Machine Learning: Pattern Mining

Steffen Rendle

Information Systems and Machine Learning Lab (ISMLL)
University of Hildesheim

Wintersemester 2010 / 2011

Pattern Mining
- Overview

Itemsets
- Task
- Naive Algorithm
- Apriori Algorithm
- Data Structure
- Eclat Algorithm

Association Rules
- Task
- Algorithm

Summary

Overview

Pattern Mining discovers regularities in data.

- Example: a transaction database of a supermarket: *someone who buys chips also buys beer*.
- Frequent patterns are found by counting the occurences in the data base.
- Types of patterns: itemsets, association rules, sequences, ...

Example

Shopping Carts
Beer, Chips, Chocolate, Cookies
Coke, Beer, Pizza, Chips
Salad, Noodles, Tomatoes, Water
Lasagne, Coke, Beer, Chips
Oranges, Apple Juice, Rice, Cabbage, Sausage
Diapers, Beer, Charcoal, Sausage
Beer, Cabbage, Sausage, Chips
...

Example

Shopping Carts
Beer, Chips, Chocolate, Cookies
Coke, Beer, Pizza, Chips
Salad, Noodles, Tomatoes, Water
Lasagne, Coke, Beer, Chips
Oranges, Apple Juice, Rice, Cabbage, Sausage
Diapers, Beer, Charcoal, Sausage
Beer, Cabbage, Sausage, Chips
...

Observations:

- Many customers buy beer.
- Beer and chips are often bought together.
- Customers who buy cabbage also buy sausage.
- Customers who buy something to eat also buy something to drink.

Outline

- **Classification** predicts class labels based on training data
- **Clustering** groups data based on similarity
- **Pattern Mining** discovers regularities in data

Itemsets

- Which itemsets frequently occur in the same transaction?
- Example: chips and beer are frequently bought together
- given
  - Items $I = \{i_i, \ldots, i_m\}$
  - Data $D \subseteq \mathcal{P}(I)$ multiset
  - Frequency threshold θ_s
- to find
  - Frequent sets $L = \{X \in \mathcal{P}(I) | support_D(X) \geq \theta_s\}$

Definitions and Terms

- $support_D(X) = \frac{|\{d \in D | X \subseteq d\}|}{|D|}$
- X is **frequent** / **large** iff $support_D(X) \geq \theta_s$

Naive Algorithm

function $\textsc{Naive}(D, \theta_s)$
    $L \leftarrow \emptyset$
    for all $X \in \mathcal{P}(I)$ **do**
        if $support_D(X) \geq \theta_s$ **then**
            $L \leftarrow L \cup \{X\}$
        end if
    end for
    return L
end function

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

find itemsets with $\theta_s \geq 0.3$

X frequent $\Leftrightarrow \#_D(X) > 2$

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

$\mathcal{P}(I)$	#
a	?
b	?
c	?
d	?
e	?
a,b	?
a,c	?
a,d	?
a,e	?
b,c	?
b,d	?
b,e	?
c,d	?
c,e	?

$\mathcal{P}(I)$	#
d,e	?
a,b,c	?
a,b,d	?
a,b,e	?
b,c,d	?
b,c,e	?
c,d,e	?
a,b,c,d	?
a,b,c,e	?
a,b,d,e	?
a,c,d,e	?
b,c,d,e	?
a,b,c,d,e	?

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

$\mathcal{P}(I)$	#
a	7
b	7
c	6
d	2
e	5
a,b	5
a,c	4
a,d	1
a,e	5
b,c	4
b,d	2
b,e	4
c,d	1
c,e	2

$\mathcal{P}(I)$	#
d,e	1
a,b,c	2
a,b,d	1
a,b,e	4
b,c,d	1
b,c,e	1
c,d,e	0
a,b,c,d	0
a,b,c,e	1
a,b,d,e	1
a,c,d,e	0
b,c,d,e	0
a,b,c,d,e	0

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

$\mathcal{P}(I)$	#
a	7
b	7
c	6
d	2
e	5
a,b	5
a,c	4
a,d	1
a,e	5
b,c	4
b,d	2
b,e	4
c,d	1
c,e	2

