Defining inductive operators using distances over lists

Authors: V. Estruch, C. Ferri, J. Hernández-Orallo & M.J. Ramírez-Quintana
(Univ. Politècnica de València)

3rd WS on Approaches and Applications of Inductive Programming

Index of Contents

- Introduction
- Motivation
- Distance-based generalisation (*dbg*) operators
- *Dbg* operators for lists
- Future work

Generalising data embedded in a metric space

Index of Contents

Lists are all round

- Bioinformatics
- Text mining
- Command line completion
- Ortographic correctors

Introduction

Learning from lists

Distance-based methods

• **Inductive bias:** near examples

K-Means, etc.

PROS	CONS
-Algorithms can be adapted to any data representation	-No or little expressive hypothesis

Index of Contents

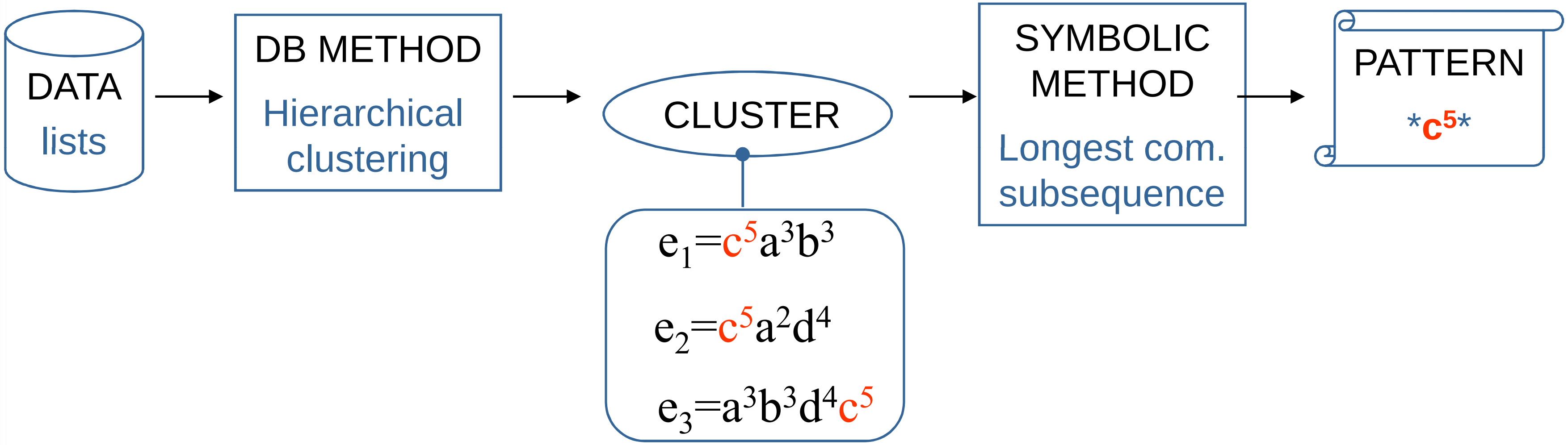
- Introduction
- Motivation
- Distance-based generalisation (*dbg*) operators
- *Dbg* operators lists
- Future work

Motivation

Could it be possible to transform distances into patterns?

Motivation

- Naive approach: db method + symbolic method (pattern)



Motivation

Little certainty about the consistency between the distance and the patterns.

Index of Contents

- Introduction
- Motivation
- Distance-based generalisation (*dbg*) operators
- *Dbg* operators for lists
- Future work

Distance-based generalisation operators

- Proposed approach:
  - Distances count differences between objects
  - Patterns drop differences between objects
  - So, drop what you count !

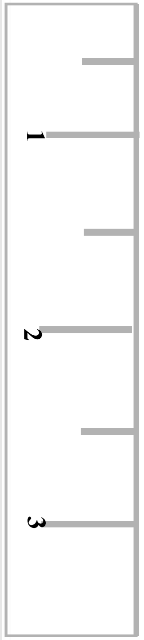
? How can we formalise the **relation of consistency** between patterns and distances?

