NUEVAS PROPUESTAS PARA BÚSQUEDAS POR SIMILITUD EN BASES DE DATOS MÉTRICAS

Dra. Nora Reyes

Dra. Karina Figueroa

Verónica del Rosario Ludueña

Patricia Roggero

UNIVERSIDAD MICHOACANA
DE SAN NICOLÁS DE HIDALGO
Cuna de héroes, crisol de pensadores

Contenido del curso (1/2)

- Conceptos Fundamentales de Espacios Métricos
  - Introducción y motivación
  - Definición de espacios métricos.
  - Funciones de Distancia: propiedades.
  - Tipos de búsquedas pos similitud más comunes.
  - Maldición de la dimensión.
- Índices para Bases de Datos Métricas
  - Taxonomía de los índices
  - Principales referentes de índices basados en particiones compactas.
  - Principales referentes de índices basados en pivotes.
  - Principales referentes de índices basados en permutaciones
  - Índices estáticos y dinámicos. Ejemplos.
  - Índices para memoria secundaria. Ejemplos.
- Algoritmos Exactos y Aproximados
  - Algoritmos Exactos.
  - Algoritmos Aproximados.
  - Medidas de evaluación de calidad de respuesta.

Contenido del curso (2/2)

- Otras operaciones de Interés sobre Bases de Datos Métricas:
  - Join por Similitud, variantes.
  - Algoritmos para Join: con índices y sin índices.
  - Ejemplos de soluciones existentes.
  - Medidas de evaluación de la dimensionalidad.

Clasificación por Tipo de Respuesta

Exactos vs. Aproximados

- Además de clasificar los algoritmos por el enfoque que utilizan (pivotes o particiones), se pueden clasificar por el tipo de respuesta a la búsqueda por similitud que permiten obtener.
- Por ejemplo: en una consulta (q, r)
  - Un algoritmo ***exacto*** obtiene todos los $x \in U$, $d(q,x) \leq r$.
  - Un algoritmo ***aproximado*** obtiene algunos $x \in U$, $d(q,x) \leq r$.

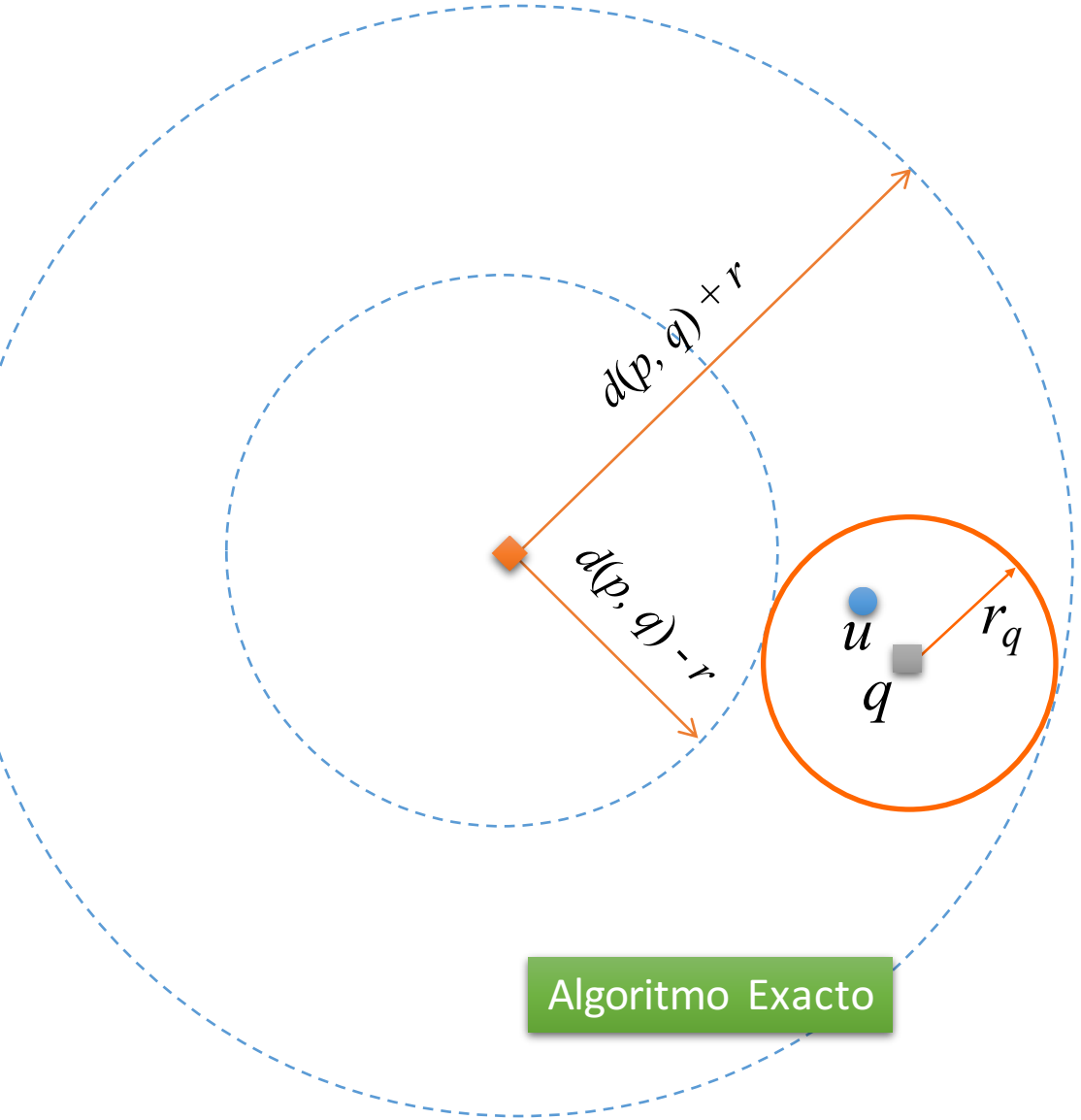
Búsqueda por Similitud Aproximada

- Estrategias de aproximación
  - **Condiciones de poda relajadas**
    - Regiones de datos que se superponen con la bola de consulta se pueden descartar dependiendo de la estrategia específica.
  - **Terminación temprana de la búsqueda**
    - El algoritmo de búsqueda podría parar antes de que todas las regiones sean accedidas.

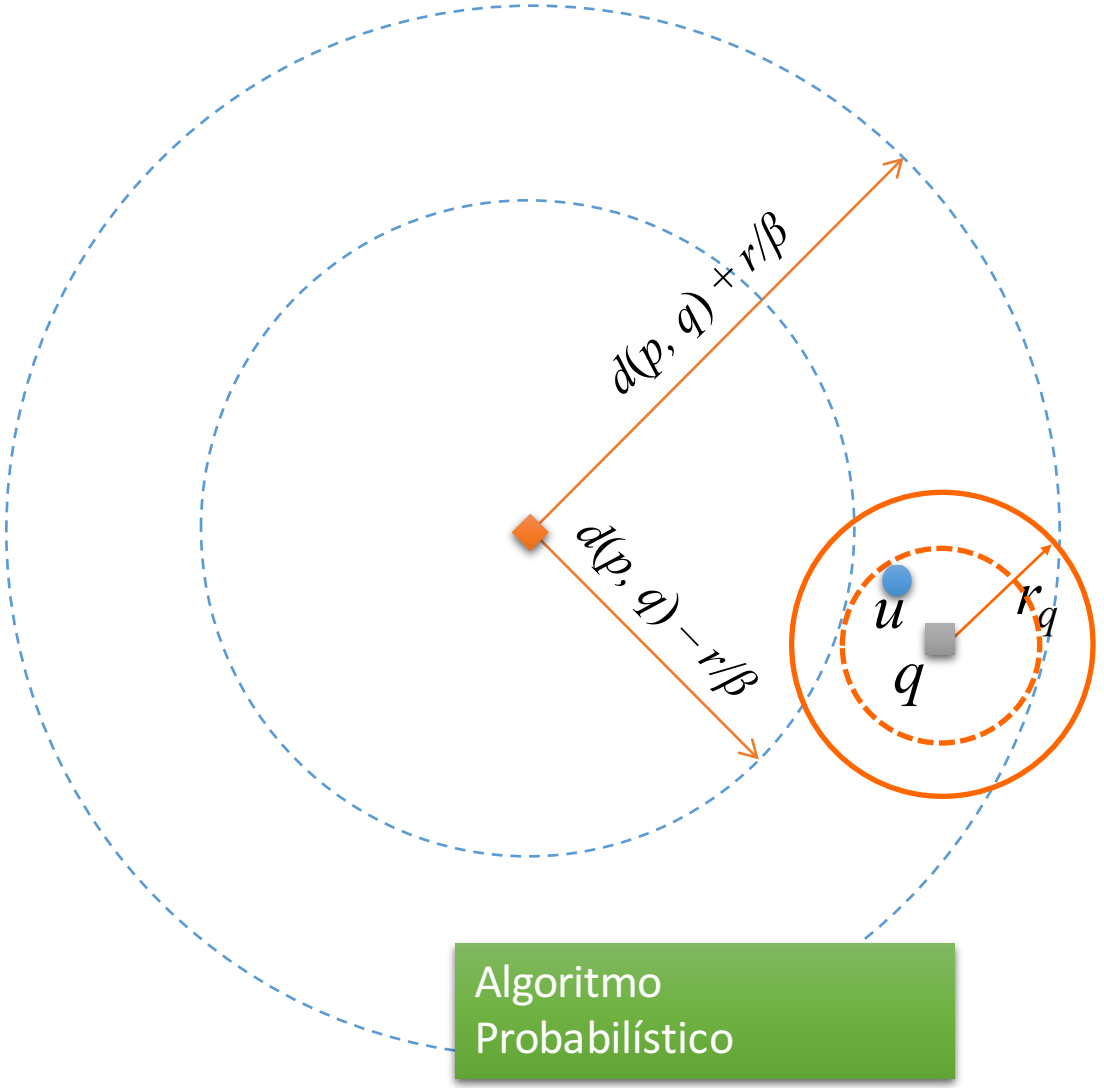
Aproximados y Probabilísticos

- Aproximados:
  - El usuario define un valor $0 < \rho \leq 1$
    - indica qué tan relajada acepta una solución respecto a la respuesta correcta.
  - Muchos algoritmos exactos usan esta estrategia para conseguir mejorar su desempeño a cambio de perder precisión.
- Probabílisticos:
  - Una variante de este tipo de algoritmos son los basados en ordenamientos.
  - En este tipo de heurísticas se propone un *orden (más prometedor primero) en que deben ser comparados los elementos*
    - *La idea es identificar rápidamente los elementos prometedores y a medida que se degrade la calidad de la respuesta dejar de comparar elementos en la base de datos.*

Exactos y aproximados



Algoritmo Exacto



Algoritmo Probabilístico

Principios de la Búsqueda

- Los algoritmos aproximados se pueden clasificar en dos categorías distintas:
  - los que explotan la transformación del espacio métrico:

    se consiguen cambiando la representación de los objetos y/o la función de distancia con el objetivo de reducir el costo de la búsqueda.

  - los que reducen el subconjunto de los datos que se examinan:

    se omiten partes del conjunto de datos que probablemente no contengan objetos relevantes para la consulta.

Principios de la Búsqueda

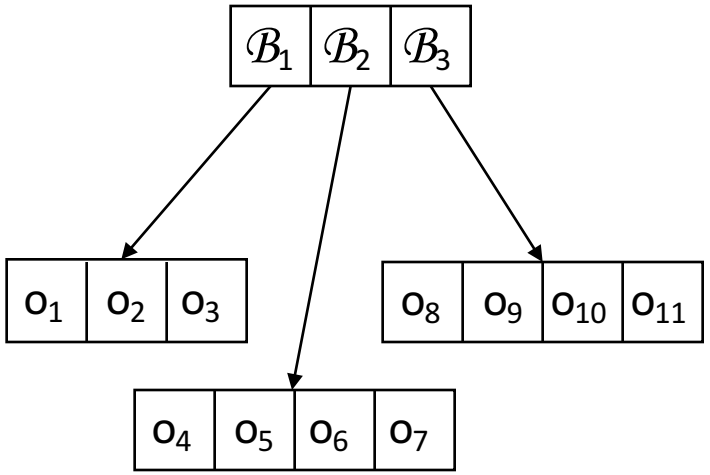
- Transformación del espacio:
  - Se usan transformaciones que preservan distancias: las distancias en el espacio transformado son más pequeñas que en el espacio original, lo que puede ocasionar *falsos aciertos*.
    - Ejemplo: técnicas de reducción de dimensionalidad.
- Reducir los subconjuntos de datos a examinar:
  - No se acceden los datos no prometedores, lo que puede ocasionar *falsas exclusiones*.

