Data Mining and Clustering

Antonio Mucherino

Laboratoire d'Informatique, École Polytechnique

Course on Operations Research (ISC610A)

Semester I - 2008/09

November 20^{th} 2008

Outline

Outline

Introduction to Data Mining

Example I - blood analysis

- A laboratory is performing blood analysis on sick and healthy patients.
- The goal is to correlate patients' illness to blood measurements.
- Let us suppose that a subgroup of blood values are found that correspond to sick patients.
- Then, the illness of a patient can be verified by checking if his blood measurement falls into the found subgroup.
- We learned how to separate sick and healthy patients on the basis of their blood measurements.
- This is a data mining problem.

Introduction to Data Mining

Example II - apples for the market

- In a farm, human experts check apples on a conveyor belt in order to separate good apples for the market from the bad ones.
- Note that the percentage of bad apples actually removed is a function of the speed of the conveyor and of the amount of human attention.
- Alternative: use a computational system for this task.
- Advantage: a computational system cannot be induced to distraction.
- Problem: our system must learn how to recognize bad apples among the good ones.
- Solution: train the system by exploiting a set of apples whose classification in good or bad is known.
- This is a data mining problem.

Introduction to Data Mining

Example III - text mining

- Let us suppose that books related to two topics, *mathematics* and *computer science*, need to be categorized.
- We can suppose that the topic of the book can be recognized by the words used in the text.
- Solution I: all the material must be read and classified.
- Disadvantage of Solution I: if a large quantity of books is available, this procedure may require years.
- Solution II: use a computational system for performing the categorization.
- Disadvantage of Solution II: the system must be programmed so that it is able to perform the categorization.
- This is a data mining problem.

Outline

Definition of Data Mining

Definition

Data mining is a nontrivial extraction of previously unknown, potentially useful and reliable patterns from a set of data. It is the process of analyzing data from different perspectives and summarizing it into useful information.

In the previous examples:

Example	Set of data	Useful information
Blood analysis	blood measurements	markers for illness
Apples for the market	apples	markers for bad apples
Text mining	books	markers for the book topic

Definition of Data Mining

Definition

Data mining is a nontrivial extraction of previously unknown, potentially useful and reliable patterns from a set of data. It is the process of analyzing data from different perspectives and summarizing it into useful information.

In the previous examples:

Example	Set of data	Useful information
Blood analysis	blood measurements	markers for illness
Apples for the market	apples	markers for bad apples
Text mining	books	markers for the book topic

Differences among the previous examples

Are there differences among the three introduced examples?

Blood analysis

- Set of data: blood measurements;
- Question: can we have information about the illness of the patient corresponding to some blood measurements?
- Answer: yes, we can perform a different analysis on some patients and check their illnesses.
- Observation: the known pairs (blood measurement,patient illness) can be used for locating the markers for illness in the patients.
- Definition: the pairs (blood measurement,patient illness) form the so-called training set.

Differences among the previous examples

Are there differences among the three introduced examples?

Apples for the market

- Set of data: set of apples;
- Question: can we have clues on how to distinguish between good apples for the market and bad ones?
- Answer: yes, we can ask an expert to classify a subset of apples for us.
- Observation: the known pairs (apple,good/bad) can be used for locating the markers for bad apples.
- The pairs (apple,good/bad) form the training set related to this example.
- Once we learned how to identify bad apples, we do not need the help of the expert anymore.

Differences among the previous examples

Are there differences among the three introduced examples?

Text mining

- Set of data: set of books on the two topics *mathematics* and *computer science*;
- Question: can we have clues on how to distinguish between books on *mathematics* and *computer science*?
- Answer: yes, we could, but we should read part of the material.
- Question: could we avoid that?
- Answer: yes, we have to forget about a training set and try to partition our books on the basis of the inherent patterns hidden in the data.

Classification and clustering

Let X be a set of data. Let us suppose that X needs to be divided into disjoint parts, representing different features of the application at hand.

Definition

If there exists a subset $\hat{X}$ of X such that the classification of each sample $\hat{x} \in \hat{X}$ is known ($\hat{X}$ is a training set), then the problem of finding a partition for the samples in X is a classification problem.

Definition

If there are no possible training sets $\hat{X} \subset X$, then the problem of finding a partition for the samples in X is a clustering problem.

Definition

Classification and clustering are two data mining problems.

Classification and clustering

Let X be a set of data. Let us suppose that X needs to be divided into disjoint parts, representing different features of the application at hand.

Definition

If there exists a subset $\hat{X}$ of X such that the classification of each sample $\hat{x} \in \hat{X}$ is known ($\hat{X}$ is a training set), then the problem of finding a partition for the samples in X is a classification problem.

Definition

If there are no possible training sets $\hat{X} \subset X$, then the problem of finding a partition for the samples in X is a clustering problem.

Definition

Classification and clustering are two data mining problems.

Classification and clustering

Let X be a set of data. Let us suppose that X needs to be divided into disjoint parts, representing different features of the application at hand.