  - It must be independent of the data/pattern language and the distance definition

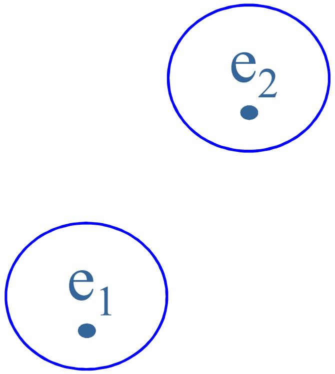
Distance-based generalisation operators

Projecting patterns in metric spaces

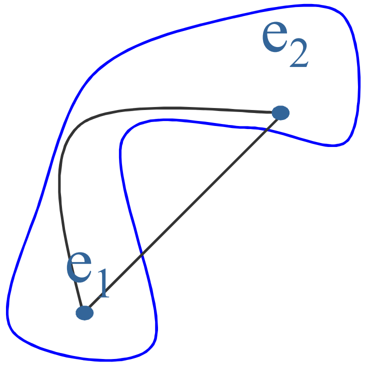
Making patterns and distance agree



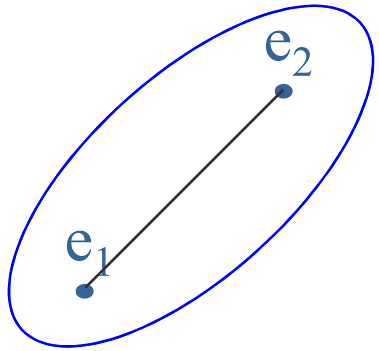
Near elements should be covered (closed balls)



Near elements lying in smooth paths should be covered (ε-path)



Intrinsic paths must be covered (*intermediate elements*)



Distance-based generalisation operators

Definition

Binary distance-based (db) pattern

Given E={e_1, e_2}, a pattern p is a binary *db* pattern of E, if

p covers <u>all</u> the intermediate elements of e_1 and e_2.

Definition

Binary distance-based generalisation (dbg) operator

Adittionally, Δ is a binary *dbg* operator if,

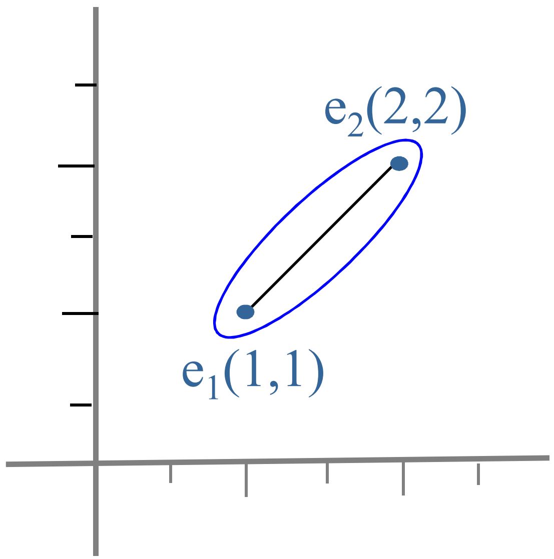
$\Delta(e_1,e_2)$ is a binary *db* pattern, for every e_1 and e_2.

.

Distance-based generalisation operators

Playing with patterns and distances

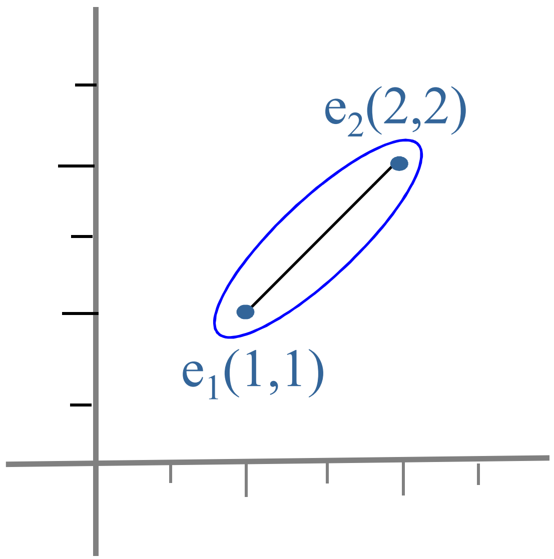
$$d(x,y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2}$$