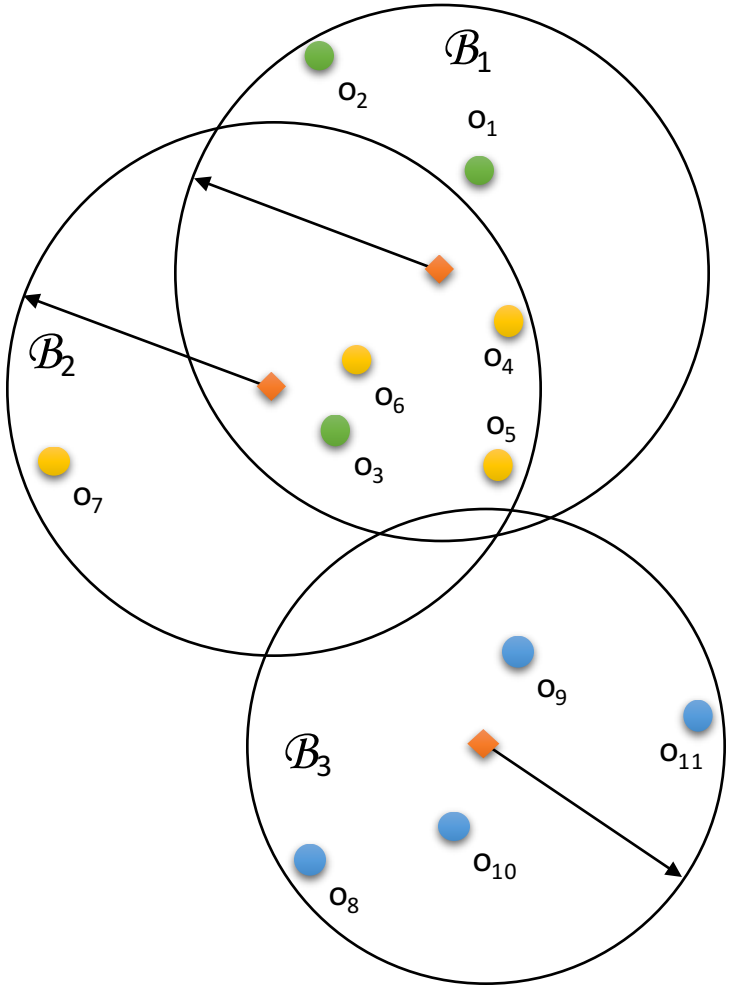
Estrategias de Ramificación Relajada

- En búsquedas exactas: se acceden todas las regiones que se solapan con la región de la consulta y se descartan todas las demás.
- En búsquedas aproximadas: se usan condiciones de poda “relajadas”, rechazando las regiones que se solapan con la consulta cuando se detecta que hay baja probabilidad de que haya objetos relevantes en la intersección.
- Estas estrategias en particular son útiles y efectivas con índices basados en una descomposición jerárquica del espacio.

Búsqueda Aproximada: Ejemplo

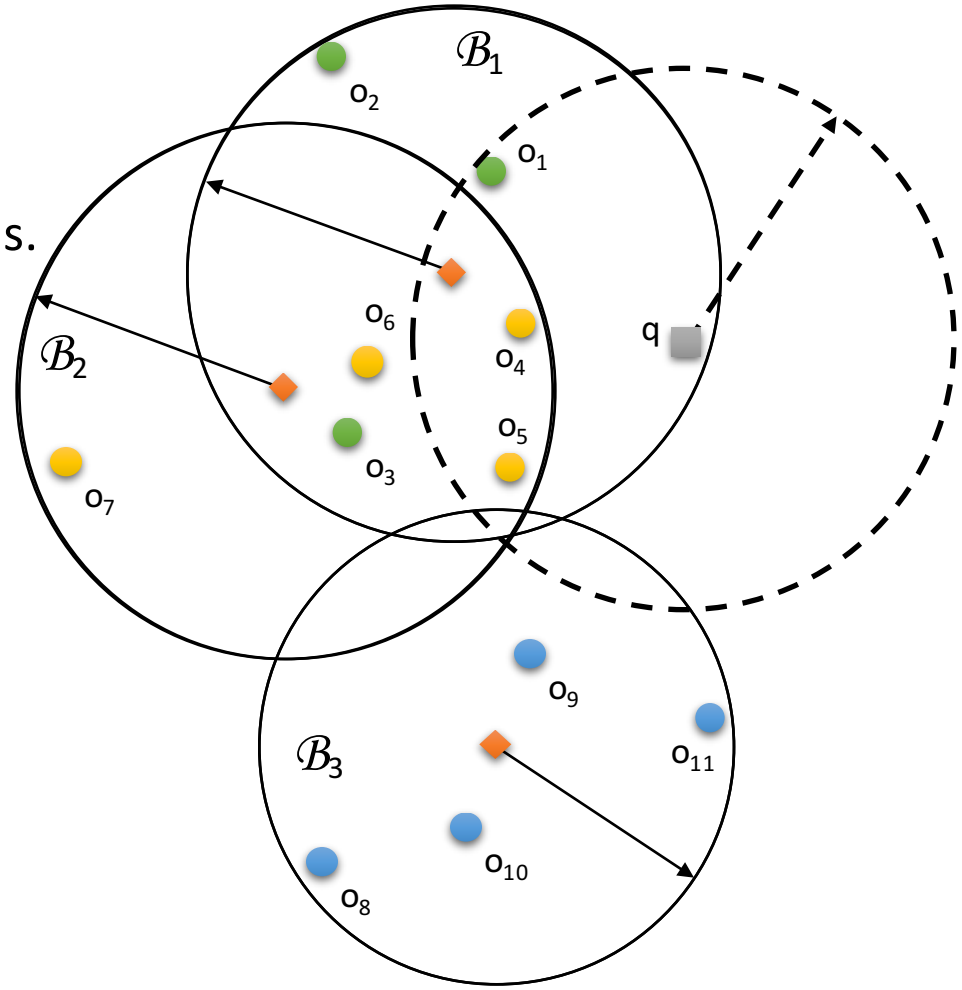
- Un índice hipotético con tres regiones:





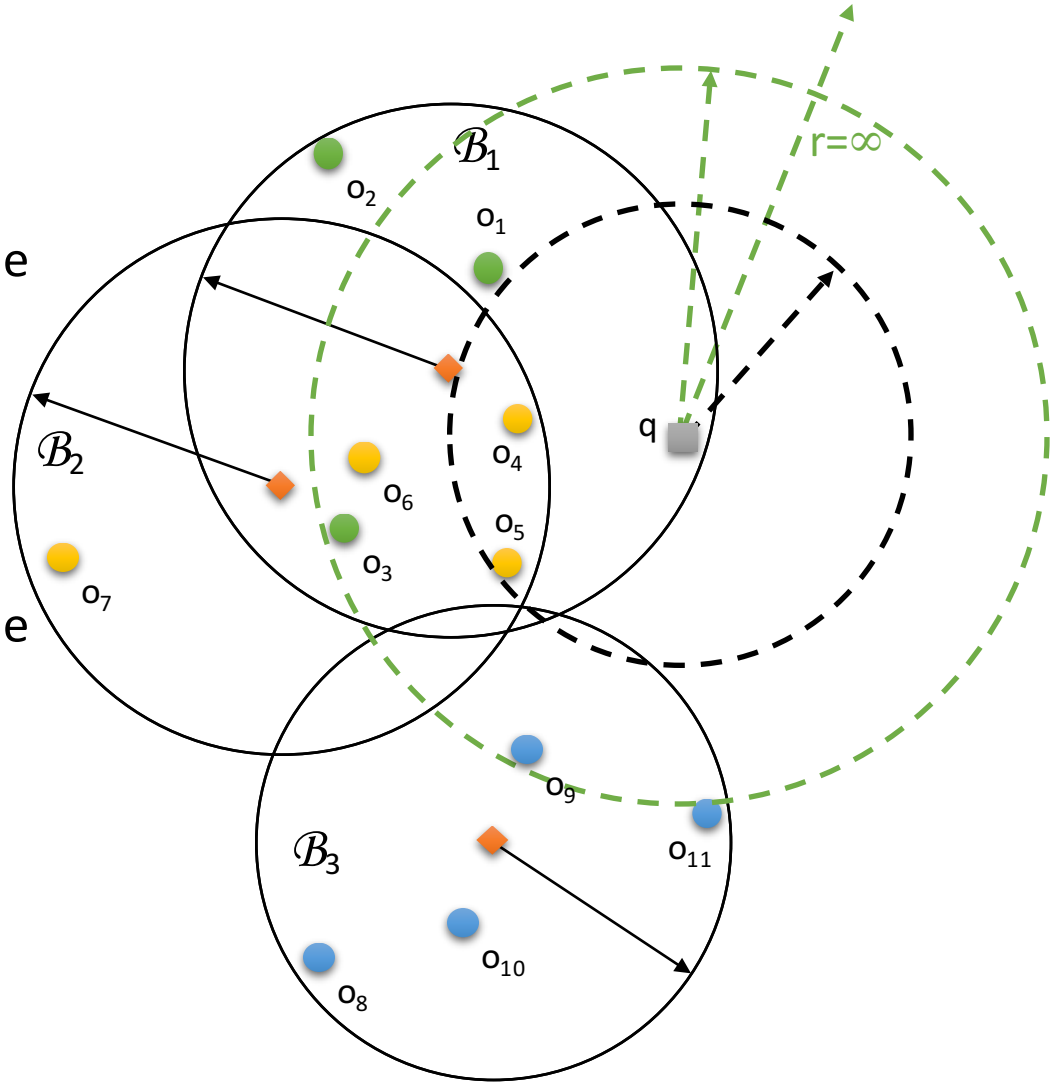
Búsqueda por Rango Aproximada

- Dada una consulta (q, r):
- Acceder $\mathcal{B}_1$
  - Reportar o_1
  - Si la terminación temprana se detiene, se podrían perder objetos.
- Acceder $\mathcal{B}_2$
  - Reportar o_4,o_5
  - Si la terminación temprana se detiene, no se perdería nada.
- Acceder $\mathcal{B}_3$
  - Nada para reportar
  - Una estrategia de ramificación relajada puede descartar esta región – no se perdería nada.



Búsqueda *2-NN* Aproximada

- Sea una consulta 2-NN(q) :
- Acceder $\mathcal{B}_1$
  - Vecinos: o_1 ,o_3
  - Si la terminación temprana detiene ahora, se podrían perder elementos.
- Acceder $\mathcal{B}_2$
  - Vecinos: o_4 ,o_5
  - Si la terminación temprana detiene ahora, no se perdería nada.
- Acceder $\mathcal{B}_3$
  - Vecinos: o_4 ,o_5 – no cambian
  - Una estrategia de ramificación relajada puede descartar esta región – no se pierde nada.



Medidas de Desempeño

- Para evaluar el desempeño de las búsquedas por similitud aproximada se debería considerar:
  - Mejoras en la eficiencia
  - Precisión de los resultados aproximados
- Existe un compromiso entre los dos: *grandes mejoras en la eficiencia se obtienen al costo de precisión en los resultados*.
- Los buenos algoritmos de aproximación deberían ofrecer *grandes mejoras en la eficiencia con alta precisión en los resultados*.

Mejoras en la eficiencia

- La mejora en la eficiencia (ME) se expresa como:

$$ME = \frac{Costo(Q)}{Costo^{Aprox}(Q)}$$

  - *Costo* y $Costo^{Aprox}$: número de accesos a disco o de evaluaciones de distancia para ejecutar la consulta Q de manera exacta y aproximada respectivamente.
- Q puede ser una consulta por rango o de *k-NN*.
- Ejemplo:
  - Ejecución de búsqueda exacta: 6 minutos
  - Ejecución de búsqueda aproximada: 36 segundos
  - *ME* = 10

La búsqueda aproximada es 10 veces más rápida.

Medidas de Desempeño

- Las medidas de Precisión y Recuperación (Precision y Recall) se usan en recuperación de la información para evaluar el desempeño.

- ***Precisión***: razón entre los objetos recuperados que califican y el total de objetos recuperados.

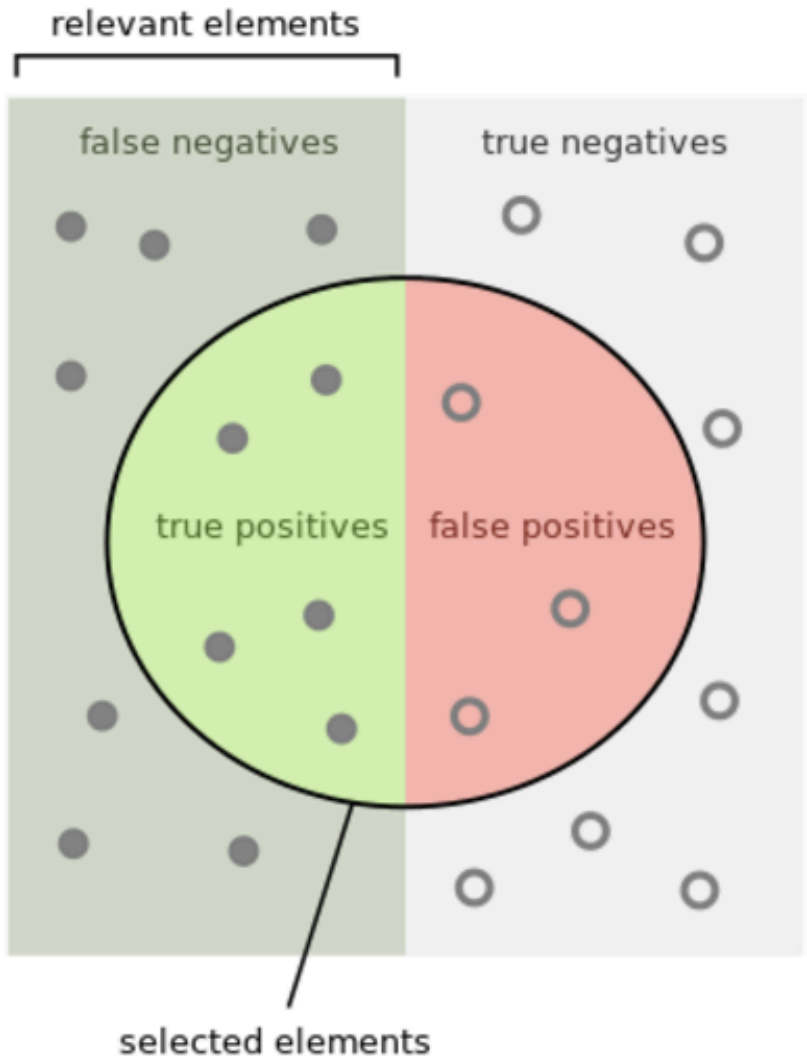
- ***Recuperación***: razón entre los objetos recuperados que califican y el número total de objetos que califican.