Definition

If there exists a subset $\hat{X}$ of X such that the classification of each sample $\hat{x} \in \hat{X}$ is known ($\hat{X}$ is a training set), then the problem of finding a partition for the samples in X is a classification problem.

Definition

If there are no possible training sets $\hat{X} \subset X$, then the problem of finding a partition for the samples in X is a clustering problem.

Definition

Classification and clustering are two data mining problems.

Outline

Techniques for classification

k Nearest Neighbor (k-NN)

It is based on the idea that, given a metric, closer samples should be similar to each other. Therefore, the classification of each sample $x \in X$ is assigned on the basis of the known classification of its neighbors $\hat{x} \in \hat{X}$.

Artificial Neural Networks (ANNs)

A network of single units (neurons) performing simple tasks is defined and trained for performing complex tasks. A neural network can be trained in order to perform classifications. The network learns how to perform a certain classification task from a training set.

Support Vector Machines (SVMs)

Basically, an SVM can classify data into two different classes, by defining a separation hyperplane in a suitable space that divides the two classes. The equation of the hyperplane can be defined on the basis of the data available in a training set.

Techniques for clustering

k-means

It tries to partition the data in clusters in which samples similar to each other are contained. The similarities are evaluated by a suitable distance metric, and the representer of each cluster is defined as the mean among all the samples in the cluster. This is a very simple method, and it has several variants.

biclustering

Biclustering techniques aims at finding suitable partitions a given set of data simultaneously on two dimensions. While standard clustering techniques consider only the set of samples and look for a suitable partition, biclustering methods partition simultaneously the set of samples, and also the set of attributes used for representing them, in biclusters.

Advertisement: Textbook in Data Mining

Data Mining in Agriculture

by **A. Mucherino, P.J. Papajorgji, P.M. Pardalos**

The book describes the latest developments in data mining, giving a particular attention to problems arising in the agricultural field.

- Clustering techniques: k-means (and most of its variants h-means, J-means, Y-means, etc.) and Biclustering techniques;
- Classification techniques: k Nearest Neighbor, Artificial Neural Networks, Support Vector Machines;
- Applications in Agriculture: wine fermentation processes, grading of fruits for the market, analysis of animal sounds for discovering diseases, and many others;
- Validation techniques: different ways for validating the results obtained by data mining techniques;
- Parallel Computing: strategies for implementing data mining techniques in a parallel computational environment;
- Programming: an entire application in C programming language is presented and discussed;
- Examples and Exercises: several examples and exercises help understanding the discussed topics.

Outline

Definition of Clustering

So, what is a clustering problem?

- Let X be a set of samples whose partition is unknown.
- Let us suppose that there is no previous knowledge about the data (no training set is available).
- Clustering is aimed at finding a partition $\{C_1, C_2, \ldots, C_K\}$ of the set of data, such that

$$X = \bigcup_{i=1}^{K} C_i, \quad \forall i, j | 1 \leq i < j \leq K \quad C_i \cap C_j = \emptyset.$$

- Each cluster represents a subset of features of the samples that it contains.
- The technique used for clustering needs to be able to identify the inherent patterns hidden in the set of data.

Outline

Definition of graph

What is a graph?

Definition

A *graph* is an ordered pair $G = (V, E)$ comprising a set V of vertices or nodes together with a set E of edges or links, which are 2-element subsets of V.

- **Undirected graph**: a graph in which edges have no orientation.
- **Directed graph** or **Digraph**: a graph $G = (V, A)$, where A is a set of *ordered* pairs of vertices, even called arcs or directed edges.
- **Weighted graph**: a graph in which numbers (weights) are assigned to each edge. It can be *directed* and *undirected*. It is denoted by $G = (V, E, w)$ or $G = (V, A, w)$, where w represents the weights.

Clustering on graphs

What if the vertices of a graph need to be partitioned?

- Let $G = (V, E)$ be a connected undirected graph.
- Let us suppose that the vertices of G need to be partitioned in clusters.
- Question: in this case, how can we evaluate the quality of a partition in clusters?
- Answer: intuitively, the best partition is the one that separates sparsely connected dense subgraphs from each other.
- Question: how can we evaluate this?
- Answer: by using one of the possible validation indices.

Les misérables

What does the best partition is the one that separates sparsely connected dense subgraphs from each other mean?