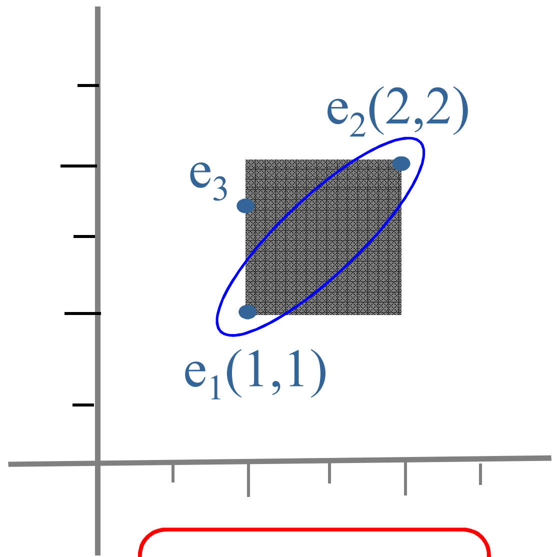
Distance-based generalisation operators

Playing with patterns and distances

$$d(x,y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2}$$



$$d(x,y)=|x_1-y_1|+|x_2-y_2|$$



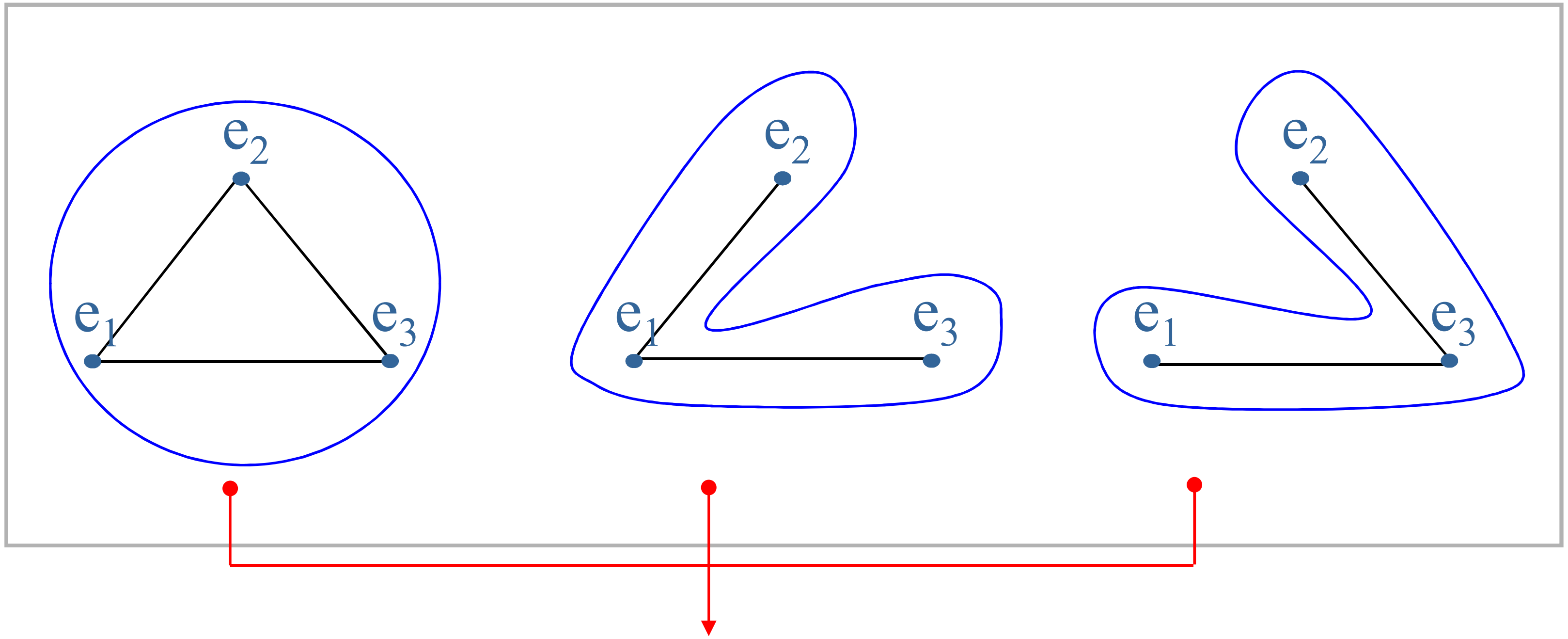
$e_3(1,1.75)$ is in between!

$d(e_1,e_3)+d(e_3,e_2) = 0.75 + 1.25 = 2 = d(e_1,e_2)$

Distance-based generalisation operators

Moving to n-ary generalisations

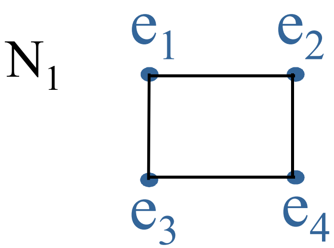
- Generalisation can be an *n*-ary operator but distance is binary