$\mathcal{P}(I)$	#
d,e	1
a,b,c	2
a,b,d	1
a,b,e	4
b,c,d	1
b,c,e	1
c,d,e	0
a,b,c,d	0
a,b,c,e	1
a,b,d,e	1
a,c,d,e	0
b,c,d,e	0
a,b,c,d,e	0

$$L = \{\{a\}, \{b\}, \{c\}, \{e\}, \{a, b\}, \{a, c\}, \{a, e\}, \{b, c\}, \{b, e\}, \{a, b, e\}\}$$

Properties of Naive Algorithm

- returns correct result
- always terminates
- **But**: counting support for each itemset $X \subset \mathcal{P}(I)$ is not applicable as $|\mathcal{P}(I)|$ is exponential in $|I|$

Observations

$\text{support}_D(X) \geq \text{support}_D(X \cup Y)$

- $\text{support}_D(X) \geq \theta_s \Rightarrow \forall Y : Y \subset X : \text{support}_D(Y) \geq \theta_s$
  „all subsets of a frequent set are frequent“
- $\text{support}_D(X) < \theta_s \Rightarrow \forall Y : Y \supset X : \text{support}_D(Y) < \theta_s$
  „all supersets of an infrequent set X are not frequent“
- example: $\text{support}_D(\{a, b\}) \geq \text{support}_D(\{a, b, c, d\})$

Apriori Algorithm

- Breadth-first/ levelwise search
  1. find frequent itemsets of length 1
  2. find frequent itemsets of length 2
  3. ...

- only explores itemsets where all subsets are known to be frequent

Apriori Algorithm

function $\text{Apriori}(D, \theta_s)$
  $k \leftarrow 1$
  $L_k \leftarrow \{\{i\} | i \in I, \text{support}_D(\{i\}) \geq \theta_s\}$
  while $L_k \neq \emptyset$ **do**
    $C_{k+1} \leftarrow \text{generateCandidates}(L_k, k+1)$
    $L_{k+1} \leftarrow \{X \in C_{k+1} | \text{support}_D(X) \geq \theta_s\}$
    $k \leftarrow k+1$
  end while
  return $\bigcup_{k=1}^{\infty} L_k$
end function

Candidate Generation

generates candidates of length k from frequent itemsets L of length $k-1$

function $\textsc{generateCandidates}(L, k)$
$C \leftarrow \{X \cup Y | X, Y \in L \wedge |X \cup Y| = k\}$
$C \leftarrow \{X \in C | \forall Y \subset X : |Y| = k-1 \Rightarrow Y \in L\}$
return C
end function

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

find itemsets with $\theta_s \geq 0.3$

X frequent $\Leftrightarrow \#_D(X) > 2$

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_1	#
a	?
b	?
c	?
d	?
e	?

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_1	#
a	7
b	7
c	6
d	2
e	5

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_1	#
a	7
b	7
c	6
d	2
e	5

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_1	#
a	7
b	7
c	6
d	2
e	5

L_1
a
b
c
e

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_2	#
a,b	?
a,c	?
a,e	?
b,c	?
b,e	?
c,e	?

L_1
a
b
c
e

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_2	#
a,b	5
a,c	4
a,e	5
b,c	4
b,e	4
c,e	2

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_2	#
a,b	5
a,c	4
a,e	5
b,c	4
b,e	4
c,e	2

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_2	#
a,b	5
a,c	4
a,e	5
b,c	4
b,e	4
c,e	2

L_2
a,b
a,c
a,e
b,c
b,e

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_3	#
a,b,c	?
a,b,e	?
a,c,e	?
b,c,e	?

L_2
a,b
a,c
a,e
b,c
b,e

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_3	#
a,b,c	?
a,b,e	?

Pruning: $\{c, e\} \notin L_2$

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_3	#
a,b,c	2
a,b,e	4

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_3	#
a,b,c	2
a,b,e	4

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_3	#
a,b,c	2
a,b,e	4

L_3
a,b,e

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

C_4	#
-	-

L_3
a,b,e

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

L_1
a
b
c
e

L_2
a,b
a,c
a,e
b,c
b,e

L_3
a,b,e

$$L = \{\{a\}, \{b\}, \{c\}, \{e\}, \{a, b\}, \{a, c\}, \{a, e\}, \{b, c\}, \{b, e\}, \{a, b, e\}\}$$

