

- Search space exploration
  - Using algebraic transformation
  - Using heuristics
  - Random exploration
- Execution cost estimation
  - For each generated plan
  - Using cost function or rules
- Extensive use of metadata and statistics that are not always available

Content

- ✓ Ubiquitous computing environments
- ✓ Classical query optimization techniques and ubiquitous environments
- Query optimization using case-based reasoning
  - Data model
  - Pseudo-random query plan generation
  - Case-based reasoning adaptation to query optimization
- Conclusions and future work

Query optimization using case-based reasoning



- The cost of execution plans cannot be evaluated *a priori*, so **lets try and see**!
  - (Pseudo-)random search space exploration where just ONE execution plan is generated
  - Measures during query plan execution
  - Reuse learned plans
  - Continuous learning

Advantages

☺No need of statistics

- Optimization where classical techniques are not applicable
- Progressive optimization
  - Can be accelerated by preloading/sharing of case bases

☺Personalization of optimization objectives

- Cost function depending on measures

New Problem

R1 : Restaurant(<u>nom</u>, adresse, ville)
R2 : Ville (<u>nomVille</u>, NumDept)
R3 : Region(NumDept, <u>NomRegion</u>)

- Q

  Select Restaurant.nom, Restaurant.adresse, Restaurant.ville
  From Restaurant, Ville, Region
  Where Region.nom = ‘Rhone Alpes’ and Restaurant.specialite = ‘Italianne’ and Restaurant.ville = Ville.nom and Ville.numDept = Region.numDept

+getQuery(): Query
+getContext():Context
+getContext: List<Parameter>

Problem

Q

Select = {R1.a1, R1. a2, R1. a3}
From = {R1, R2, R3}
Where = {Sel(R3.nom=‘Rhone Alpes’), Sel(R1.specialite=‘Italianne’), Join(R1.ville = R2.nom), Join(R2.numDept = R3.numDept)}