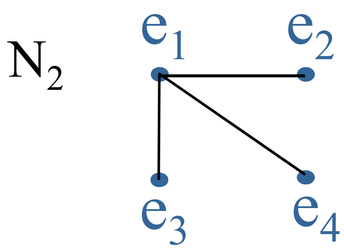


Distance-based generalisation operators

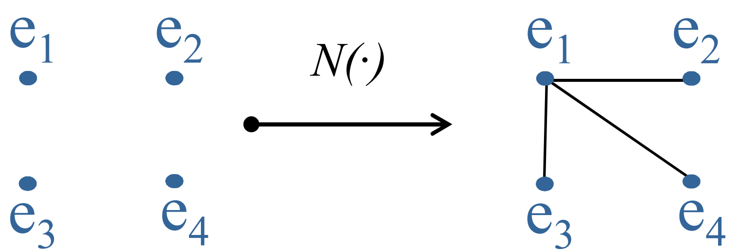
Moving to n-ary generalisations

- ***Nerve***: undirected connected graph whose vertices correspond to examples

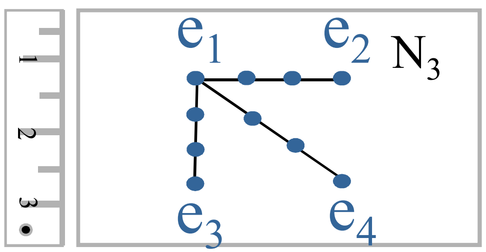




- ***Nerve function***: from examples to nerves



- ***Skeleton(N_i)***: filling the nerve $(\forall (e_i, e_j) \in N_i, \forall e \in X : \text{if } e \text{ is between } e_i \text{ and } e_j \Rightarrow e \in skeleton(N_i))$



Distance-based generalisation operators

Moving to n-ary generalisations

Definition

N-ary *db* pattern

Given a finite set of elements *E*, a pattern *p* is a *n*-ary *db* pattern of *E*, if

there exists a nerve ν of *E* such that $skeleton(\nu) \subset Set(p)$

Definition

N-ary *distance-based generalisation (dbg)* operator

Adittionally, Δ is a *n*-ary *dbg* operator, if

$\Delta(E)$ is a *n*-ary db pattern of *E* (for every *E*)

Distance-based generalisation operators

Moving to n-ary generalisations

Definition

N-ary *db* pattern relative to a nerve ν

Given a finite set of elements *E*, *p* is a *n*-ary *db* pattern of *E relative to* ν, if

$$skeleton(\nu) \subset Set(p)$$

Definition

N-ary *dbg* operator relative to a nerve function *N*

Additionally, Δ is a *n*-ary *dbg* operator relative to *N*, if

$\Delta(E)$ is a *n*-ary *db* pattern relative to *N(E)* (for every finite set *E*)

Distance-based generalisation operators

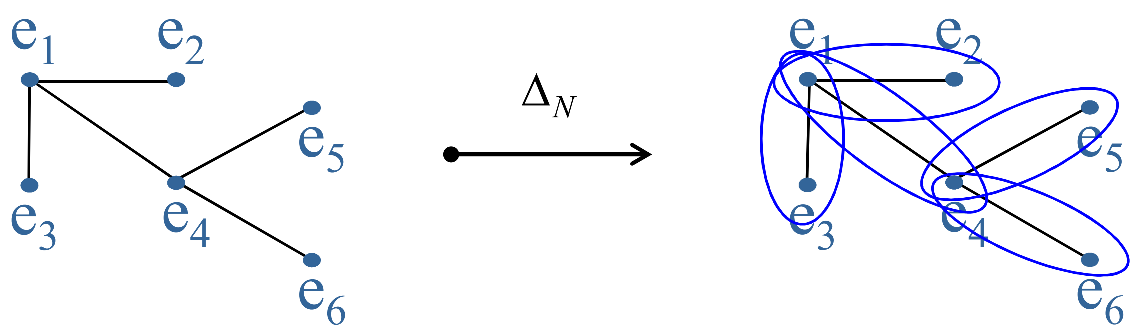
From binary to n-ary *db* generalisations

Proposition

Let $\mathcal{L}$ be a pattern language endowed with the operation + and let Δ^b be a binary *dbg* operator in $\mathcal{L}$. Given a finite set of elements *E* and a nerve function *N*, then

$$\Delta_N(E) = \sum_{(e_i, e_j) \in N(E)} \Delta^b(e_i, e_j)$$

is a *dbg* operator w.r.t. *N*.