Precisión y Recuperación

- La exactitud se puede cuantificar con *Precisión (P) y Recuperación (R)*:

$$P = \frac{\left|S \cap S^A\right|}{\left|S^A\right|}, \qquad R = \frac{\left|S \cap S^A\right|}{\left|S\right|}$$

  - *S* – objetos que califican; es decir, objetos recuperados por el algoritmo exacto
  - S^A – *objetos realmente recuperados; es decir, objetos recuperados por el algoritmo aproximado*



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

Precisión y Recuperación

- Estas medidas son muy intuitivas, pero en este contexto su interpretación no es obvia y es engañosa.
- Para la *búsqueda por rango* se tiene que $S^A \subseteq S$ y por lo tanto la Precisión es siempre 1 en este caso.
- Los resultados de *k-NN(q)* tienen siempre tamaño *k* y por lo tanto, Precisión es siempre igual a la Recuperación en este caso.
- Cada elemento tiene la misma importancia: *perder el primer objeto en lugar del objeto 1000 es lo mismo.*

Precisión y Recuperación

- Supongamos la siguiente consulta *10-NN*(*q*):
  - $S = \{1,2,3,4,5,6,7,8,9,10\}$
  - $S^{A1} = \{2,3,4,5,6,7,8,9,10,11\}$
  - $S^{A2} = \{1,2,3,4,5,6,7,8,9,11\}$

se pierde el objeto 1

se pierde el objeto 10

- En ambos casos: $P = R = 0.9$
  - Sin embargo, S^{A2} se puede considerar mejor que S^{A1}.

Precisión y Recuperación

- Supongamos una consulta *1-NN*(*q*):
  - $S=\{1\}$
  - $S^{A1}=\{2\}$
  - $S^{A2}=\{10000\}$

sólo se salta un objeto

se saltaron los primeros 9.999 objetos

- En ambos casos: $P = R = 0$
  - Sin embargo, se puede considerar a S^{A1} mucho mejor que S^{A2}.

Error Relativo en Distancias

- Otra posibilidad para evaluar la exactitud es el error relativo en las distancias (*ED*).
  - ED compara las distancias desde un objeto de consulta a los objetos en los resultados exactos y aproximados:

$$ED = \frac{d(o^A,q) - d(o^N,q)}{d(o^N,q)} = \frac{d(o^A,q)}{d(o^N,q)} - 1$$

donde o^A y o^N son el vecino más cercano aproximado y real, respectivamente.

- La generalización al caso del *j*-ésimo-*NN*:

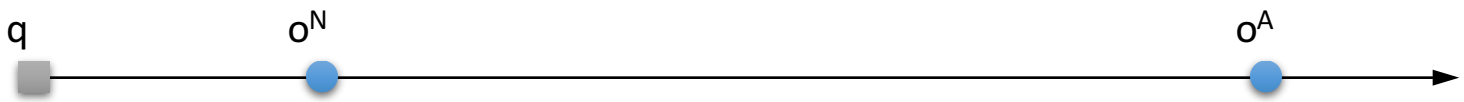
$$ED_j = \frac{d(o_j^A,q)}{d(o_j^N,q)} - 1$$

Error Relativo en Distancias

- La desventaja es que no considera la distribución de las distancias.

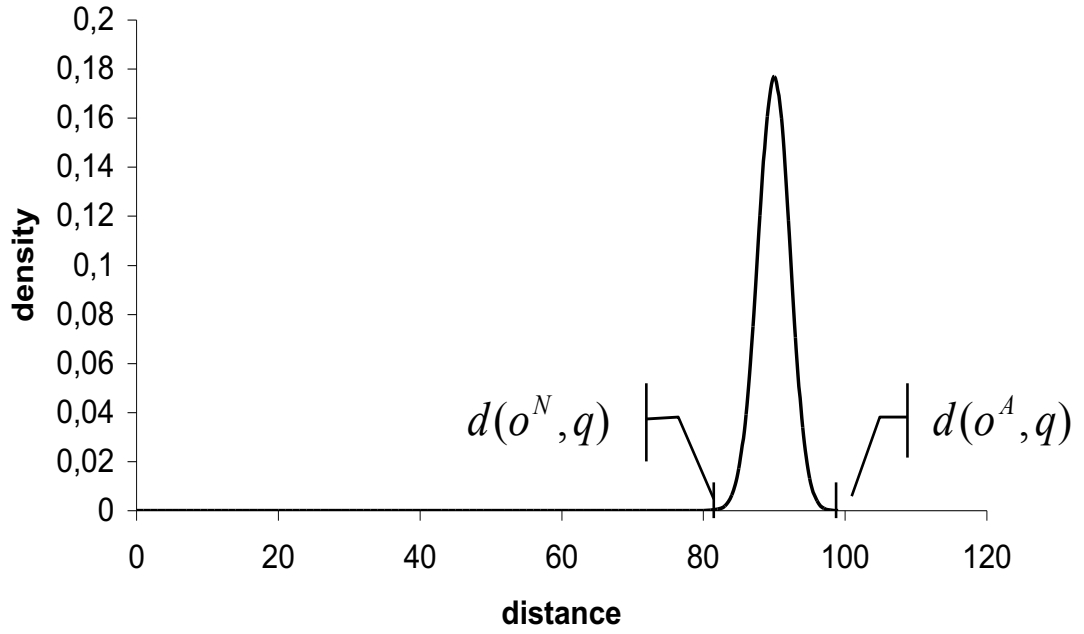
- Ejemplo1: La diferencia en distancia desde el objeto de consulta *q* a o^N y a o^A es amplia (comparada al rango de distancias)

  - Si el algoritmo pierde a o^N y toma a o^A, *ED* es amplia aún si sólo un objeto se ha perdido.



Error Relativo en Distancias

- Ejemplo 2: Casi todos los objetos tienen la misma (gran) distancia desde el objeto de consulta *q*.
  - Eligiendo el más lejano más que el más cercano podría producir un *ED* pequeño, aún si casi todos los objetos se han perdido.



Error de Posición o exactitud

- La exactitud también se puede medir como el Error de Posición (*EP)*; es decir, la discrepancia entre los rankings en los resultados aproximados y exactos.
- *EP se puede obtener usando la distancia Sperman Footrule (SFD):*

$$SFD = \sum_{i=1}^{|X|} \left| S_1(o_i) - S_2(o_i) \right|$$

siendo $S_i(o)$ la posición del objeto o en la lista ordenada S_i y $|X|$ la cardinalidad del conjunto de objetos.

Error de Posición

- La posición en el resultado aproximado es siempre menor o igual a una del resultado exacto
  - S^A *es* una sublista de *OX*.
  - $S^A(o) \leq OX(o)$.
  - *OX La lista con la bd ordenada respecto a q*
- Se puede usar también un factor de normalización $|S^A| \cdot |X|$.
- El *error de posición (EP)* se define como:

$$EP = \frac{\sum_{i=1}^{|S^A|} \left(OX(o_i) - S^A(o_i) \right)}{|S^A| \cdot |X|}$$

Error de Posición

- Supongamos |X|=10.000
- Consideremos una consulta 10-*NN*(*q*):
  - S = {*1,2,3,4,5,6,7,8,9,10*}
  - S^{A1}= {*2,3,4,5,6,7,8,9,10,11*}
  - S^{A2}= {*1,2,3,4,5,6,7,8,9,11*}
- Como sugiere la intuición:
  - Para S^{A1}, EP = 10 / (10 · 10.000) = 0.0001
  - Para S^{A2}, EP = 1 / (10 · 10.000) = 0.00001

se pierde el objeto 1

se pierde el objeto 10

Error de Posición

- Supongamos que |X|=10.000
- Consideremos una consulta *1-NN*(*q*):
  - $S = \{1\}$
  - $S^{A1} = \{2\}$
  - $S^{A2} = \{10.000\}$

sólo se salta un objeto

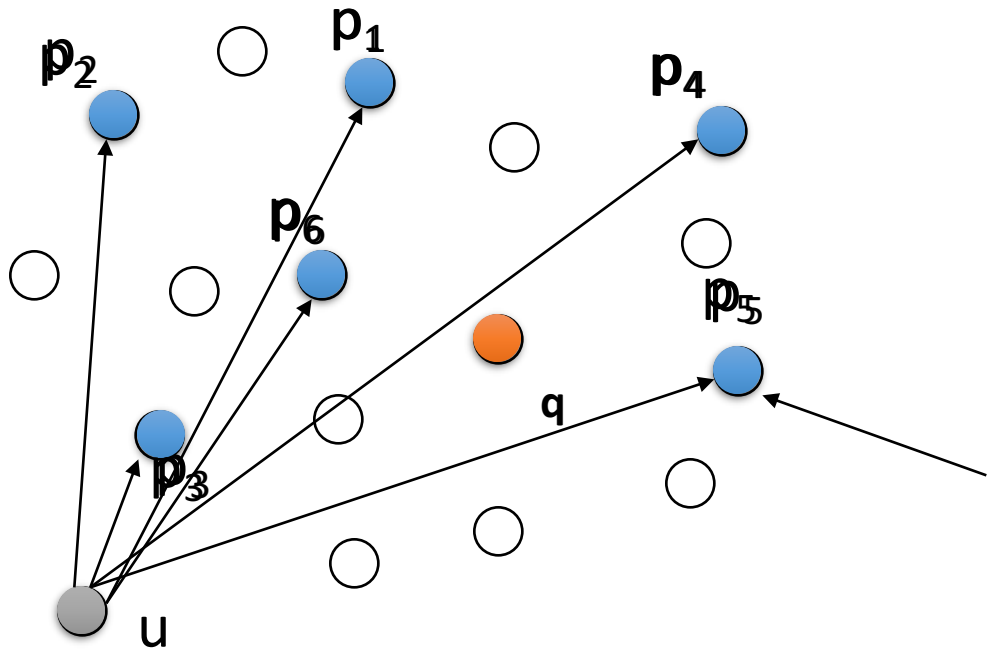
se saltaron los primeros 9.999 objetos

- Como también sugiere la intuición:
  - Para S^{A1}, $EP = (2-1)/(1 \cdot 10.000) = 1/(10.000) = 0.0001$
  - Para S^{A2}, $EP = (10.000-1)/(1 \cdot 10{,}000) = 0.9999$

Algoritmos Aproximados

Índice de Permutaciones

IDEA



$$\Pi_u = p_3, p_6, p_2, p_1, p_5, p_4$$
$$\Pi_q = p_6, p_5, p_4, p_1, p_3, p_2$$

Permutaciones

- Elementos iguales tienen la misma permutación.
- ***Elementos similares*** deberían tener ***permutaciones similares.***
- Similitud entre permutaciones
  - Spearman Footrule *O(k)*
  - Spearman Rho *O(k)*
  - Kendall Tau *O(k^2)*

$$F(\Pi_q, \Pi_u) = \sum_{1 \le i \le k} |\Pi_u^{-1}(i) - \Pi_q^{-1}(i)|$$

$$S_\rho(\Pi_q, \Pi_u) = \sum_{1 \le i \le k} |\Pi_u^{-1}(i) - \Pi_q^{-1}(i)|^2$$

Similitud entre Permutaciones

$$\Pi_q = p_1, p_2, p_3, p_4, p_5, p_6$$

$$\Pi_u = p_3, p_6, p_2, p_1, p_5, p_4$$

1-3, 2 - 6, 3-2, 4-1, 5-5, 6-4

Diferencia de posiciones

Spearman Footrule

$$F(\Pi_q, \Pi_u) = |1-3| + |2-6| + |3-2| + |4-1| + |5-5| + |6-4| = 12$$

Spearman Rho

$$S_p(\Pi_q, \Pi_u) = |1-3|^2 + |2-6|^2 + |3-2|^2 + |4-1|^2 + |5-5|^2 + |6-4|^2 = 34$$

Índice de Permutaciones

- El índice almacena las permutaciones de todos los objetos de *X* respecto de un conjunto P de permutantes.

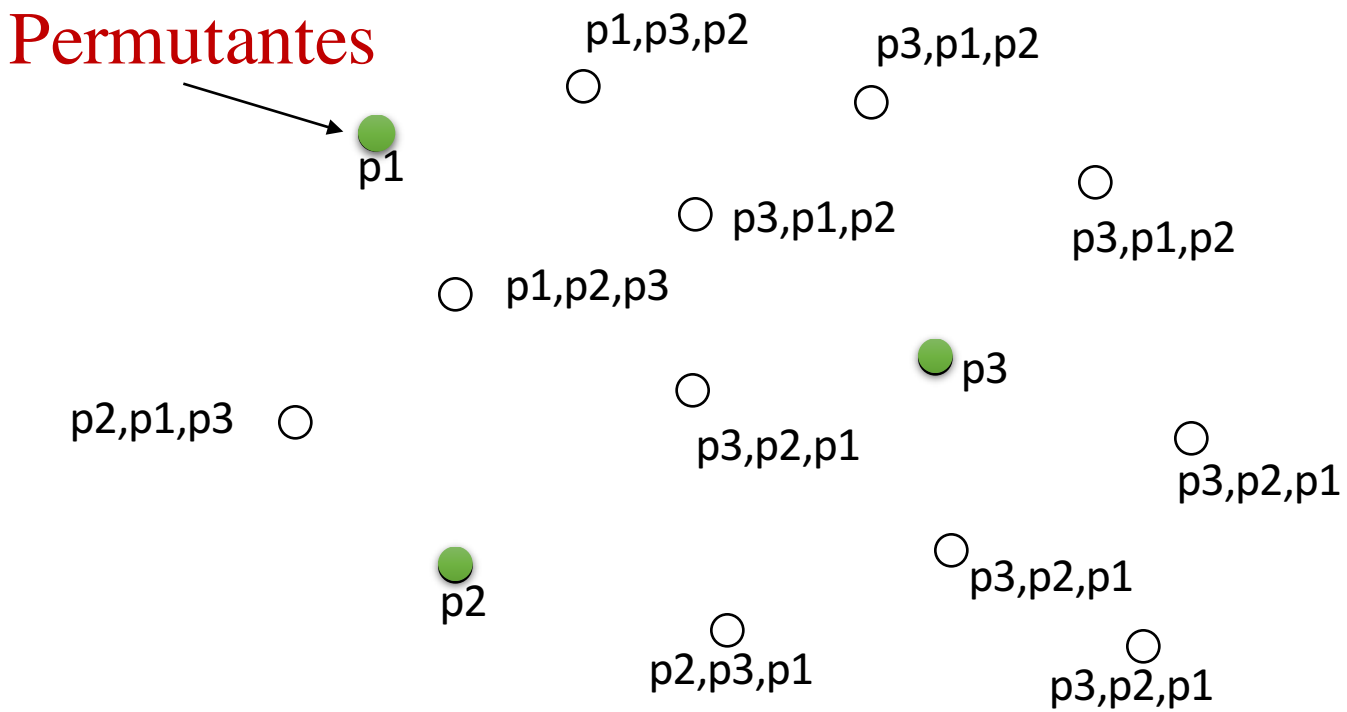
- Ante una consulta (*q*, *r*), se obtiene la permutación de *q* y se revisa una fracción *f* de *U* de los objetos con permutaciones más similares a las de *q*.