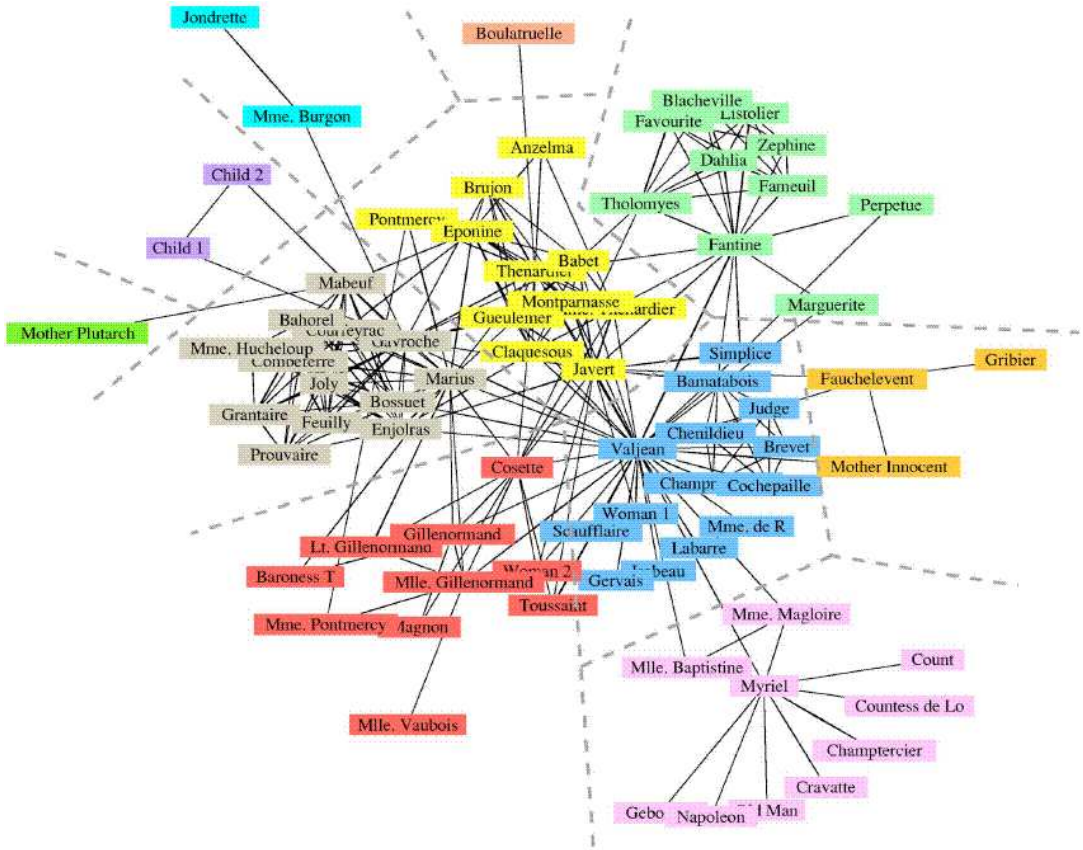


Image from Newmann and Girvan, Phisical Review E**69**, 026113, 2004.

Introducing some notations

- $G = (V, E)$, connected undirected graph.
- k, the number of vertices in V; m, the number of edges in E.
- $deg(v)$, number of edges connected to the vertex $v \in V$.
- C_i, generic cluster of V, such that $\bigcup_i C_i = V$.
- P, partition in clusters, defined as $P = \{C_1, C_2, \ldots, C_l\}$.
- $m(P)$, number of intra-cluster edges in the partition P.
- $\bar{m}(P)$, number of inter-cluster edges in the partition P.
- $\bar{m}(P)^c$, number of non-connected couples of vertices that are not in the same cluster.

Notations from Gaertler, Gorke, Wagner, LNCS **4508**, 2007.

Some validation indices

How to evaluate the quality of a partition?

- **coverage**: it measures the fraction of intra-cluster edges:

$$cov(P) = \frac{m(P)}{m} = \frac{m(P)}{m(P) + \bar{m}(P)}.$$

- **performance**: it represents the fraction of correctly classified couples of vertices:

$$perf(P) = \frac{m(P) + \bar{m}(P)^c}{\frac{1}{2}k(k-1)}.$$

- **modularity**: it gives an evaluation of the partition which is based on the actual density of the clusters compared to the expected density:

$$mod(P) = \frac{m(P)}{m} - \frac{1}{4m^2} \sum_{C_i \in P} \left(\sum_{v \in C_i} deg(v) \right)^2.$$

Graph partitioning as an optimization problem

A graph partitioning problem can be formulated as a global optimization problem:

$$\max_{P} f(P)$$

where the objective function $f(P)$ is one of the validation indices:

- coverage $cov(P)$,
- performance $perf(P)$,
- modularity $mod(P)$.

The optimization problem can be eventually subject to constraints:

- the clusters C_i cannot exceed a certain cardinality,
- ...

Outline

The T-Sale firm

T-Sale is a large multinational firm which is often employed by national governments and other large institutions to provide very large-scale services. The upper management of T-Sale noticed some inefficiencies in the way commercial offers are put together.
The following is a way for overcoming the problem.

- T-Sale keeps a detailed project database.
- the database contains information regarding the past projects of T-Sale.
- all the details of projects, from the commercial offer to the final product, are kept in the database.
- Question: can this information be exploited for having clues about the development of current projects?
- Answer: yes, a search engine for projects can be developed and embedded into the database, so that current and past projects can be quickly compared.