Minimal dbg operators

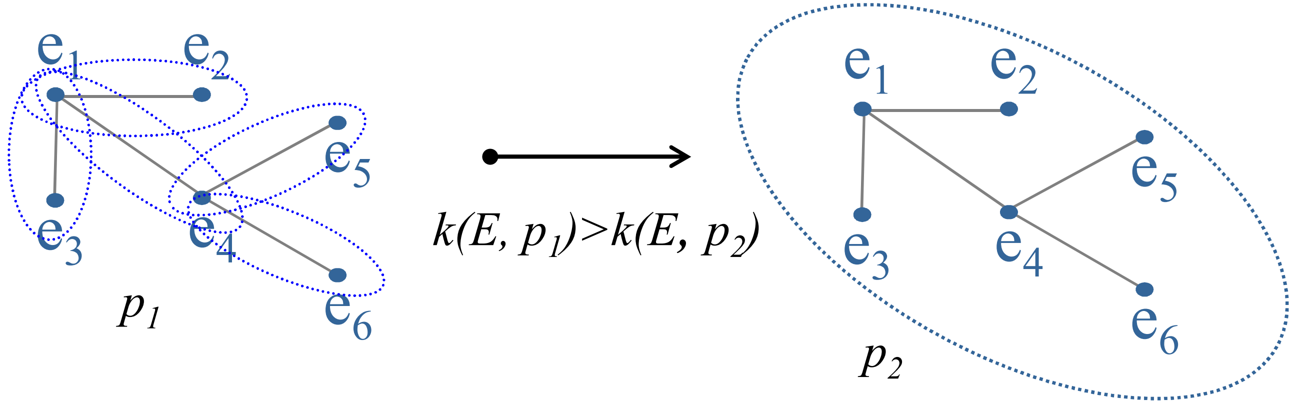
How to organise the hypothesis space?

Minimal dbg operators

- A *db* cost function is introduced (a *db* MML/MDL formulation)

$$K(E,p) = \underbrace{c(E|p)}_{\text{Semantic cost function}} + \underbrace{c(p)}_{\text{Syntactic cost function}}$$

- Hypotheses are organised according to its fitness (in terms of the distance) and (if necessary) complexity



Minimal dbg operators

- c(E|p) can be expressed as:

$\mathcal{I}$	c(E|p)	Description
Any	$\sum r_e$ $r_e = \inf_{r \in R} B(e,r) \not\subset Set(p)$	Uncovered balls of infimum radius
Any	$\sum r_e$ $r_e = \sup_{r \in R} B(e,r) \subset Set(p)$	Covered balls of supremum radius
Sets with border	$\sum \min_{e' \in \partial Set(p)} d(e,e')$	Minimum to the border
Set(p) is a bound set	$\sum \min_{e' \in \partial Set(p)} d(e,e') + \max_{e'' \in \partial Set(p)} d(e,e'')$	Minimum and maximum to the border

Minimal dbg operators

- c(p) can be expressed as:

Sort of data	*L*	c(p)	Example
Numerical	Closed intervals	Length of the interval	c([a,b])=b-a
Finite lists over an alphabet of symbols	Patterns built from the alphabet and variable symbols	Number of symbols in the pattern	$c(V_0abV_1V_2)=5$
First order atoms	Herbrand base with variables	Number of symbols	c(q(a,X,X))=4
Any	Any	Constant function	c(p)=constant

Minimal dbg operators

Definition

Minimal distance-based generalisation (*mdbg*) operators

Given a cost function k, Δ is a *mdbg* operator, if

$$k(E, \Delta(E)) \leq k(E, \Delta'(E)), \text{ for every } E \text{ and } dbg\ \Delta'$$

Definition

Mdbg operator relative to a nerve function *N*

Additionally, given a nerve function N, Δ is a *mdbg* operator relative to N, if

$$k(E, \Delta(E)) \leq k(E, \Delta'(E)), \textit{ for every } E \textit{ and } dbg\ \Delta' \text{ relative to } N$$

Index of Contents

- Introduction
- Motivation
- Distance-based generalisation (*dbg*) operators
- *Dbg* operators for lists ←
- Future work

Dbg operators for lists

Preliminaries

- ***Metric space*** (X,d)
  - $X = \sum^{*}$ E.g. X = { a, aa,..., ab, abb, ...}
  - d ≡ Edit distance where *ins* = *del* = 1
- **↑-*Transformation*** (binary operator)
  - $p_1 = V^3bcV^2$
  - $p_2 = V^2caV^3$
  - $p_3 = \uparrow(p_1,p_2) = V^4cV^4$
- ≤ (***strategy*** to apply ↑(·,·) over an n-ary set of patterns)
  - $\{p_i\}_{i=1..n}$, $S=\{a_j \text{ in } \sum: a_j \text{ in } Seq(p_i)_{1\le i\le n}\}$
  - ≤ ≡ Find p_i, p_j: exists a_k in S and a_k in $Seq(\uparrow(p_i,p_j))$