-fromClause: From
-value: any
+getName() String

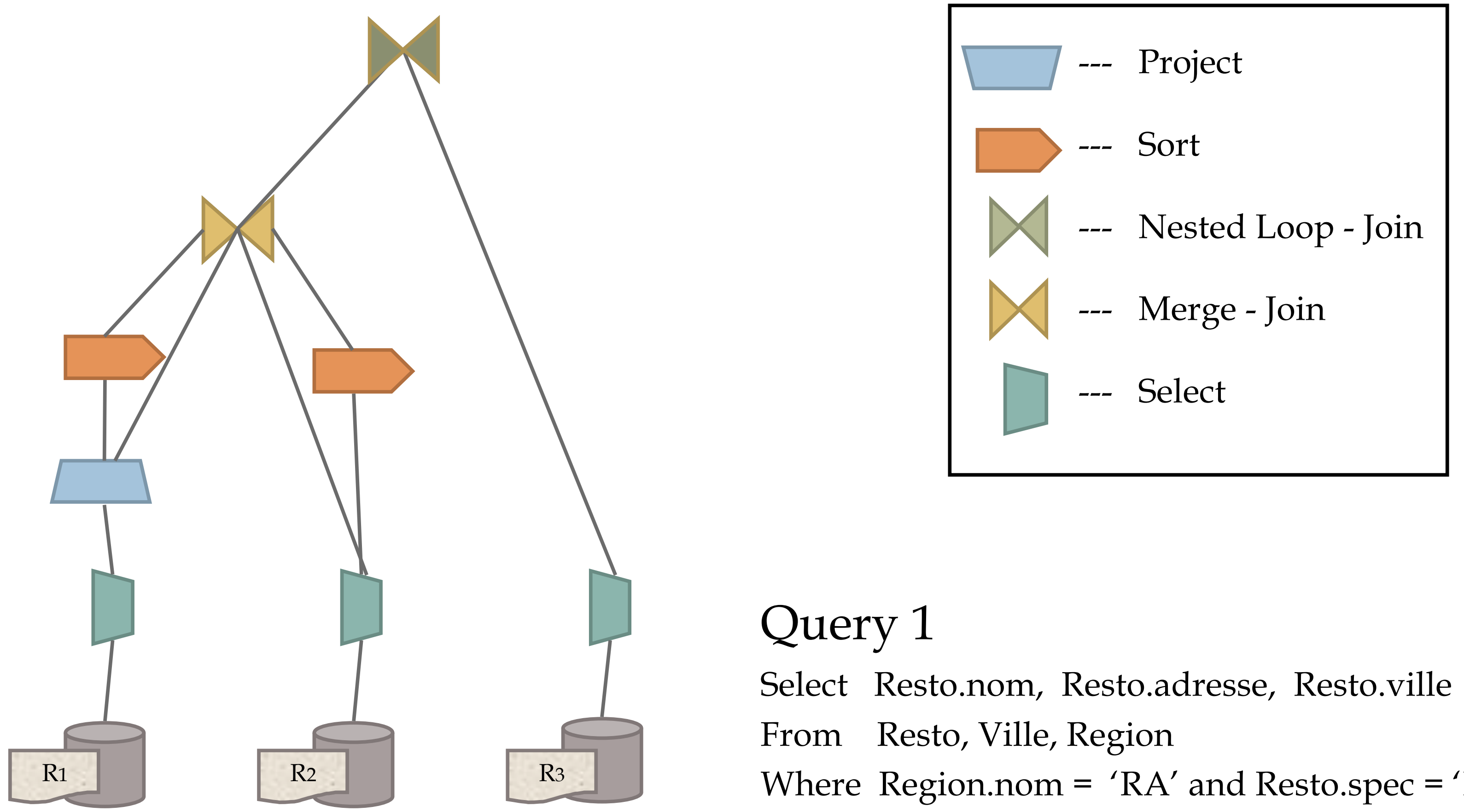
Context

Free memory = 400 KB
CPU charge = 75%
Available Energy = 70%

Optimization objective

F(memory)