T-Sale contacts the VirtualClass firm

This is what T-Sale asks VirtualClass to develop:

> *We want a sort of "Google" for starting projects. We want to find all past projects which were similar at the initial stage and we want to know how they developed; this should give us some idea of future development of the current project.*

VirtualClass decides to work on this project, and it starts to plan the general architecture of the software to develop.

The name given to the software is Proogle, which stands for Project Google.

Meeting at VirtualClass

What are the functionalities the software to develop must have?

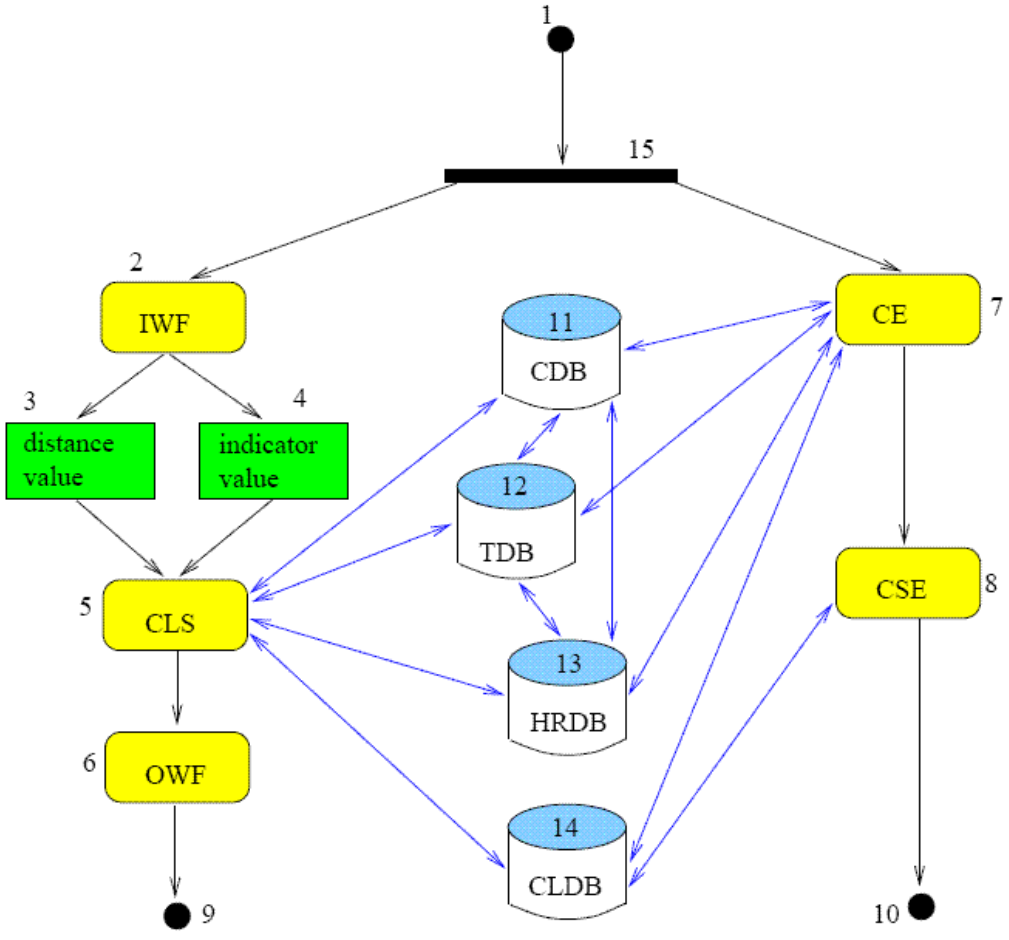
- an input/output web user interface, so that information about the new projects can be inserted online, and the results can be seen on the screen in real-time.
- a computational engine for clustering projects, in accordance with suitable indicators.
- a computational engine for evaluating the significance of the obtained clusterings.
- a computational engine for evaluating the compatibility between clusterings.
- a new database module for storing the obtained clusterings.
- a new database module for querying the stored clusterings.

Outline

The software functionalities in summary

1. Input web form (**IWF**): user inputs early indicator values concerning a new project;
2. Output web form (**OWF**): user sees similar projects with relevant indicator values;
3. Clustering engine (**CE**): given a set of objects and their pairwise distances, perform a clustering minimizing the inter-cluster distances;
4. Clustering significance evaluator (**CSE**): Given a clustering, does it match well to another given clustering?
5. Classification (**CLS**): given an indicator for a given type of clustering, find the cluster it belongs to in the given and all similar clusterings;
6. Customer's DB: split in Commercial (**CDB**), Human resources (**HRDB**), Technical (**TDB**) data repositories;
7. Clustering DB (**CLDB**): repository for existing clusterings.

Proogle project: 1st diagram



Vertices: logic anchors (black), actions (yellow), data (green), databases (blue).

Arcs: logic flow (black), data flow (blue)

Outline

Some definitions

Graph coloring

Let $G = (V, A)$ be a directed graph. A function

$$\mu : A \longrightarrow N,$$

that associates an integer number to each arc of G, is called arc coloring of G.