Dbg operators for lists

Setting 1

Pattern language	Cost Function
$\mathcal{L}_0$: *lists with variables* $p=a_1a_2V_1V_2V_3$	$k_0\ (E,p) = c'(E\|p)$

Proposition

Let P be the set all of the *optimal alignment patterns* of the lists e_i and e_j. Given a nerve function N then

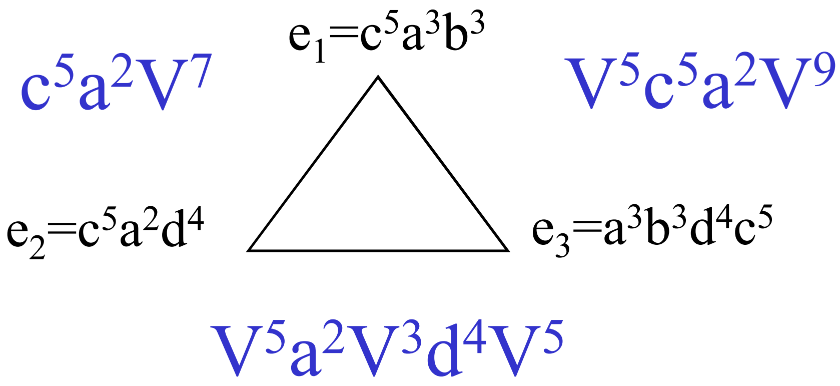
$$\Delta^b(e_i, e_j) = \uparrow((P, \leq))$$

$$\Delta(E) = \uparrow(\{\Delta^b(e_i, e_j)\}_{e_i, e_j \in N(E)}, \leq))$$

are *mdbg* operators relative to N.

Dbg operators for lists

- An illustrative example



$$\uparrow(\{\text{Patterns},\leq\}) = V^{10}a^2V^{12}$$

Dbg operators for lists

Setting 2

Pattern language	Cost Function
$\mathcal{L}_1 = (L_0,+)$ $p = a_1a_2V_1V_2V_3 + V_1a_3$	$k_1\ (E,p) = c(p) + c'(E\|p)$

$$\Delta_N = \sum \Delta^b(e_i, e_j)$$

$$\Delta^{\sim}(E) = \uparrow(\Delta_N, \leq), \text{ where } \leq: \uparrow \text{ driven by } k_1$$

The *mdbg* is not always obtained via ↑

NP-Hard for a version of L_1

Index of Contents

- Introduction
- Motivation
- Distance-based generalisation (*dbg*) operators
- *Dbg* operators for lists
- Future work

Future work

- Including other similarity functions
  - Normalised distances ($0 \leq d \leq 1$)
  - Pseudo-distances (weighted edition distance, kernel functions, etc.)
- Making *dbg* operators more practical
  - Formalisation of the notion of weak *dbg* operator
  - Further results about composability of *dbg* operators
  - Overlapping control in cluster descriptions
- Exploring new pattern languages
  - Regular languages.
- Studying new cost functions
  - Improving the semantic cost function

Thanks for your attention!

Semantic cost functions

L_0 (single list pattern language)

$$C'(E|p) = \begin{cases} j\text{-}\mathit{max}\{\mathit{Length}(e)\}_{e \text{ in } E}, & \mathit{if}\ \ p = V^j \\ |E|, & \mathit{otherwise} \end{cases}$$

L_1 (multiple list pattern language)

$$C'(E|p) = \begin{cases} |E\text{-}E_1| + c(E_1|p_k),\ p_k = V^j\ \&\ E_1 = \{e \text{ in } E: \mathit{Length}(e) \leq j\} \\ |E|, \mathit{otherwise} \end{cases}$$

Distance-based generalisation operators

Common concepts in metric spaces

(closed ball)