- Entonces, sólo se calculan *f* distancias a objetos de *U.*

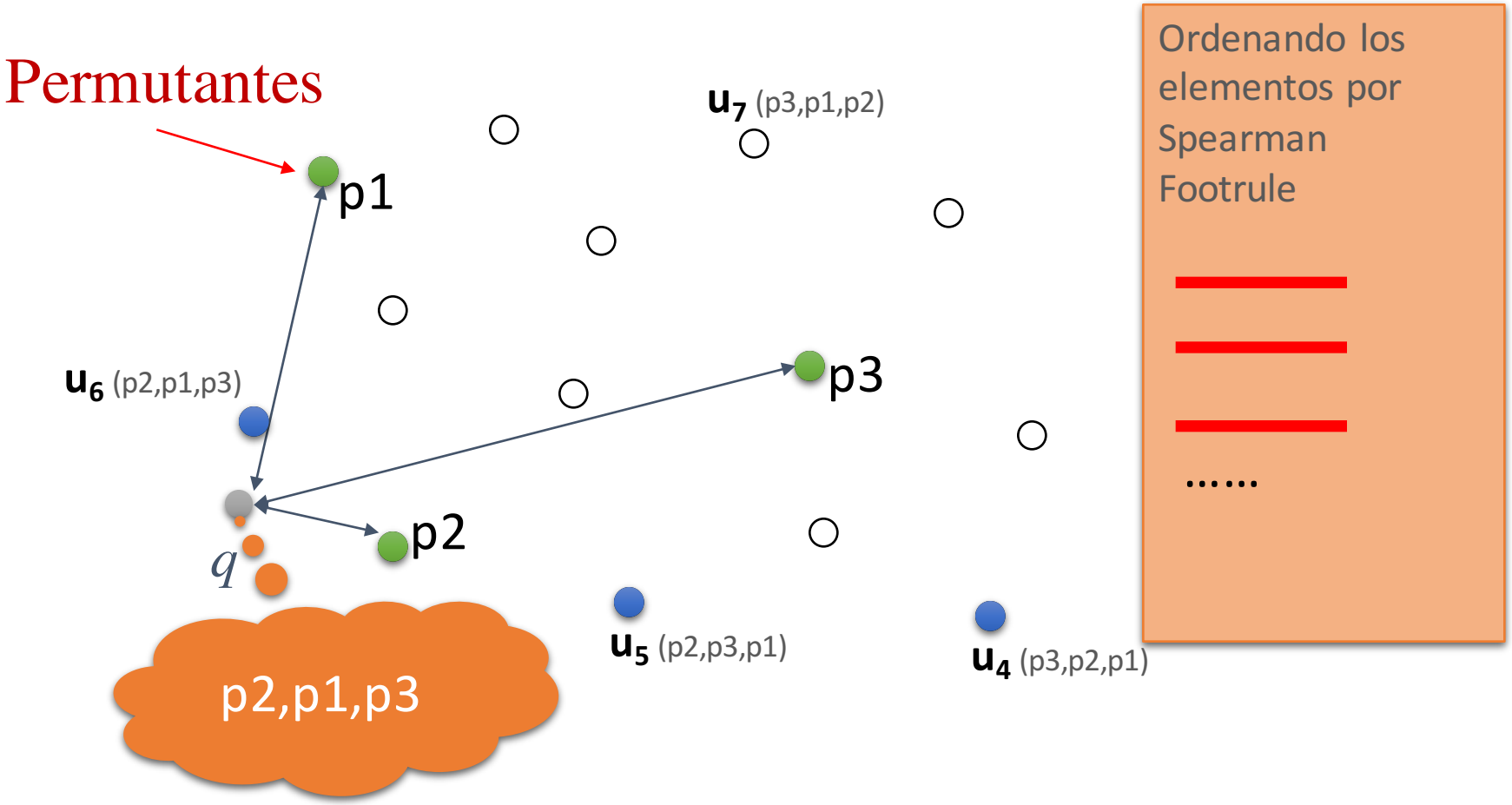
Índice de Permutaciones

- Sin embargo, se calculan *n* distancias entre permutaciones.
- Si aumenta *f* aumenta la calidad de la respuesta y el costo de la búsqueda en evaluaciones de distancia.
- Si aumenta el número de permutantes ¿aumenta la calidad de la respuestas? ¿qué costos aumentan?

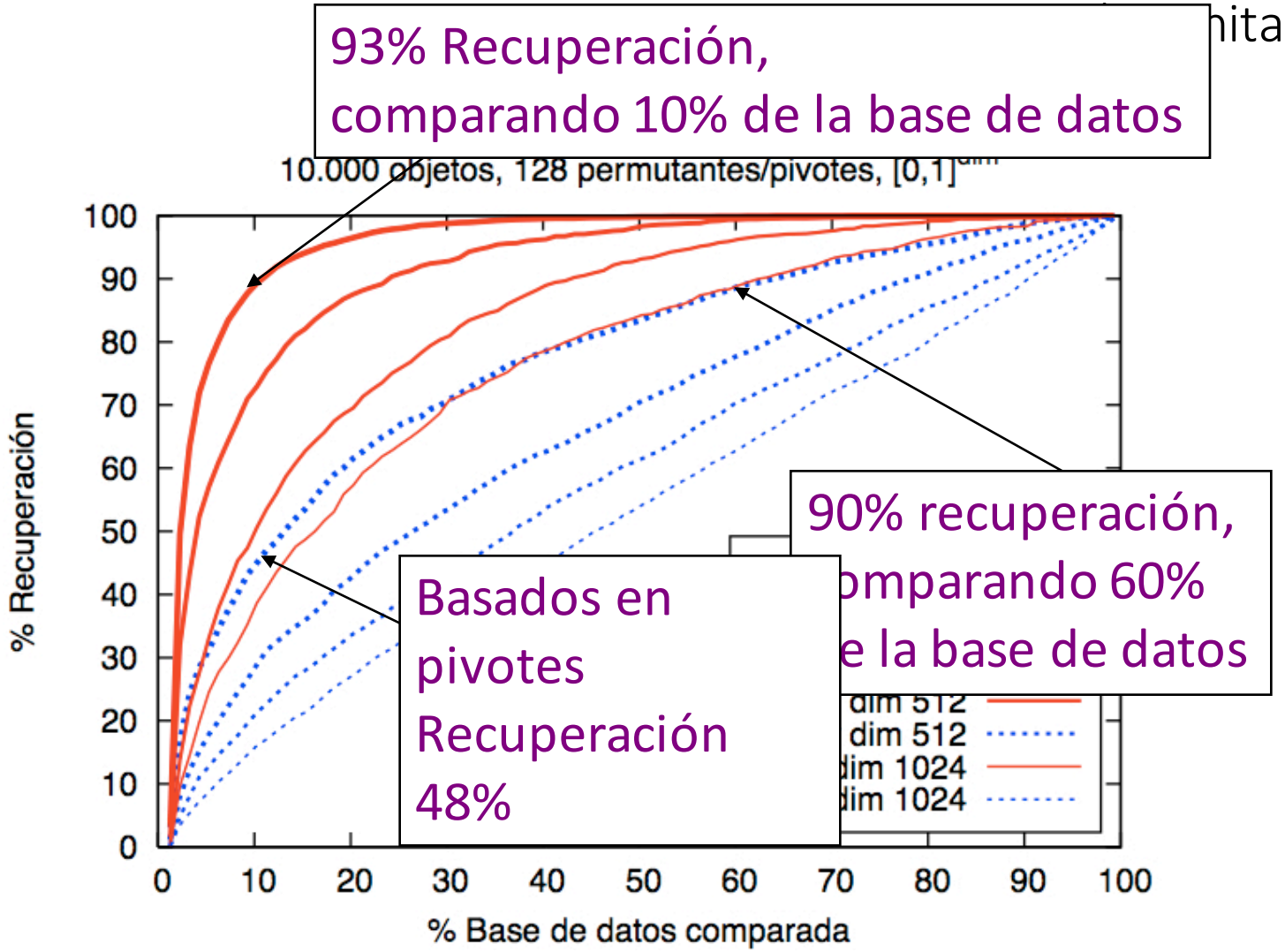
Preprocesamiento



Búsqueda


Permutantes
p1
u_7 (p3,p1,p2)
u_6 (p2,p1,p3)
p3
p2
q
p2,p1,p3
u_5 (p2,p3,p1)
u_4 (p3,p2,p1)
Ordenando los elementos por Spearman Footrule
.....