Observation: if we associate a color to each integer number, then the function μ associates a color to each arc of G.

Observation: note that a subgraph of G can be located by considering all the arcs having the same color:

$$H = (U, F) \quad : \quad U \subseteq V, F \subseteq A, \quad \forall e, f \in F \ \mu(e) = \mu(f).$$

Some definitions

Interface

An interface is a module in the software diagram whose only purpose is that of passing data between the other modules. The aim is to reduce the total number of connections by adding interface modules.

- let $G = (V, A)$ be a directed graph.
- let $\mu : A \longrightarrow N$ be a graph coloring.
- let $H = (U, F)$ be a densest uniformly colored subgraph associated to a color.
- the aim is to find a graph G' defined as G with the subgraph $H = (U, F)$ replaced by $H' = (U', F')$ where $U' = U \cup \{\imath\}$.
- $\imath$ represents the interface.
- the arcs $(u, \imath), (\imath, u) \in F'$ if and only if $(u, v) \in F$.
- $|F'| < |F|$.

Creating an interface by optimization

How can we find the densest uniformly colored subgraph $H = (U, F)$ of G?

- We can find it by solving a suitable optimization problem.
- Once formulated, the optimization problem can be solved by CPLEX/AMPL.
- The found densest subgraph H can then be modified by adding an interface.
- We are going to see all these steps in details, except the resolution of the problem in AMPL, which is given as exercise.

Creating an interface by optimization

Parameters

- V, set of vertices of G
- A, set of arcs of G
- μ, arc coloring of graph G
- k, a prefixed arc color

Variables

- x_v, binary, indicates if the vertex v is contained into the densest uniformly colored subgraph (U, F):

$$x_v = \begin{cases} 1 & \text{if } v \in U \\ 0 & \text{otherwise} \end{cases}$$

Creating an interface by optimization

Objective function

- The densest subgraph has the maximum number of arcs and the minimum number of vertices:

$$\max \left(\sum_{(u,v) \in A} x_u x_v - \sum_{v \in V} x_v \right)$$

Constraints

- There cannot be arcs having different colors:

$$\forall (u,v) \in A$$

$$x_u x_v \leq \min \left(\max(0, \mu(u,v) - k + 1), \max(0, k - \mu(u,v) + 1) \right)$$

Inserting the interface

Given

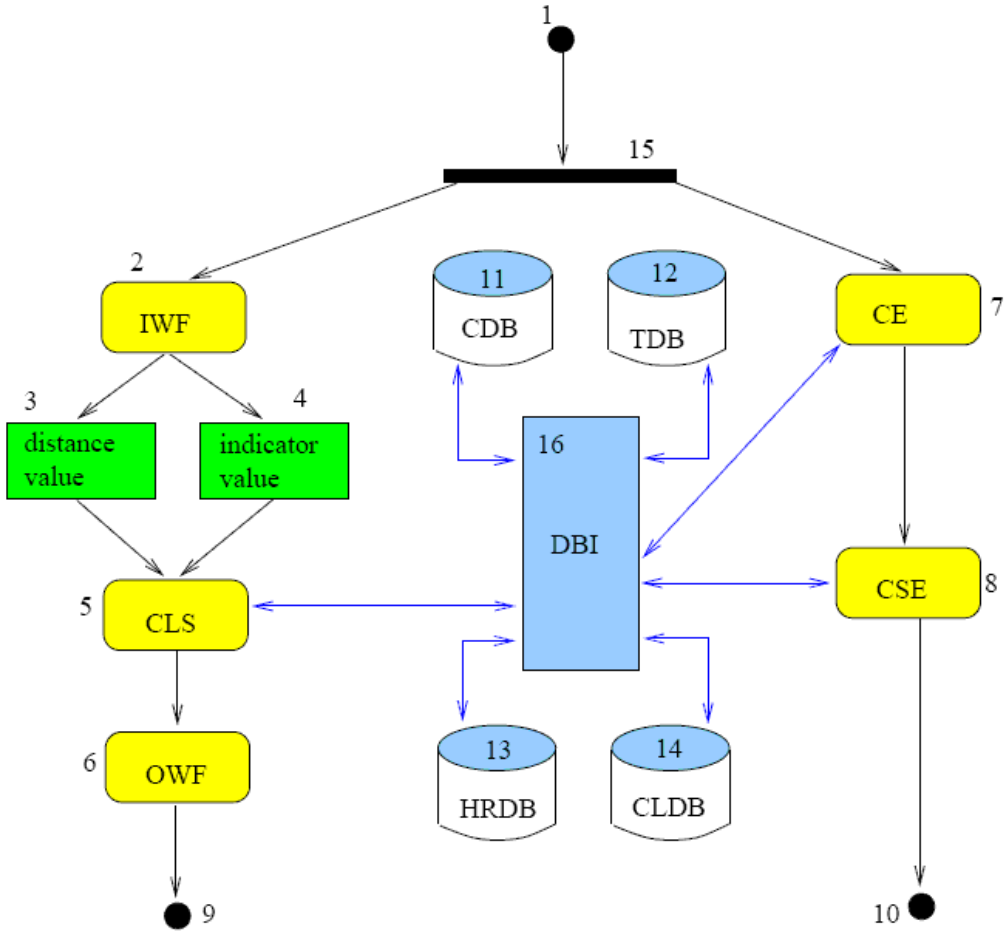
- a digraph $G = (V, A)$,
- an arc coloring μ,
- a predetermined color k,

the densest uniformly colored subgraph $H = (U, F)$ of G can be obtained by solving the previous optimization problem.