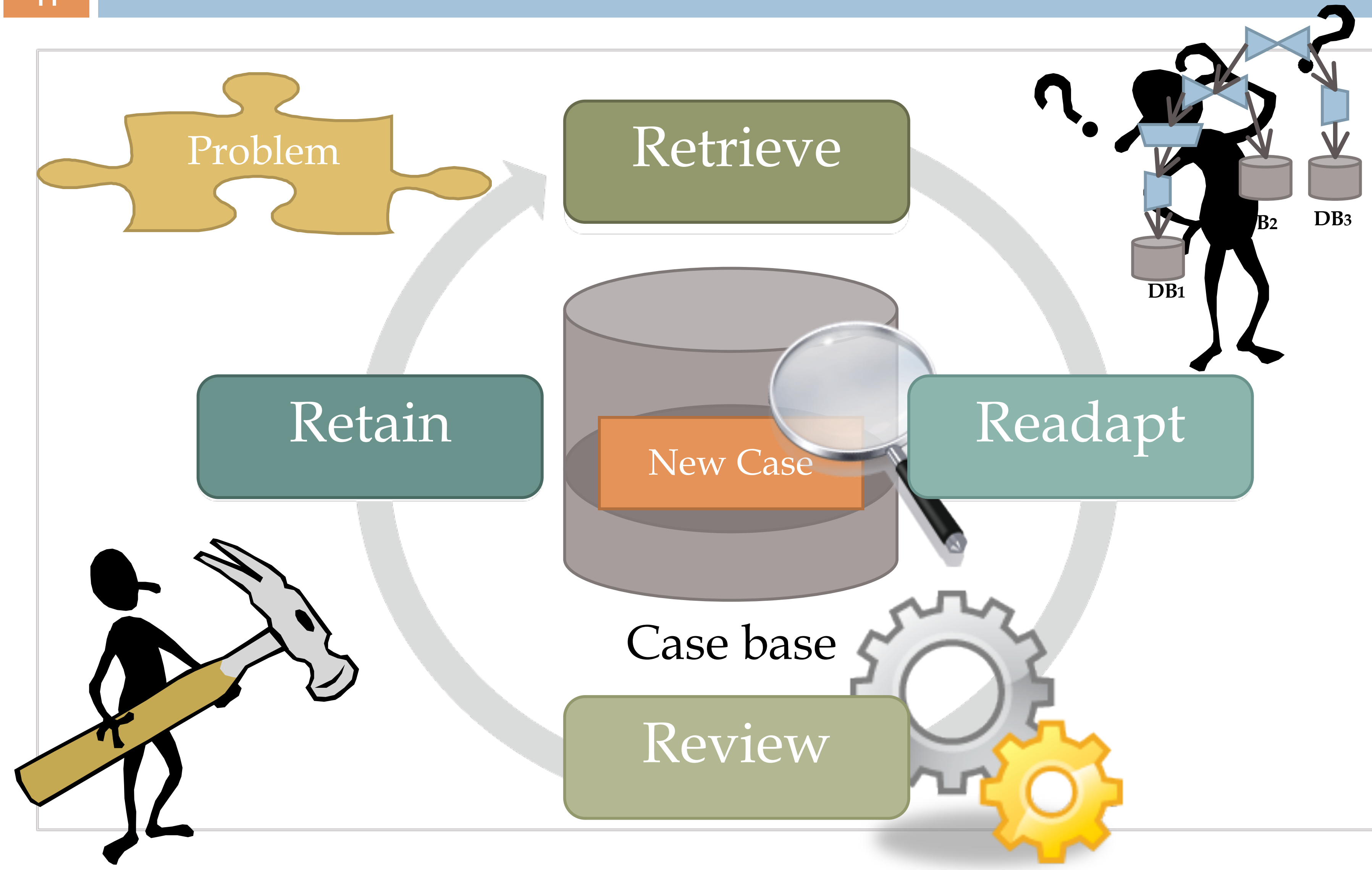
Pseudo-random query plan generation



Query 1

Select Resto.nom, Resto.adresse, Resto.ville
From Resto, Ville, Region
Where Region.nom = ‘RA’ and Resto.spec = ‘It’ and
Resto.ville = Ville.nom and
Ville.numDept = Region.numDept

Case-based reasoning process adaptation to query optimization



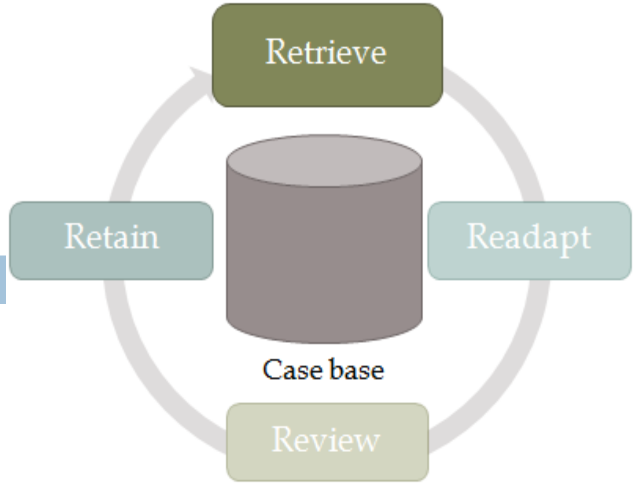
Query classification and similarity

- Similarity guided by adaptation possibilities

- Query classification
  - Queries that can be solved by the same set of query plan shapes
  - Common features
    - Same data sources
    - Same join conditions
    - Same selection operators (but not values)

- Query similarity
  - Queries in the same query class are similar
  - Similarity is the base for retrieval process
    - Identify query class
    - Explore equivalent cases
  - Similarity guide storage of cases

Retrieve similar queries



- Q1

Select	= {R1.nom, R1. addres, R1. specialite}
From	= {R1, R2, R3}
Where	= {Sel(R3.nom='Rhone Alpes'), Sel(R1.specialite='Italianne'), Join(R1.ville = R2.nom), Join(R2.numDept = R3.numDept)}

- Q2

Select	= {R1.nom, R1. addres}
From	= {R1, R2, R3}
Where	= {Sel(R3.nom='Alsace'), Sel(R1.specialite='Indienne'), Join(R1.ville = R2.nom), Join(R2.numDept = R3.numDept)}

Case

R1 : Restaurant(nom, adresse, ville)
R2 : Ville (nomVille, NumDept)
R3 : Region(NumDept, NomRegion)

- Q1

  Select Restaurant.nom, Restaurant.adresse
  From Restaurant, Ville, Region
  Where Region.nom = 'Alsace' and Restaurant.specialite = 'Indienne' and Restaurant.ville = Ville.nom and Ville.numDept = Region.numDept

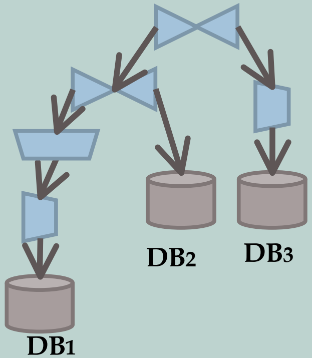
Case
-query: Query
-queryPlan: Plan
-evaluation: Evaluation
+getQuery(): Query

Case 3

Q

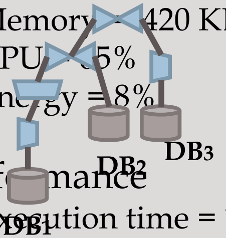
Select = {R1.a1, R1. a2, R1. a3}
From = {R1, R2, R3}
Where = {Sel(R3.nom='Rhone Alpes'), Sel(R1.specialite='Italianne'), Join(R1.ville = R2.nom), Join(R2.numDept = R3.numDept)}