Resultados



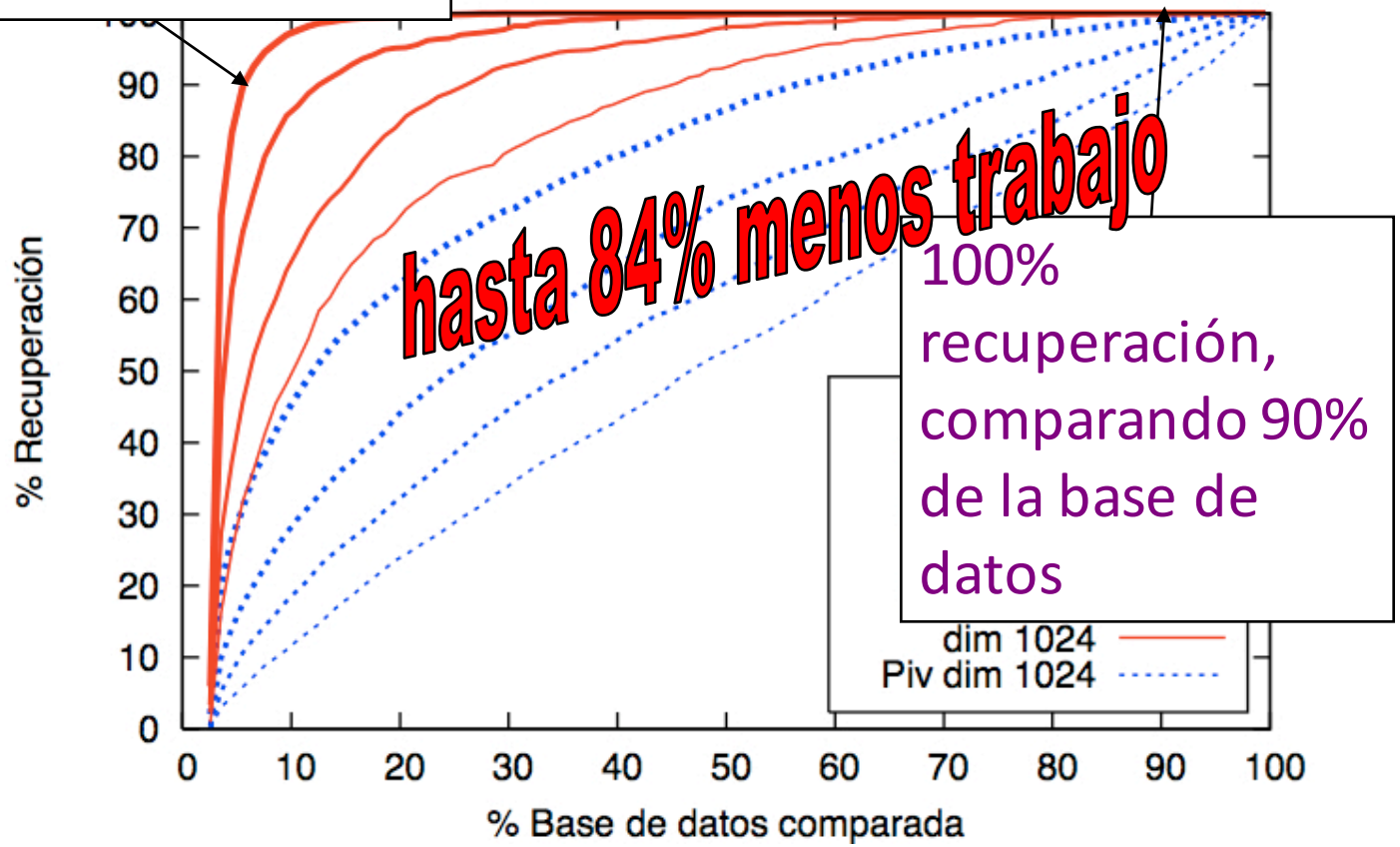
Probabilístico

Resultados

Cubo unitario

90% recuperación, comparando 5% de la base de datos

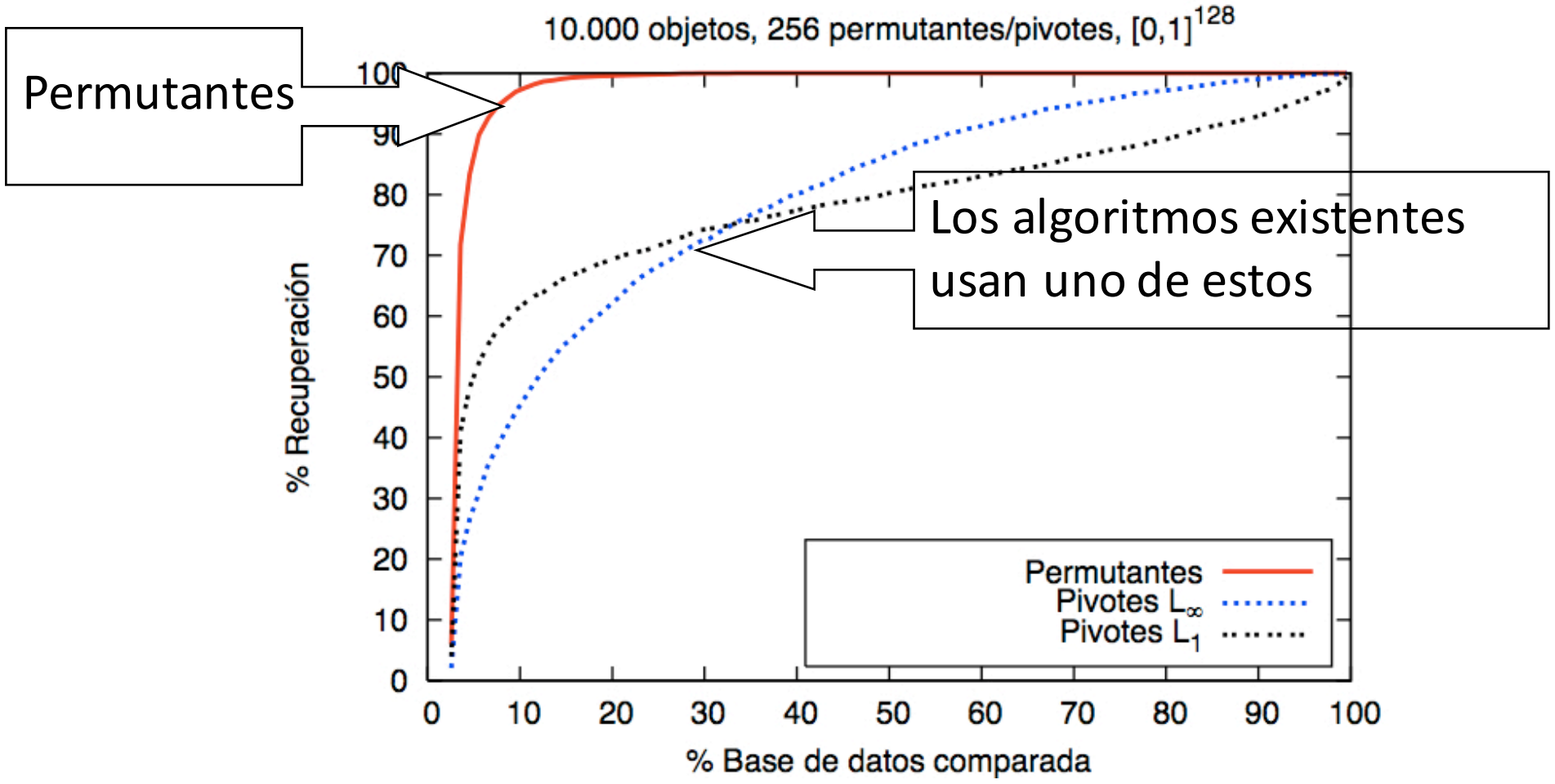
100% Recuperación, Comparando 15% de la base de datos



100% recuperación, comparando 90% de la base de datos

Probabilístico

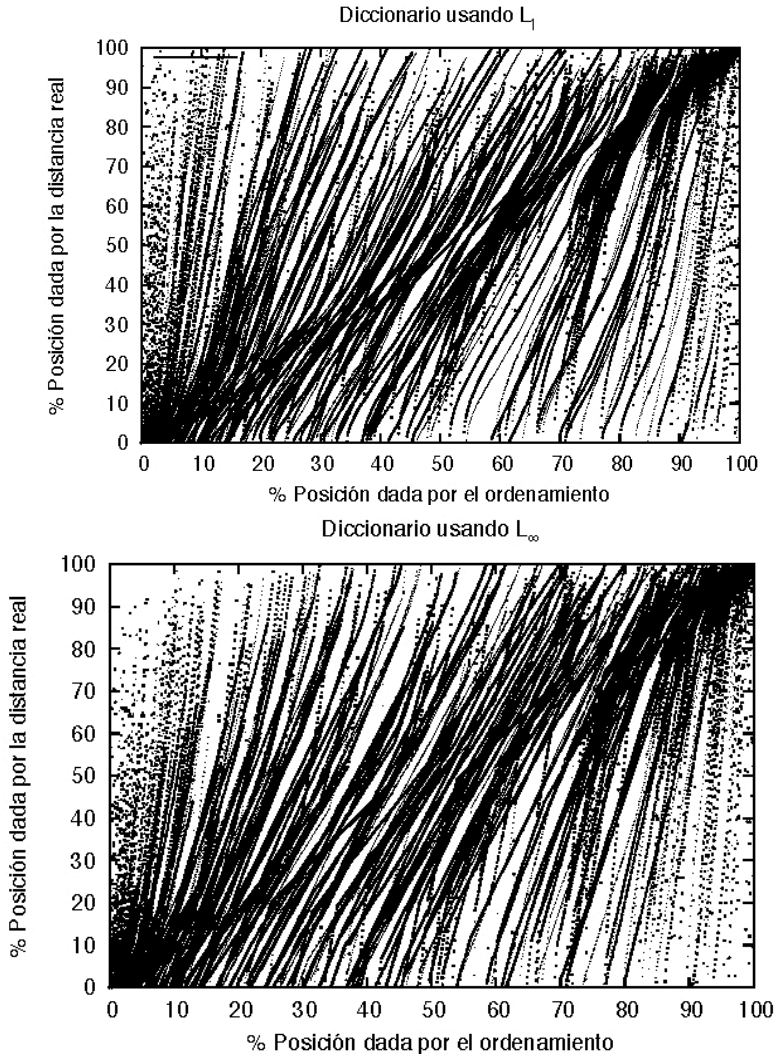
Nuestro Predictor

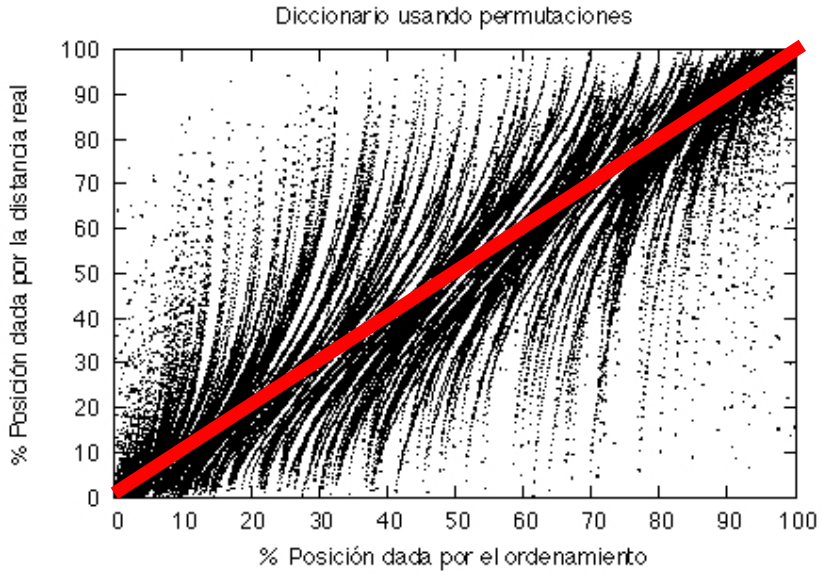


Probabilístico

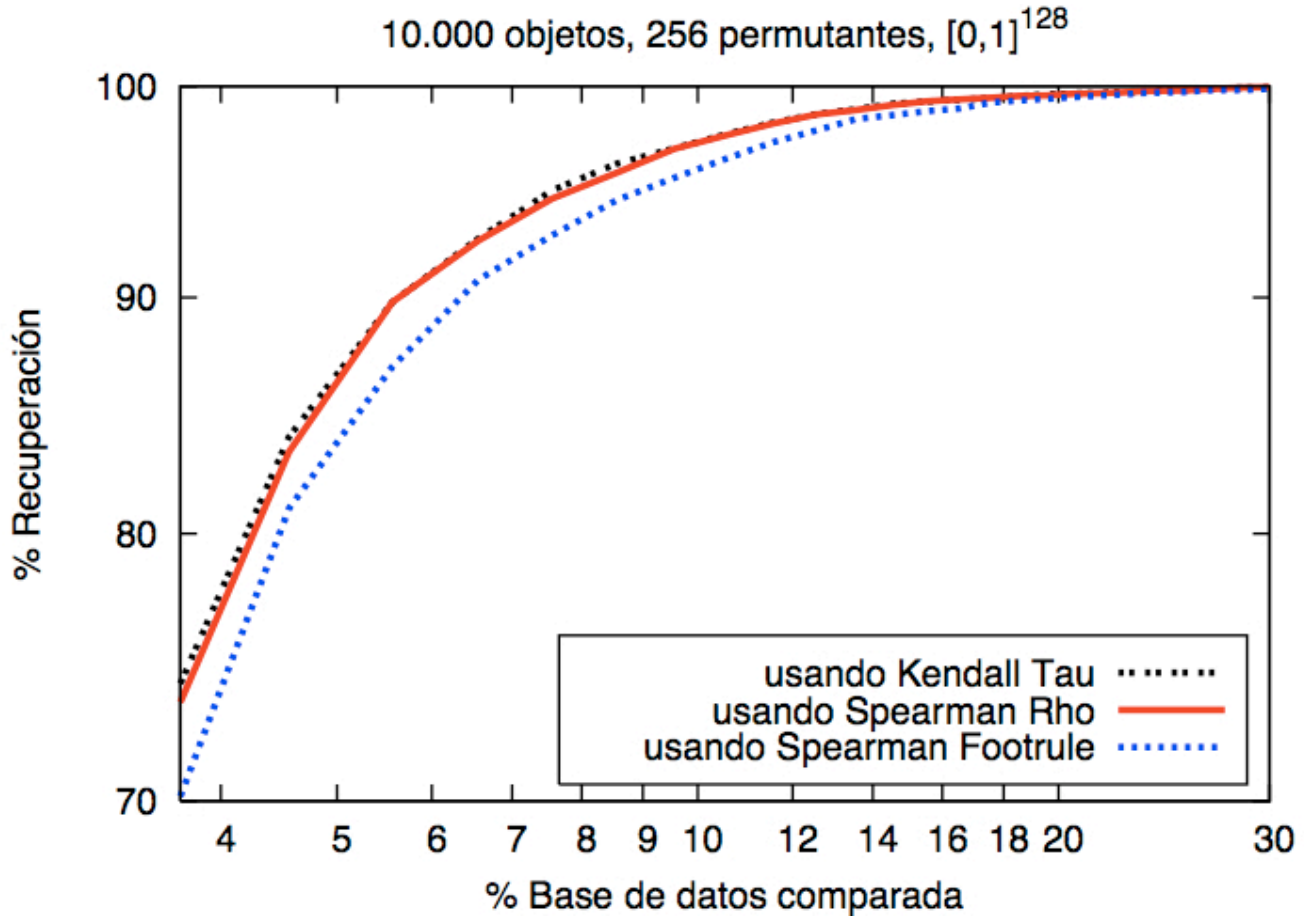
Predictores

Diccionario





Similaridades entre permutaciones



Probabilístico

iAESA

Figueroa K, Chávez E., Navarro G. Paredes R. On the least cost for proximity searching in metric spaces. In WEA 2006, LNCS. Pages 279-290

iAESA

Un estimador distinto

- *Costo empírico O(1)* cálculos de distancia
- $O(n^2)$ distancias precalculadas
- *O(n)* pivotes
- Nuevo estimador de proximidad