Then, a new interface can be inserted in the graph as follows:

- we add a new vertex $\imath$ in the graph, that represents the interface,
- we remove all the arcs in F,
- we add the (bidirected) arcs in $F' = \{(u, \imath) | u \in U\}$.

Proogle project: the diagram after interfacing



the interface is represented by the vertex 16 (**DBI**)

Outline

A graph partitioning problem

There are at least two good reasons for clustering the modules in a software architecture:

- to give an idea of the independent, or nearly independent, “streams” in the architecture,
- to be able to assign separate sets of modules to separate teams.

Advantages: if the inter-dependency of the clusters is minimal, the interactions among the teams are also minimal, and this helps the development process.

How to do that? Solve a graph partitioning problem on the weighted undirected graph defined by the software architecture.

Graph partitioning by optimization

How can we solve the graph partitioning problem?

- We need to find a partition in clusters of a weighted undirected graph $G = (V, E, c)$, where
  - V is the set of vertices of G,
  - E is the set of edges of G (it doesn't matter their direction),
  - c is the set of weights eventually assigned to the edges.
- We can formulate this problem as an optimization problem.
- Once formulated, the optimization problem can be solved by CPLEX/AMPL.
- The obtained solution allows us to improve the software diagram again.
- We are going to see all these steps in details, except the resolution of the problem in AMPL, which is given as exercise.

Graph partitioning by optimization

Parameters

- V, set of vertices of G
- E, set of edges of G
- c, set of weights of G
- K, number of desired clusters in the partition

Variables

- x_{uk}, binary, indicates if the vertex u is contained into the cluster $k \leq K$:

$$x_{uk} = \begin{cases} 1 & \text{if } u \in k^{th} \text{ cluster} \\ 0 & \text{otherwise} \end{cases}$$

Graph partitioning by optimization

Objective function

- We want the total weights of the edges between different clusters to be as minimum as possible:

$$\min \frac{1}{2} \sum_{k \neq l \leq K} \sum_{(u,v) \in E} c_{uv} x_{uk} x_{vl}$$

Note that this is the same as maximizing the coverage index.

Do we need to add constraints?

Yes, there are many constraints we can use.

Graph partitioning by optimization

Constraint I

- Each vertex must be assigned to only one cluster:

$$\forall u \in V \quad \sum_{k \leq K} x_{uk} = 1$$

Constraint II

- The trivial solution (all the verteces into one cluster) must be excluded:

$$\forall k \in K \quad \sum_{u \in V} x_{uk} \geq 1$$

Graph partitioning by optimization

Constraint III (in general, optional)

- Each cluster cannot exceed a certain cardinality:

$$\forall k \leq K \quad \sum_{u \in V} x_{uk} \leq C$$

Constraint IV (in general, optional)

- Vertices having different color cannot be clustered together:

$$\forall u \neq v \in V, k \neq l \leq K, x_{uk} x_{vl} \leq \gamma_{uv}$$

where

$$\gamma_{uv} = \begin{cases} 1 & \text{if } u \text{ and } v \text{ have the same color} \\ 0 & \text{otherwise} \end{cases}$$

Graph partitioning by optimization

Constraint V (in general, optional, substitutes Constraint II)

- Empty clusters can be controlled:

$$\forall k \leq K \quad \sum_{u \in V} x_{uk} \geq z_k$$

where

$$z_k = \begin{cases} 1 & \text{if cluster } k \text{ is not empty} \\ 0 & \text{otherwise} \end{cases}$$