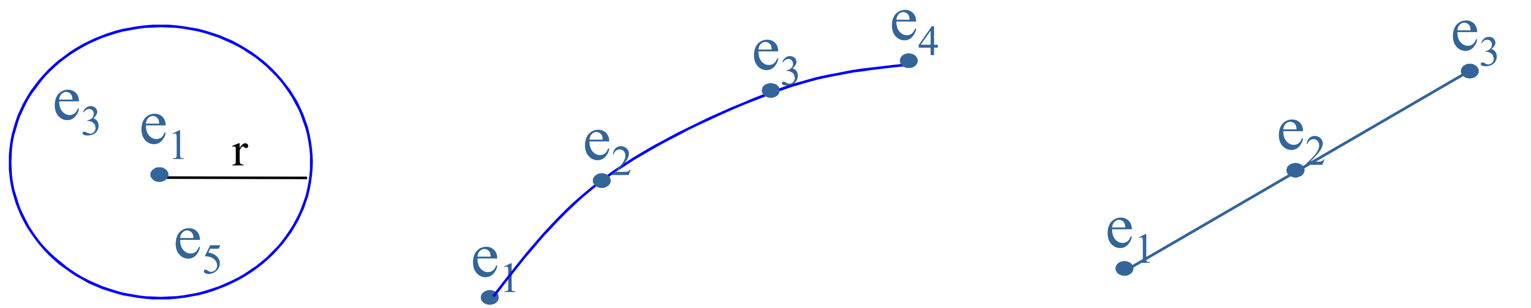
$B(e_1, r) = \{e_i \in X : d(e_1, e_i) \leq r\}$

(ε - path)

$\forall 1 \leq i \leq 3 : \mathrm{d}(\mathrm{e}_i, \mathrm{e}_{i+1}) \leq \varepsilon$

(intermediate element)

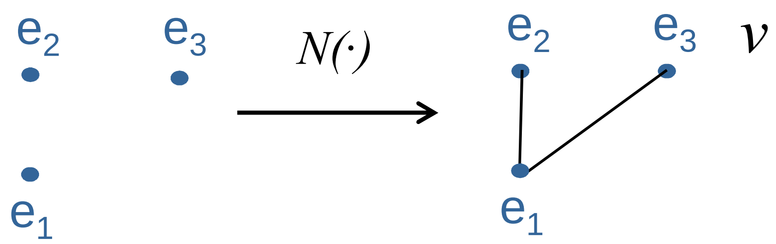
$d(e_1, e_2) + d(e_2, e_3) = d(e_1, e_3)$

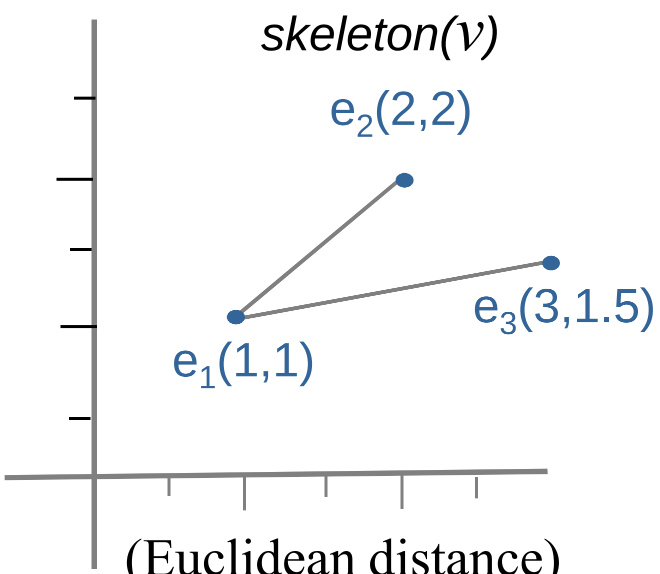


Distance-based generalisation operators

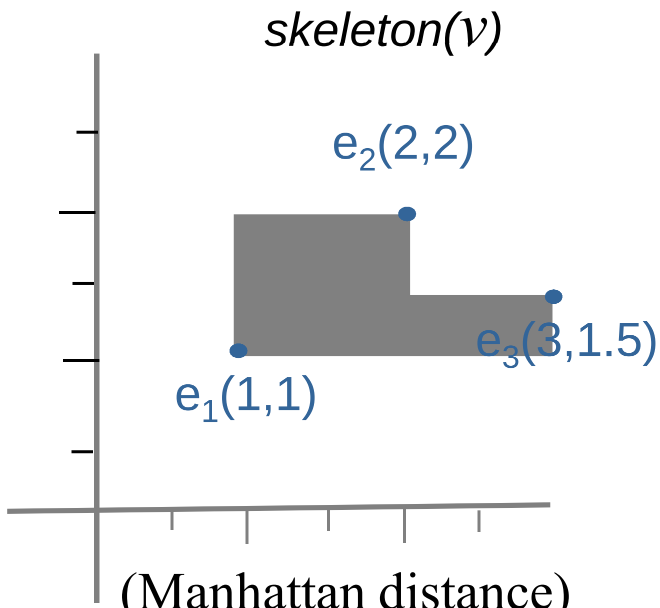
Moving to n-ary generalisations

- Given $N(\cdot)$,





(Euclidean distance)