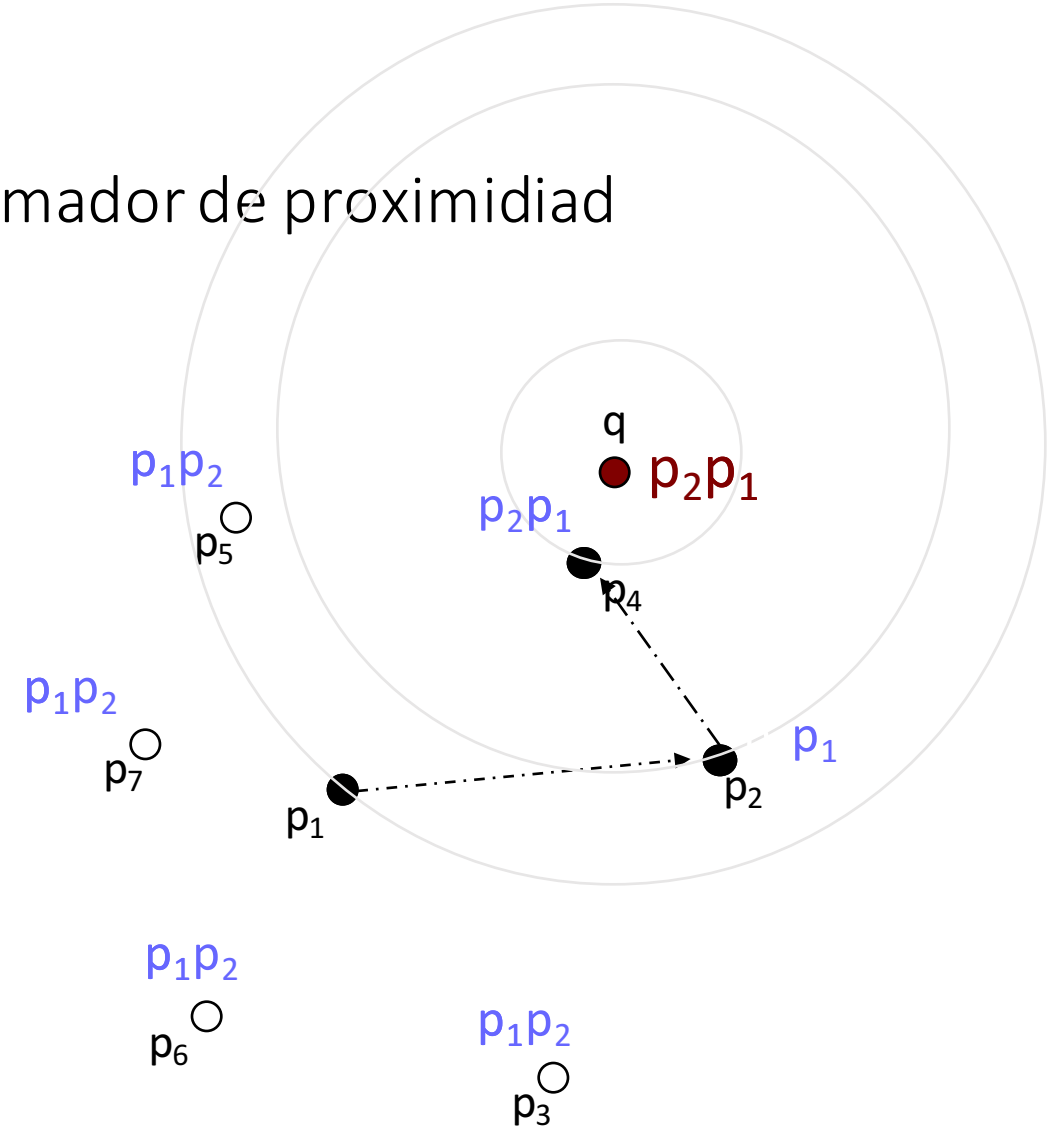
$$S_p(u) = \sum_{i=1}^{|P|} |\Pi_u^{-1}(p_i) - \Pi_q^{-1}(p_i)|^2$$