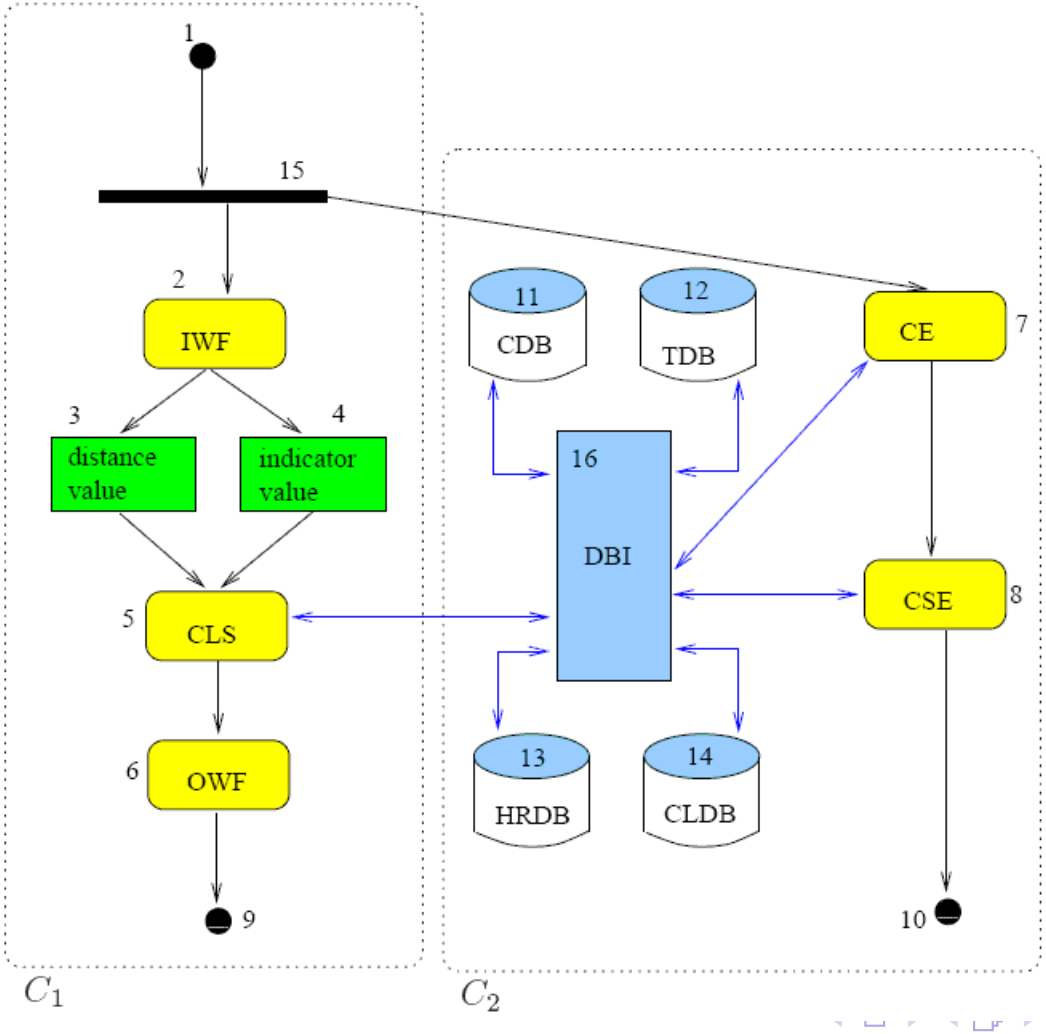
The term

$$\sum_{k \leq K} z_k$$

can be added to the objective function, in order to require the minimum possible number of clusters, by forcing some of the K clusters to be empty.

Proogle project: the diagram after clustering

This is how the software diagram can be improved after the resolution of the optimization problem