Trie / Prefix Tree

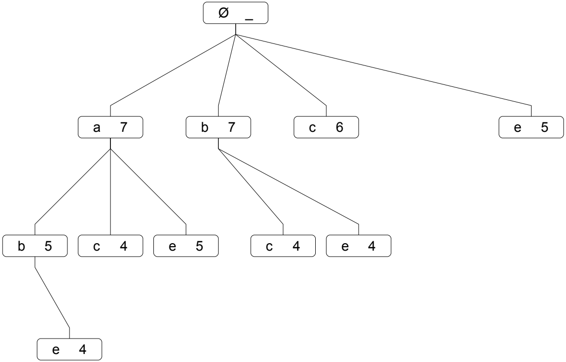
For candidate generation and frequency counting, a trie can be used:

- a trie is a tree
- each node contains an item and a frequency counter
- each path from the root to a node corresponses to an itemset
- the k-th level represents itemsets of length k
- the items in a trie are ordered

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

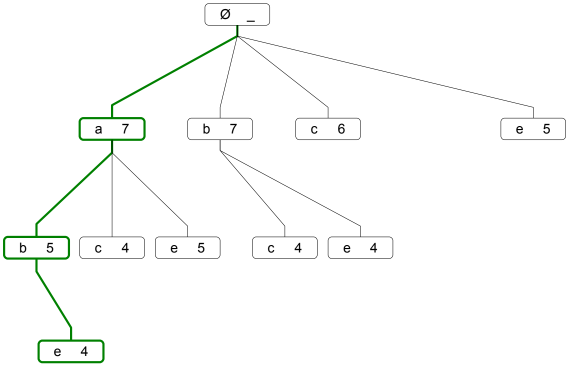
Prefix Tree



Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

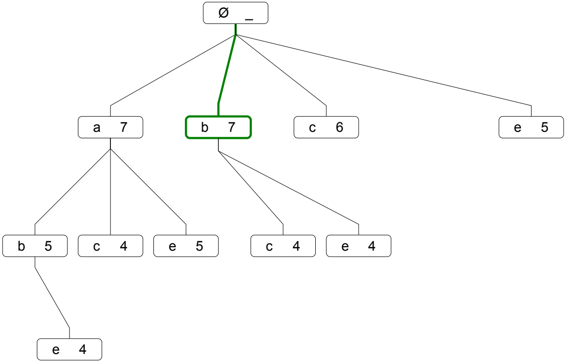
$\#\{a,b,e\} = 4$



Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

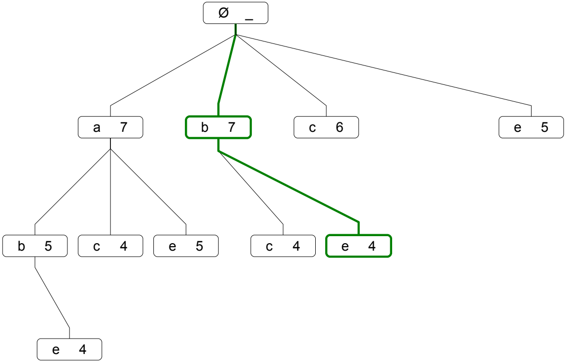
$\#\{b\} = 7$



Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

$\#\{b,e\} = 4$



Trie: Frequency Counting

To count frequencies with a trie, each transaction $d \in D$ is handled the following way:

1. sort d
2. start at the root
3. for each item $i \in d$ follow the node i, increase it by one and recursively repeat this for $d \setminus \{i\}$

Trie: Candidate Generation

Candidates of length k can be generated from a trie of depth $k-1$:

1. for each node at level $k-1$ append its siblings
2. prune infrequent childs

Trie: Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

find itemsets with $\theta_s \geq 0.3$

X frequent $\Leftrightarrow \#_D(X) > 2$

... see blackboard ...