Exacto

iAESA, Ejemplo

Permutaciones como un estimador de proximidiad

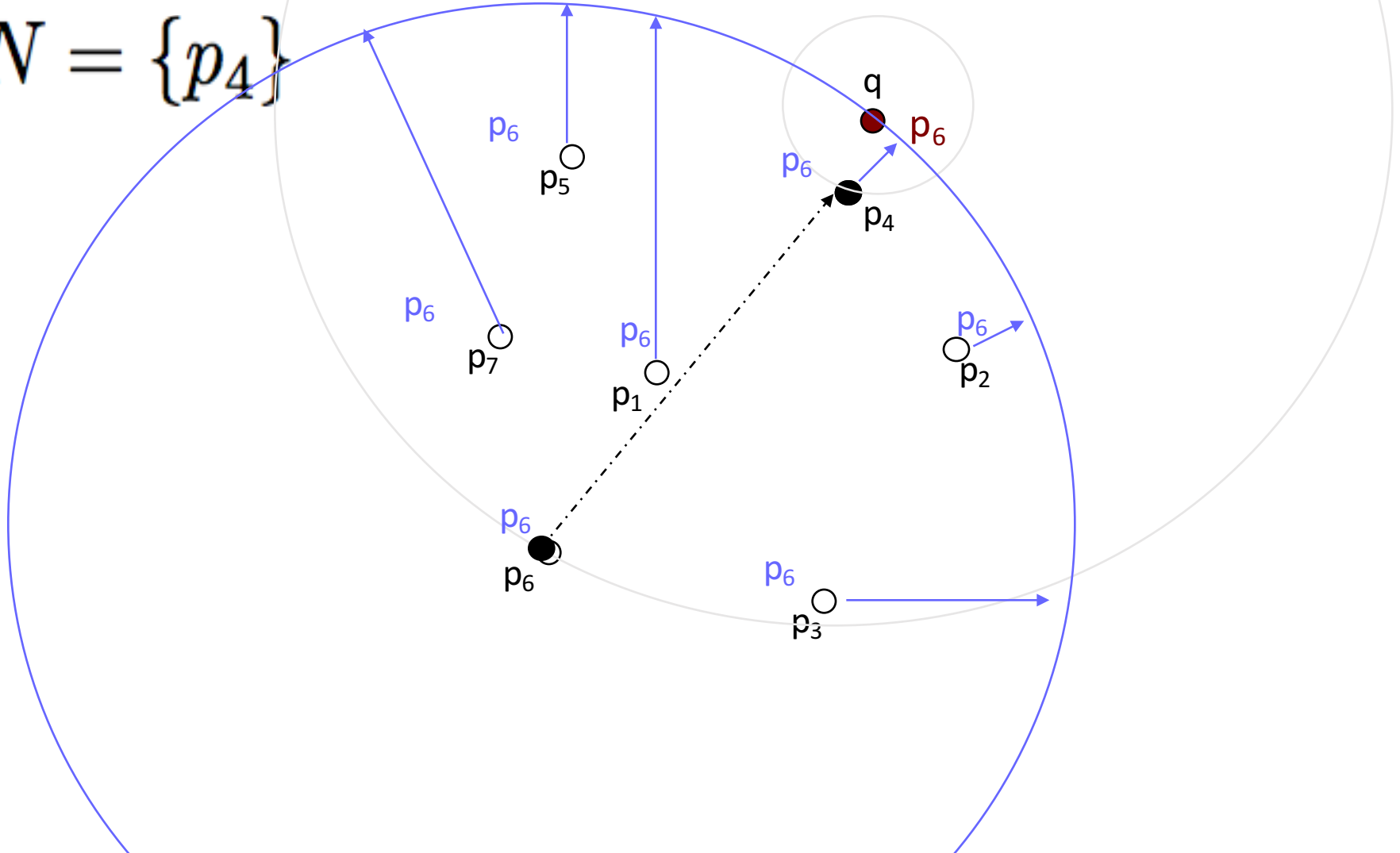
$$NN = \{p_4\}$$



iAESA2

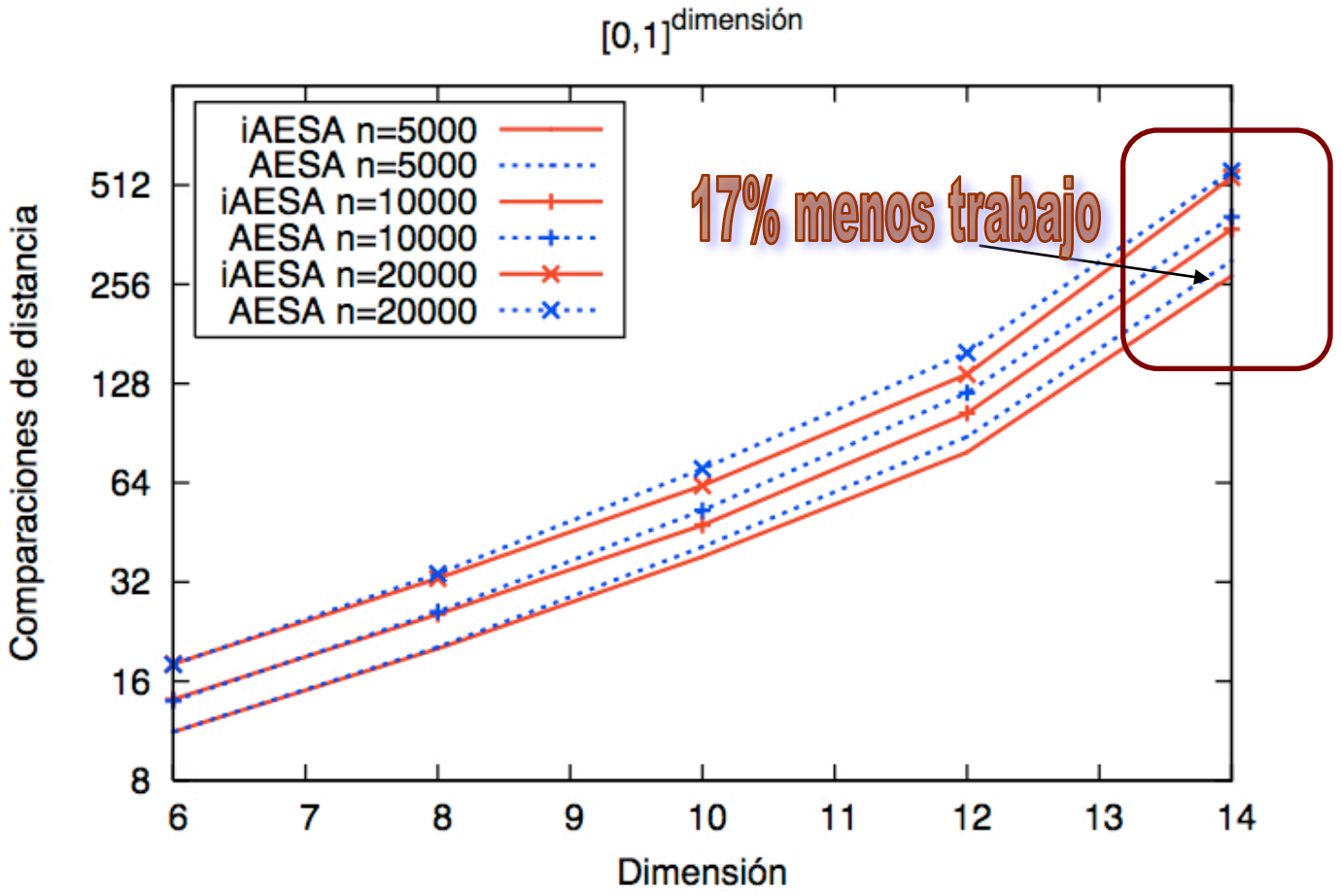
Dos estimadores de proximidad

$$NN = \{p_4\}$$



Resultados

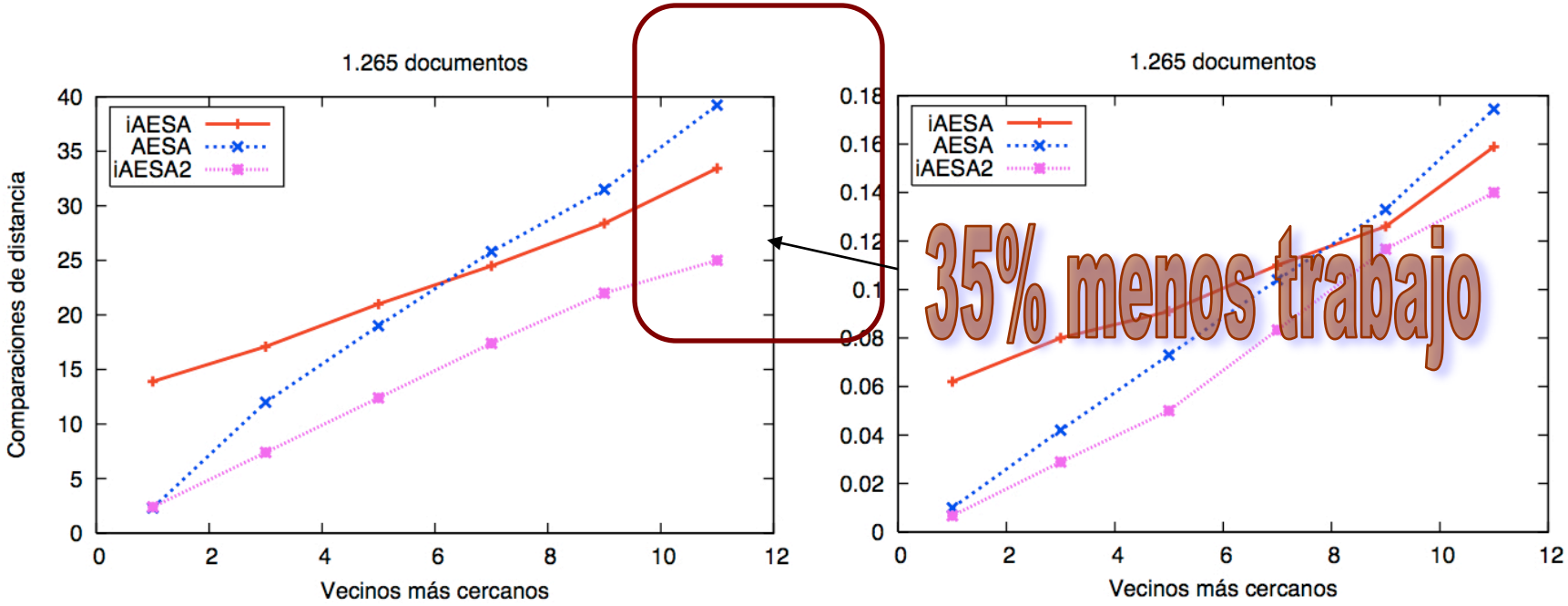
Cubo unitario



Exacto

Resultados

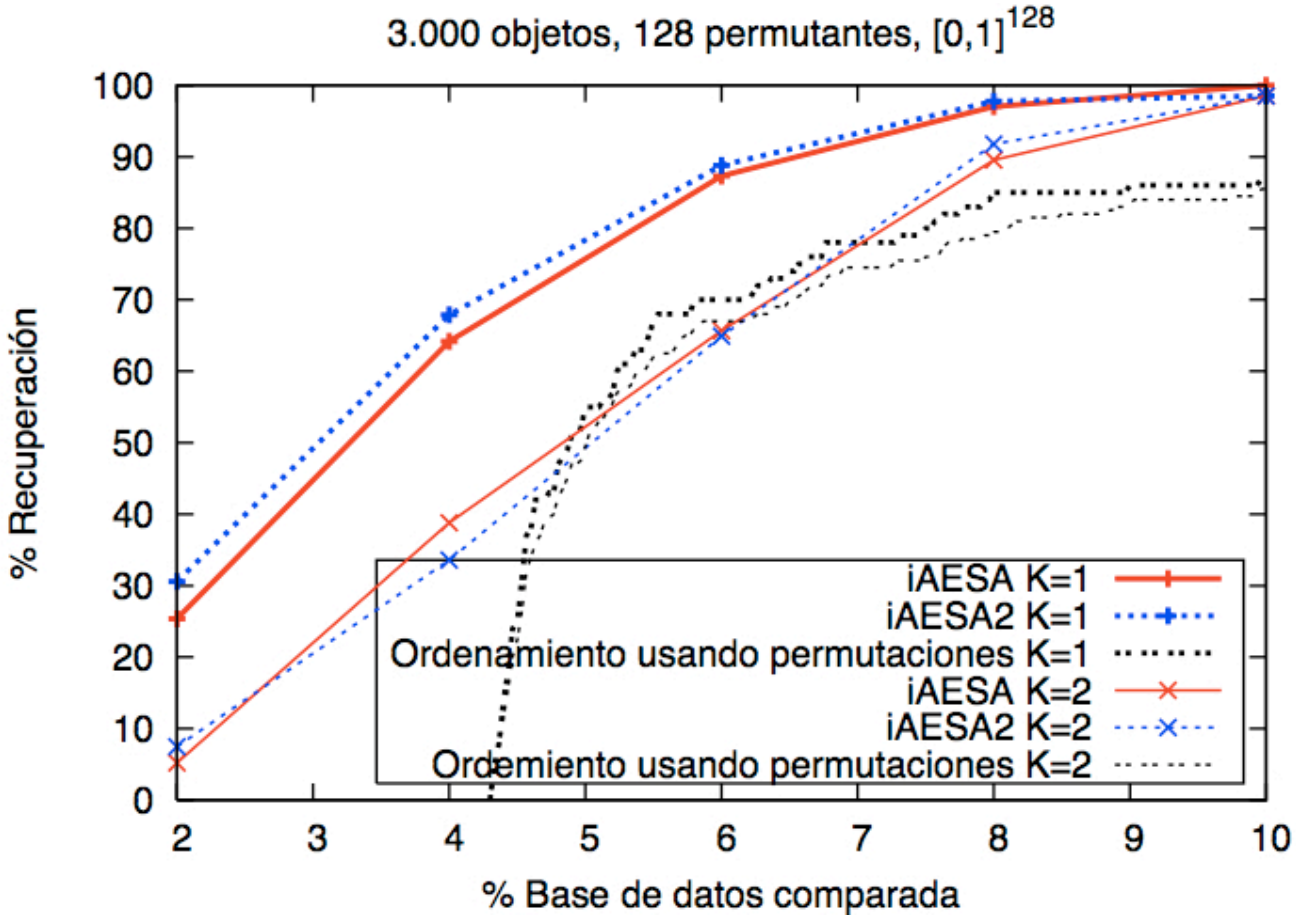
Documentos



Exacto

Resultados

iAESA contra Permutaciones



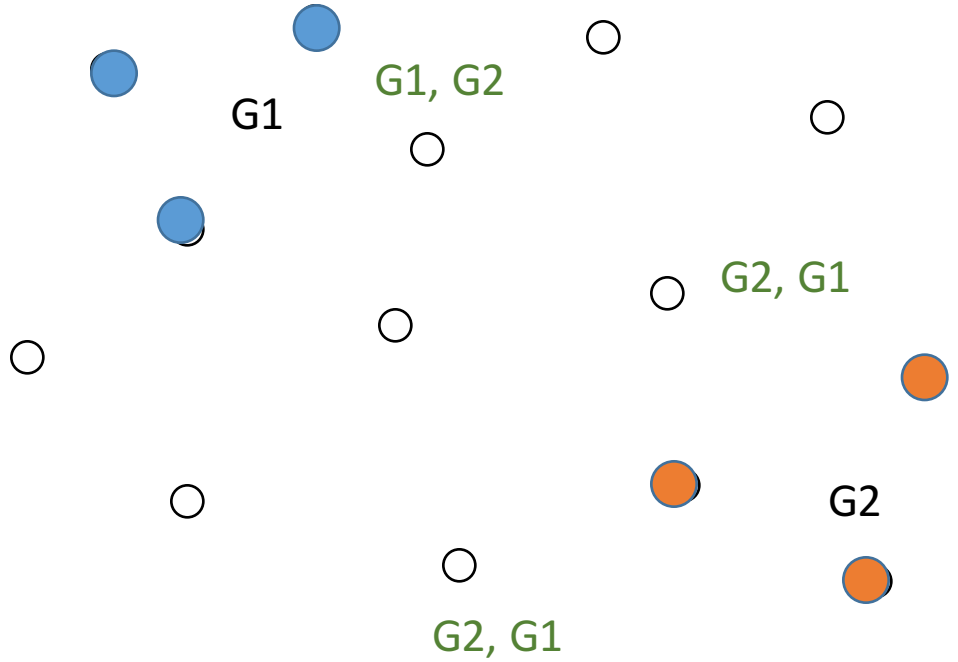
Exacto / Probabílístico

Grupos de permutaciones

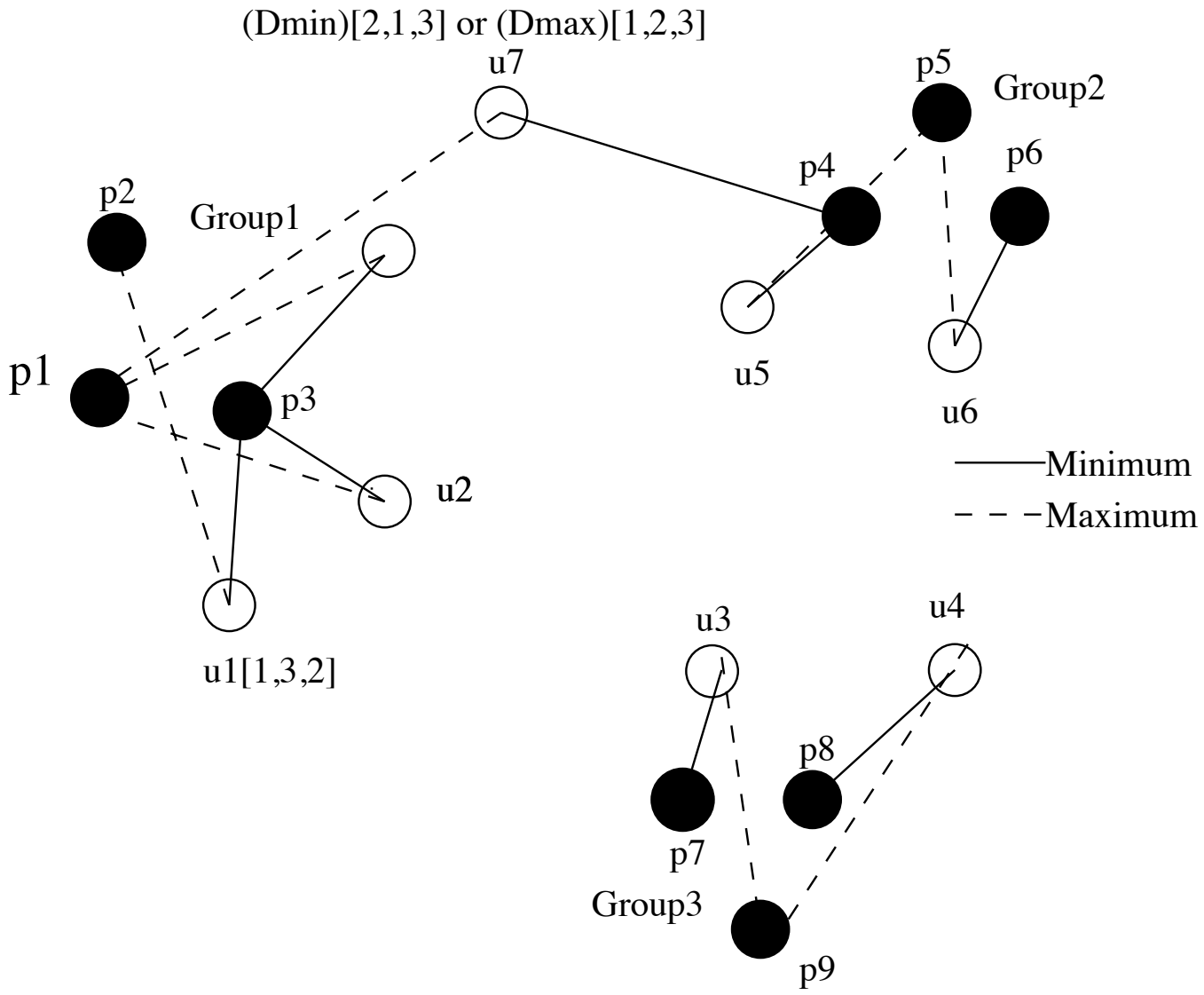
K. Figueroa, R. Paredes. Efficient Group of Permutants for Proximity Searching. MCPR 2011.

En proceso la versión de revista

Idea



Idea


(Dmin)[2,1,3] or (Dmax)[1,2,3]
u7
p5
Group2
p6
p4
p2
Group1
p1
p3
u5
u6
u2
Minimum
Maximum
u1[1,3,2]
u3
u4
p8
p7
Group3
p9