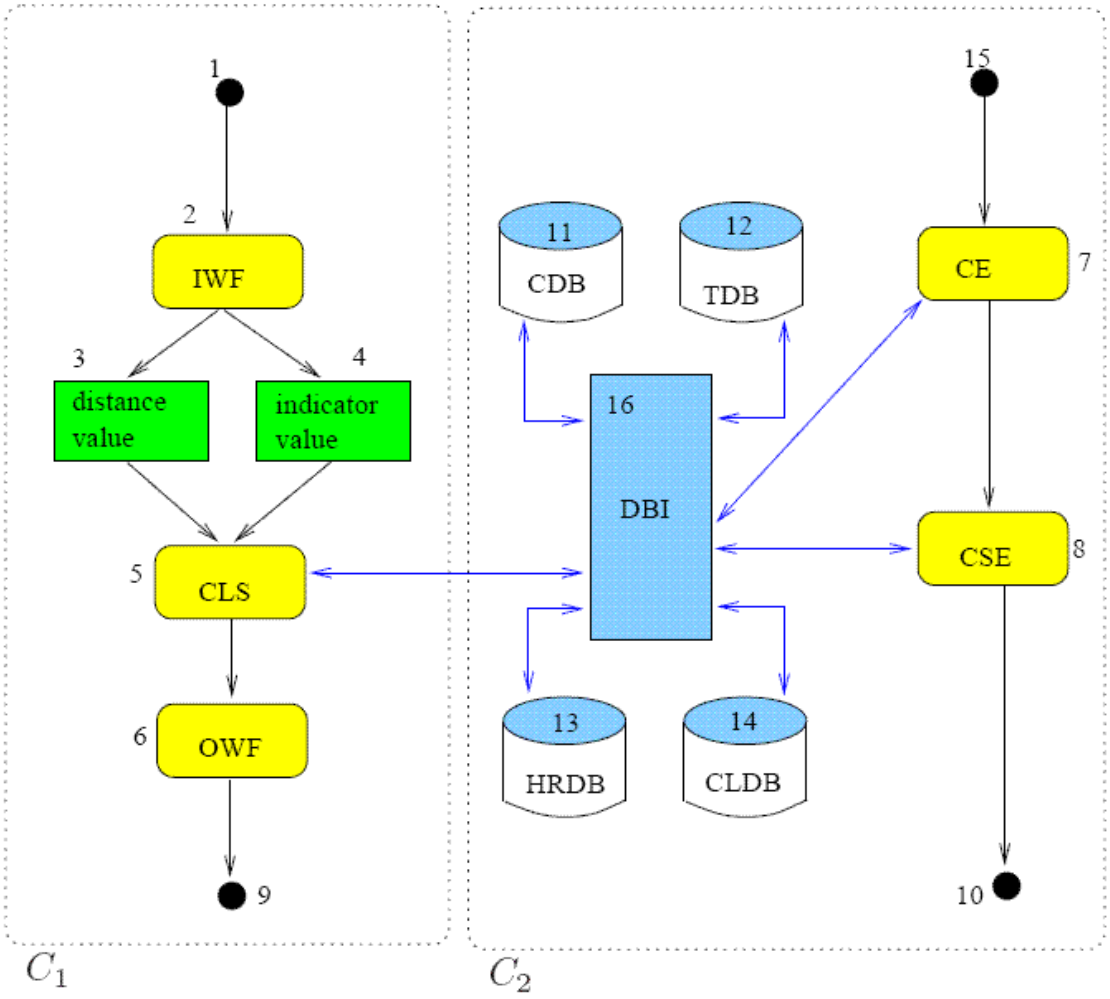
Outline

Observations

- The software architecture is composed by two main subsystems.
- The first one corresponds to activity processes performed by the user (*foreground processes*).
- The second one corresponds to activity processes performed by the program (*background processes*).
- There are only two arcs between the two clusters.
- One of them can be removed, by considering two starting points for the foreground and background processes.
- The last arc between the two clusters cannot be removed, because the classification module (**CLS**) needs to communicate with the database (**DBI**).
- This last arc suggests where the two teams working on the two processes will need to interact.

Proogle project: the final diagram

This is the final software diagram



Summary

What did we learn?

- What data mining is.
- We learned that data mining can be divided in classification and clustering.
- We analyzed in details the graph partitioning problem.
- We studied an application in which
  - a software must be developed for clustering old and current projects of an important firm,

  and, during the development of the software architecture,
  - an interface is added to it by solving an optimization problem,
  - a partition in clusters of its modules is found by optimization.

Summary

What did we learn?

- What data mining is.
- We learned that data mining can be divided in classification and clustering.
- We analyzed in details the graph partitioning problem.
- We studied an application in which
  - a software must be developed for clustering old and current projects of an important firm,

  and, during the development of the software architecture,
  - an interface is added to it by solving an optimization problem,
  - a partition in clusters of its modules is found by optimization.

Summary

What did we learn?

- What data mining is.
- We learned that data mining can be divided in classification and clustering.
- We analyzed in details the graph partitioning problem.
- We studied an application in which
  - a software must be developed for clustering old and current projects of an important firm,

  and, during the development of the software architecture,
  - an interface is added to it by solving an optimization problem,
  - a partition in clusters of its modules is found by optimization.

Summary

What did we learn?

- What data mining is.
- We learned that data mining can be divided in classification and clustering.
- We analyzed in details the graph partitioning problem.
- We studied an application in which
  - a software must be developed for clustering old and current projects of an important firm,

and, during the development of the software architecture,

  - an interface is added to it by solving an optimization problem,
  - a partition in clusters of its modules is found by optimization.

Summary

What did we learn?

- What data mining is.
- We learned that data mining can be divided in classification and clustering.
- We analyzed in details the graph partitioning problem.
- We studied an application in which
  - a software must be developed for clustering old and current projects of an important firm,

  and, during the development of the software architecture,
  - an interface is added to it by solving an optimization problem,
  - a partition in clusters of its modules is found by optimization.

Bibliography


I.S. Dhillon and D.M. Modha, *Concept Decompositions for Large Sparse Text Data using Clustering*, Machine Learning **42**(1), 143–175, 2001.


M. Gaertler, R. Gorke and D. Wagner, *Significance-Driven Graph Clustering*, Lecture Notes in Computer Science **4508**, 11–26, 2007.


S. Gunter, H. Bunke, *Validation Indices for Graph Clustering*, Pattern Recognition Letters **24**, 1107–1113, 2003.


L. Liberti, *Software Modelling and Architecture: Exercises*, exercise book for the students of the École Polytechnique.


A. Mucherino, P.J. Papajorgji, P.M. Pardalos, *Data Mining in Agriculture*, Springer, 2009.


M.E.J. Newman and M. Girvan, *Finding and Evaluating Community Structure in Networks*, Physical Review E**69**, 026113, 2004.


M.A. Shahin, E.W. Tollner, R.W. McClendon, *Artificial Intelligence Classifiers for Sorting Apples based on Watercore*, Journal of Agricultural Engineering Research **79**(3), 265–274, 2001.