Query plan



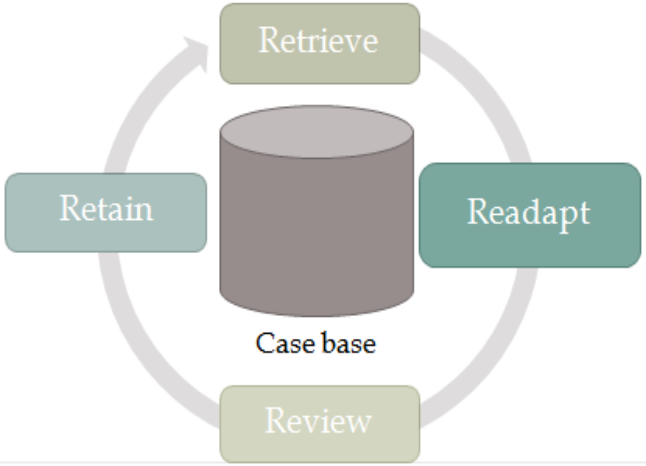
Evaluation

Consumed resources
Memory = 420 KB
CPU = 5%
Energy = 8%



Performance
Execution time = 150 ms

Adaptation process



- Adaptation concerns
  - Projection attributes
  - Selection condition (fix values for selection)

- The adaptation process depends on the similarity level between the queries

Similarity level	Equivalent clauses	Different clauses
4	selectClause, fromClause and whereClause	---
3	fromClause and whereClause	selectClause
2	selectClause and fromClause	whereClause
1	fromClause	selectClause and whereClause
0	---	fromClause, selectClause and whereClause

Content

- ✓ Ubiquitous computing environments
- ✓ Classical query optimization techniques and ubiquitous environments
- ✓ Query optimization using case-based reasoning
- □ Conclusions and future work

Conclusions

- We proposed a query optimization technique that
  - Allows the query optimization when no metadata is available
  - Exploits case-based reasoning
  - Allows the personalization of optimization objectives

- The contribution of our work is
  - Case-based reasoning adaptation to query optimization
  - Data model for the knowledge representation
  - First prototype implementation

First results

- Evaluation time

Future work

- Extensive experimental evaluation
- Improvement of knowledge acquisition and exploitation
  - Sub-queries
  - Knowledge preloading/sharing
- Dynamicity management
  - React to changes in the environment
  - Knowledge maintenance
- Other application domains

Thank you

Bibliography

[1] Features of similarity and category-based onduction, 1997

[2] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications, 7(1):39{59, 1994

[3] E. Bardram. Activity-based computing: support for mobility and collaboration in ubiquitous computing. Personal Ubiquitous Comput., 9(5):312{322, 2005

[4] G. A. A. Deyand, P. Brown, and et al. Towards a better understanding of context and context-awareness. In: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing (HUC 1999), September 1999

[5] M. Franklin. Challenges in ubiquitous data management. In R. Wilhelm, editor, Informatics - 10 Years Back. 10 Years Ahead., pages 24{33. Springer-Verlag, 2001

[6] Y. Ioannidis. Query optimization. ACM Comput. Surv., 28(1):121{123, 1996

[7] L. D. Mantaras, R. McSherry, and et al. Retrieval, reuse, revision and retention in case-based reasoning. Knowl. Eng. Rev., 20(3):215{240, 2005

Bibliography

[8] Microsoft. Execution plan caching and reuse. http://technet.microsoft.com/enus/library/bb545450.aspx, 2008

[9] J. Nilsson. Introduction to Machine Learning (An Early Draft to a proposed textbook). 1996

[10] M. Othman-Abdallah. Optimisation de requetes par apprentissage. PhD thesis, Institut National Polytechnique de Grenoble

[11} F. Perich, S. Avancha, and et al. Prole driven data management for pervasive environments. In In: Proceedings of the 13th International Conference on Database and Expert Systems Applications (DEXA 2002), pages 361{370, London, UK, 2002. Springer-Verlag

[12] M. Stillger, G. Lohman, V. Markl, and M. Kandil. Leo - db2's learning optimizer. In: Proceedings of the 27th International Conference on Very Large Data Bases (VLDB 2001), pages 19{28, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc

[13] D. Subramanian and K. Subramanian. Query optimization in multi-database systems. Distrib. Parallel Databases, 6(2):183{210, 1998

[14] A. Tversky and I. Gati. Studies of similarity. 1978