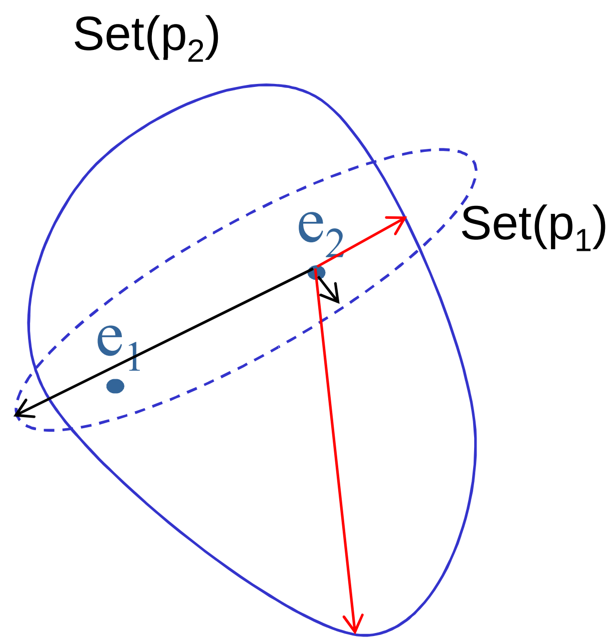
(Manhattan distance)

Minimal dbg operators

Limitations of inclusion ($\subset$)

- Distance function is ignored (many patterns become incomparable)

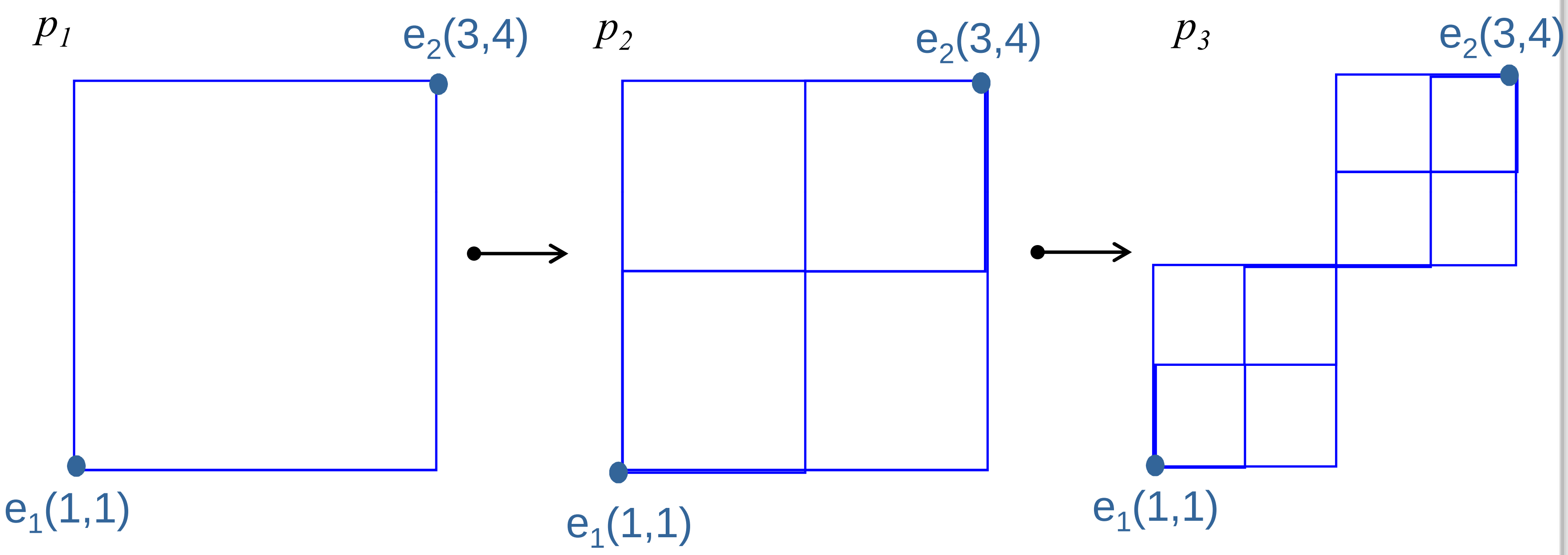
E.g.: Neither p_1 is more general than p_2 nor vice versa



Minimal dbg operators

Limitations of inclusion (⊂)

- The complexity of the pattern is ignored



Least general generalisation
might not exist!