Eclat Algorithm

- Algorithm for itemset mining
- Depth-first algorithm
- Vertical data base layout
  - For each pattern: store the cover, i.e. all transactions that include this pattern. e.g. $(a, \{d_1, d_3, d_4, d_5, d_7, d_8, d_9\})$
  - Count frequency by intersection

Eclat Algorithm

function $\textsc{Eclat}(D, \theta_s)$
$$C_\emptyset = \{(i, \{d \in D | i \in d\}) | i \in I\}$$
$$L_\emptyset = \left\{(i, D_i) \in C_\emptyset | \frac{|D_i|}{|D|} \geq \theta_s\right\}$$
return $\textsc{EclatRecursion}(L_\emptyset, \emptyset, \theta_s)$
end function

Eclat Algorithm

function $\textsc{EclatRecursion}(L, p, \theta_s)$
$F \leftarrow \emptyset$
for all $(i, D_i) \in L$ **do**
$q \leftarrow p \cup \{i\}$
$F \leftarrow F \cup \{p\}$
$C_q \leftarrow \{(j, D_i \cap D_j) | (j, D_j) \in L, j > i\}$
$L_q \leftarrow \left\{(k, D_k) \in C_q | \frac{|D_k|}{|D|} \geq \theta_s\right\}$
if $L_q \neq \emptyset$ **then**
$F \leftarrow F \cup \textsc{EclatRecursion}(L_q, q, \theta_s\}$
end if
end for
return F
end function

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

find itemsets with $\theta_s \geq 0.3$

X frequent $\Leftrightarrow \#_D(X) > 2$

... see blackboard ...

Association Rules

- Which itemssets Y occur often if another itemset X appears?
  $X \Rightarrow Y$
- Example: a customer buying diapers also buys beer
  $\{\text{diapers}\} \Rightarrow \{\text{beer}\}$
- given
  - Items $I = \{i_i, \ldots, i_m\}$
  - Data $D \subseteq \mathcal{P}(I)$ multiset
  - Frequency thresholds θ_s
  - Confidence threshold θ_c
- to find
  - Rules $R = \{X \Rightarrow Y | \mathit{support}_D(X \Rightarrow Y) \geq \theta_s \wedge \mathit{confidence}_D(X \Rightarrow Y) \geq \theta_c\}$

Definitions and Terms

- **support** measures how often the rule $X \Rightarrow Y$ appears
  - $\mathit{support}_D(X \Rightarrow Y) = \mathit{support}_D(X \cup Y)$
  - $X \Rightarrow Y$ is **frequent** / **large** iff $\mathit{support}_D(X \Rightarrow Y) \geq \theta_s$
- **confidence** measures how likely it is that Y appears if X is present.
  - $\mathit{confidence}_D(X \Rightarrow Y) = \frac{\mathit{support}_D(X \Rightarrow Y)}{\mathit{support}_D(X)}$
- for a rule $X \Rightarrow Y$, Y is called **head** and X is called **body**

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

find rules with $\theta_s \geq 0.3$

X frequent $\Leftrightarrow \#_D(X) > 2$

find rules with $\theta_c \geq 0.8$

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

$$confidence_D(X \Rightarrow Y) = \frac{support_D(X \Rightarrow Y)}{support_D(X)}$$

$$support_D(X \Rightarrow Y) = support_D(X \cup Y)$$

$$L = \{\{a\}, \{b\}, \{c\}, \{e\}, \{a, b\}, \{a, c\}, \{a, e\}, \{b, c\}, \{b, e\}, \{a, b, e\}\}$$

... see blackboard ...

Example

Data D
a,b,e
b,c
a,c,e
a,b,c,e
a,b,d,e
b,c,d
a,b,c
a,c
a,b,e

$$R = \{e \Rightarrow a, e \Rightarrow b, e \Rightarrow ab, ab \Rightarrow e, ae \Rightarrow b, be \Rightarrow a\} \cup \{X \Rightarrow \emptyset | X \in L\}$$

Observations

- expanding the head of a rule by an item of the body, results in a rule with less or equal confidence.

$$\text{confidence}_D(X \setminus Z \Rightarrow Y \cup Z) \leq \text{confidence}_D(X \Rightarrow Y)$$

- proof:

$$\begin{aligned}
&\text{confidence}_D(X \setminus Z \Rightarrow Y \cup Z)\\
=&\frac{\text{support}_D((X \setminus Z) \cup (Y \cup Z))}{\text{support}_D(X \setminus Z)} = \frac{\text{support}_D(X \cup Y \cup Z)}{\text{support}_D(X \setminus Z)}\\
\leq&\frac{\text{support}_D(X \cup Y)}{\text{support}_D(X)} = \text{confidence}_D(X \Rightarrow Y)
\end{aligned}$$

- example:
  $\text{confidence}_D(\{a, b\} \Rightarrow \{c, d\}) \leq \text{confidence}_D(\{a, b, c\} \Rightarrow \{d\})$

Algorithm

Association rule mining is done in two steps:

1. find frequent itemsets (see itemset mining)
2. extract rules from the frequent itemsets

AssociationRules Algorithm

function $\textsc{AssociationRules}(D, \theta_s, \theta_c)$
    $L \leftarrow \text{Apriori}(D, \theta_s)$
    $R \leftarrow \emptyset$
    for all $I \in L$ **do**
        $k \leftarrow 1$
        $C_k \leftarrow \{\{i\} | i \in I\}$
        while $C_k \neq \emptyset$ **do**
            $H_k \leftarrow \{X \in C_k | \text{confidence}_D(I \setminus X \Rightarrow X) \geq \theta_c\}$
            $C_{k+1} \leftarrow \text{generateCandidateHeads}(H_k, k+1)$
            $k \leftarrow k+1$
        end while
        $$R \leftarrow R \cup \{I \setminus X \Rightarrow X | X \in \bigcup_{k=1}^{\infty} H_k\} \cup \{I \Rightarrow \emptyset\}$$
    end for
    return R
end function

Candidate Generation for Heads of Rules

generates candidate heads of length k from heads H of length $k-1$

function $\textsc{generateCandidateHeads}(H, k)$
  $C \leftarrow \{X \cup Y | X, Y \in H \wedge |X \cup Y| = k\}$
  $C \leftarrow \{X \in C | \forall Y \subset X : |Y| = k-1 \Rightarrow Y \in H\}$
  return C
end function

Remarks

- Calculating the confidence can be reduced to calculating the support:

$$\text{confidence}_D(I \setminus X \Rightarrow X) = \frac{\text{support}_D((I \setminus X) \cup X)}{\text{support}_D(I \setminus X)} \geq \theta_c$$

$$\Leftrightarrow \frac{\text{support}_D(I)}{\text{support}_D(I \setminus X)} \geq \theta_c$$

$$\Leftrightarrow \text{support}_D(I \setminus X) \leq \frac{1}{\theta_c} \text{support}_D(I)$$

- If $\theta_c \geq \theta_s$, the values for support_D can be looked up in the trie and no database pass is necessary.

Example

Trace of inner loop of the algorithm $\textsc{AssociationRules}$ for $I = \{a, b, e\}$.

... see blackboard ...

Outlook

- Extensions to Apriori and Eclat
- Further pattern: sequences, trees, ...
- Background knowledge: e.g. taxonomies

Sequence mining

Takes the time into account, when an action is performed. E.g.

- A database of courses attended by a student in one term, i.e. sequences of sets:
  - Student1: ({linear algebra, c++, algorithm theory}, {machine learning, numerics, economics}, {bayessian networks})
  - Student2: ({linear algebra, java}, {software engineering}, {numerics})
  - Student3: ({linear algebra, java, algorithm theory}, {economics}, {machine learning, numerics}, {bayessian networks})
  - ...
- A frequent sequence might be ({linear algebra, algorithm theory}, {machine learning, numerics}, {bayessian networks})

Use taxonomies

Background knowledge in terms of taxonomies might be used for mining patterns. E.g.

- The following taxonomy is given over subjects
  - *linear algebra* `isa` *mathematics*
  - *mathematics* `isa` *science*
  - *computer science* `isa` *science*
- In the student database one could mine the association rule using the taxonomy:
  if someone has attended machine learning then (s)he also has attended some mathematic lecture
  $\{\text{machine learning}\} \Rightarrow \{\text{mathematics}\}$

Conclusion

- Task: Finding frequent patterns in database.
- Efficient algorithms explore only promising candidates by pruning.
- Mining association rules can be reduced to mining itemsets with an additional post processing step.

Literature

R. Agrawal and R. Srikant.
Fast algorithms for mining association rules.
In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

R. Agrawal and R. Srikant.
Mining sequential patterns.
In P. S. Yu and A. S. P. Chen, editors, *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.

L. Schmidt-Thieme.
Algorithmic features of eclat.
2004.