Parámetros

Distancia al grupo

- Dmin

$$D_{i_{min}}(u, G_i) = min_{\forall p \in G_i}\ d(p, u)$$

- Dmax

$$D_{i_{max}}(u, G_i) = max_{\forall p \in G_i}\ d(p, u)$$

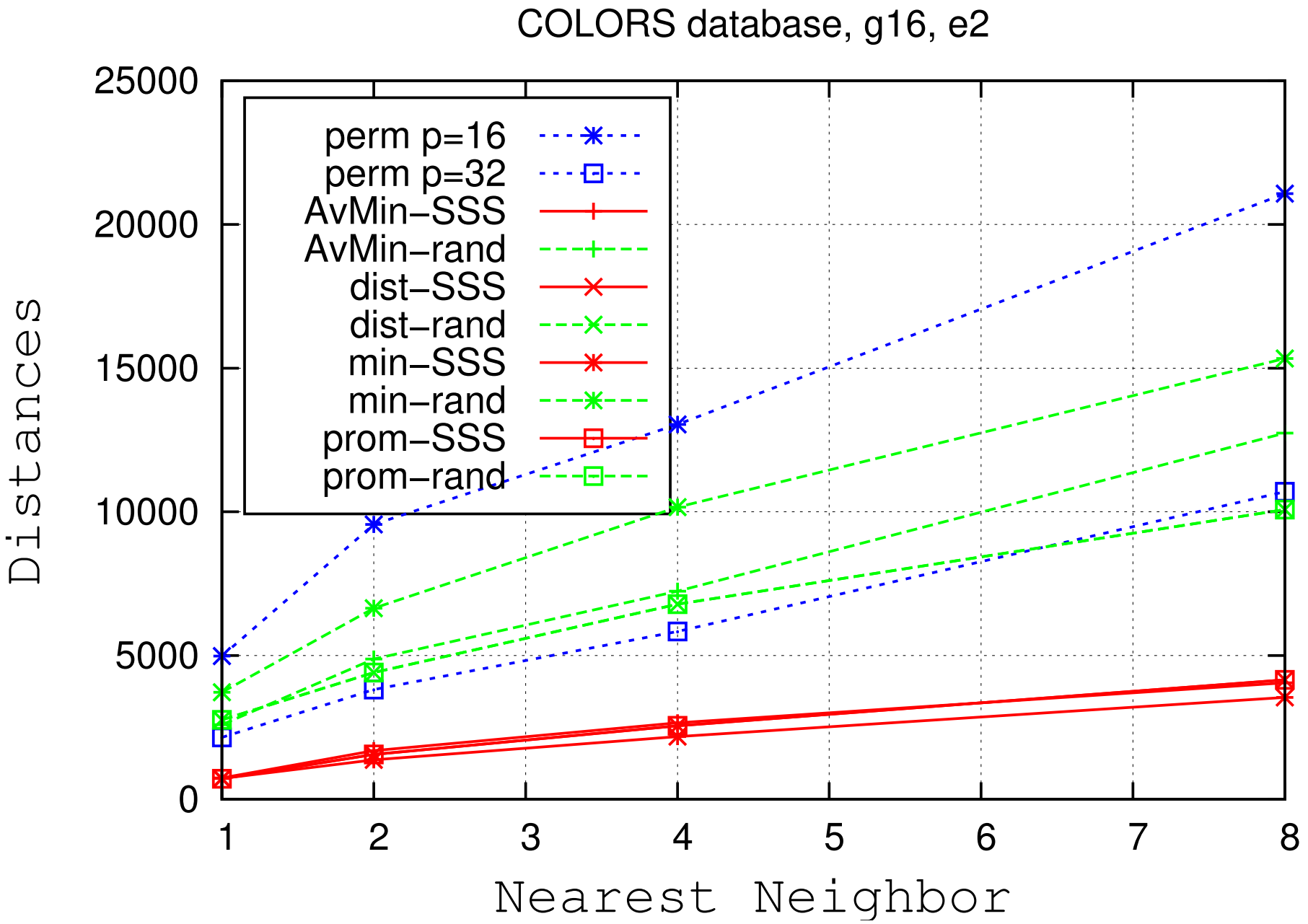
- Dav

$$D_{i_{av}}(u, G_i) = \frac{\sum_{\forall p \in G_i}\ d(p,u)}{|G_i|}$$

- Dmin+Dav

Selección de los grupos

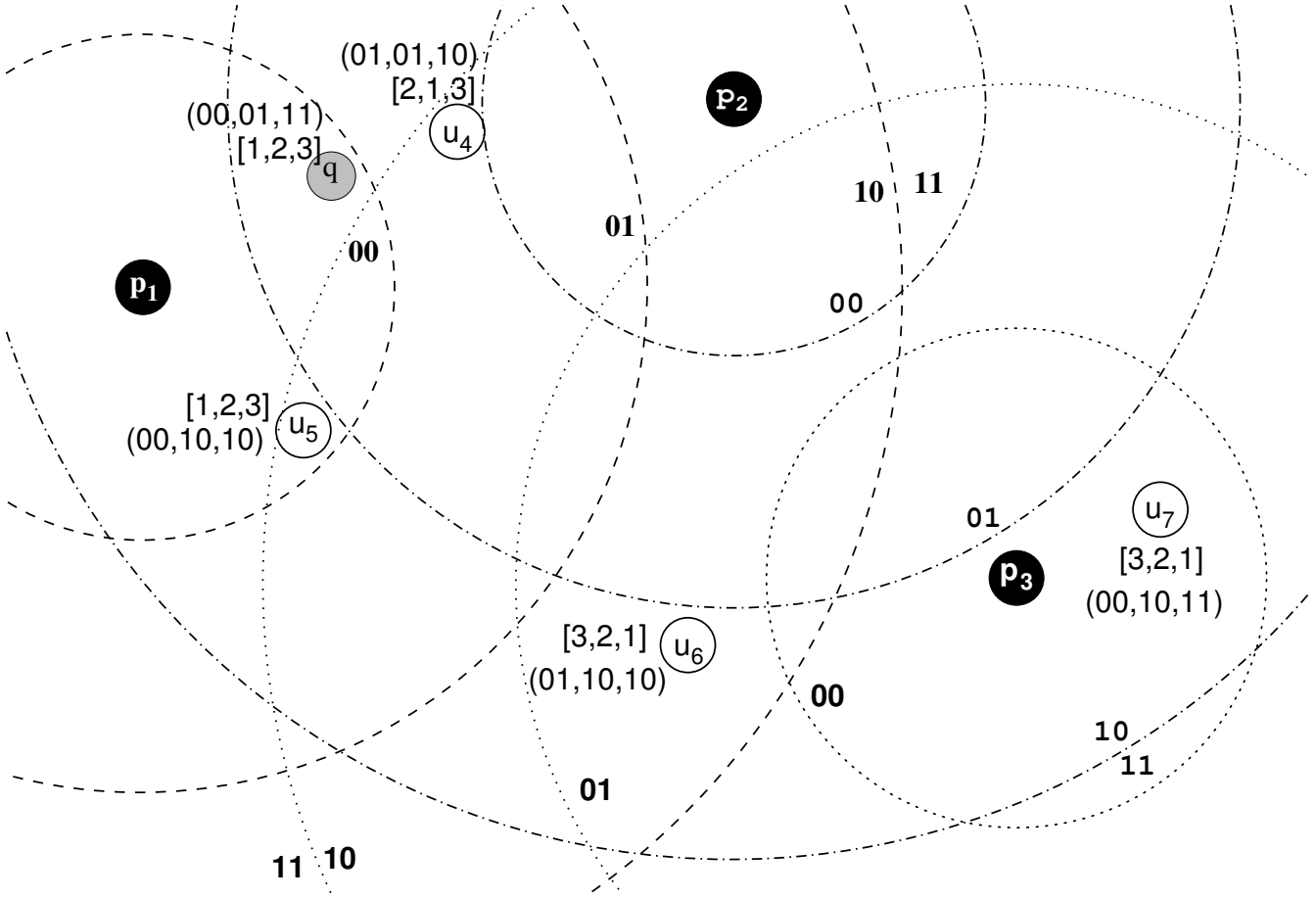
- Aleatorio
- Cercano a un elemento
- Lejano a un elemento
- Cercanos entre todos los elementos del grupo
- SSS


COLORS database, g16, e2
Distances
Nearest Neighbor
25000
20000
15000
10000
5000
0
1
2
3
4
5
6
7
8
perm p=16
perm p=32
AvMin–SSS
AvMin–rand
dist–SSS
dist–rand
min–SSS
min–rand
prom–SSS
prom–rand

Permutaciones y zonas

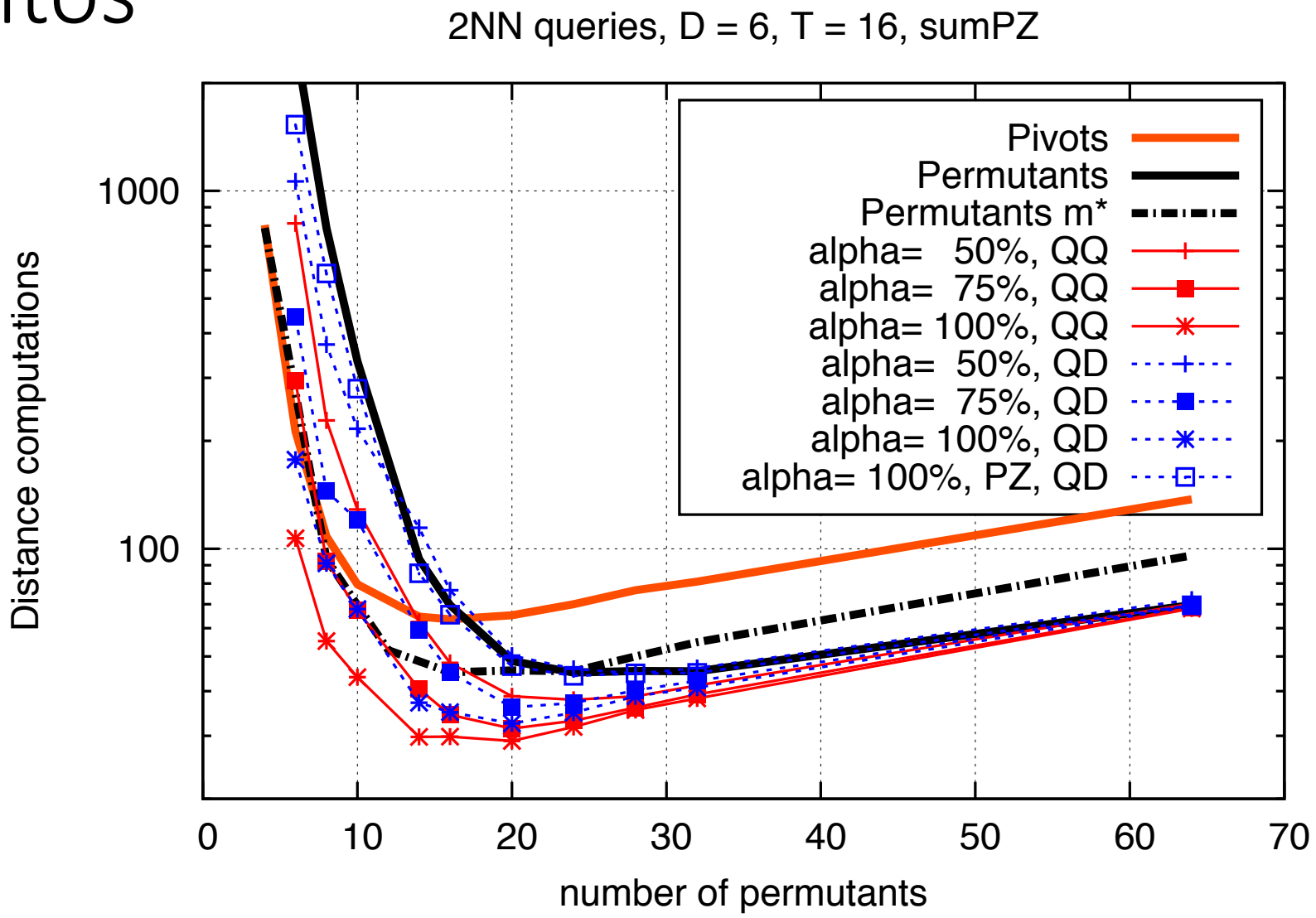
Karina Figueroaa, Rodrigo Paredes, Antonio Camarena-Ibarrola, Héctor Tejeda-Villela. Improving the permutation-based proximity searching algorithm using zones and partial information. Pattern Recognition Letters 95 (2017) 29–36

Idea



Realmente se ocupa toda la permutación?

Experimentos



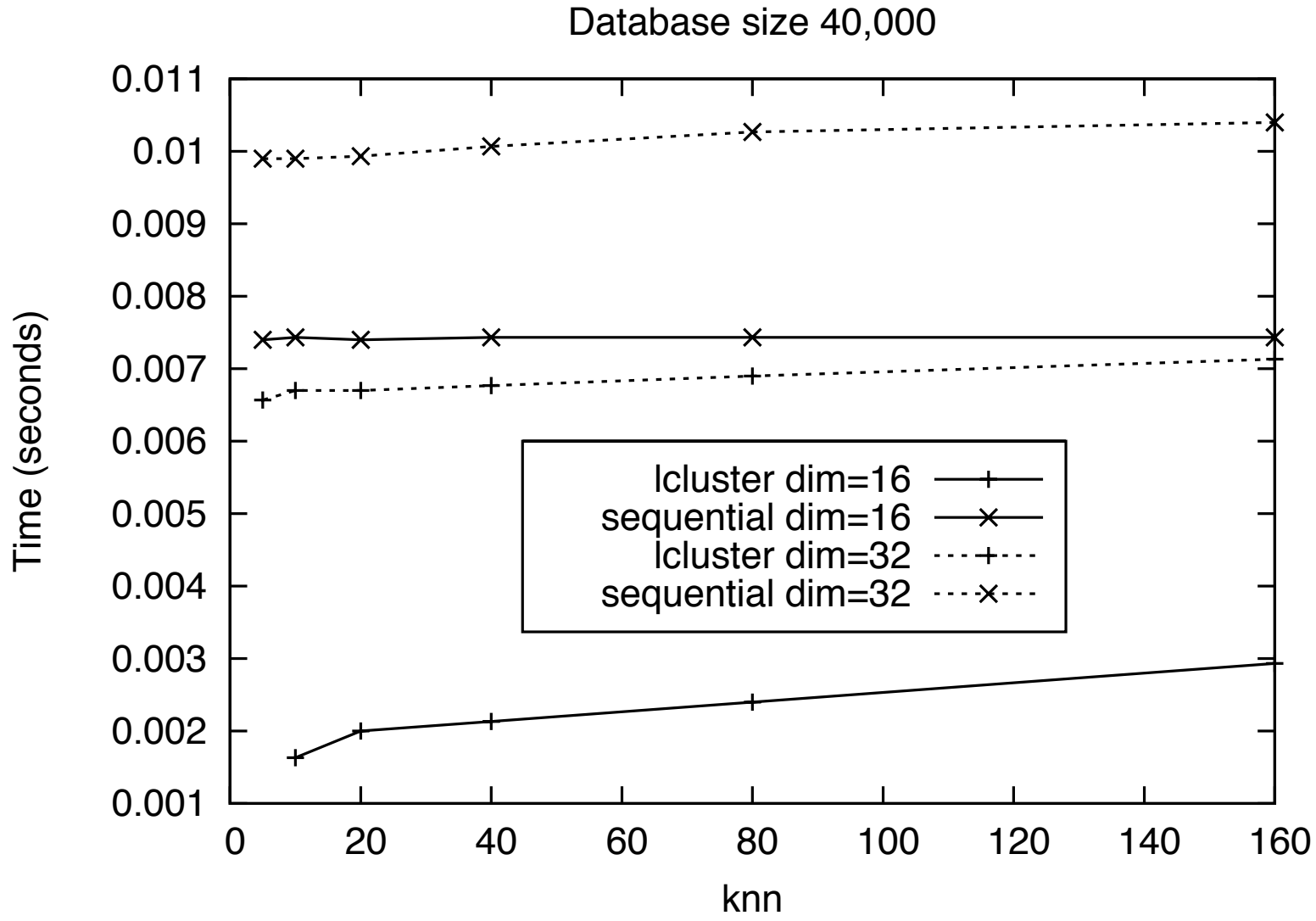
Indices para permutaciones

Karina Figueroa and Kimmo Fredriksson. Speeding up permutation based indexing with indexing. In *Similarity Search and Applications, 2009. SISAP '09. Second International Workshop on*, pages 107–114, Aug 2009.

Idea: índices en cascada

- Durante la búsqueda el algoritmo basado en permutantes consiste en:
- *Recuperar aquellos elementos que tengan la permutaciones mas similares…….*
- Qué tenemos
  - Espacio de permutaciones y una función de distancia! (footrule)
- Por qué no usar algún algoritmo de búsqueda por similaridad en un espacio métrico!!!???

Base de datos de vectores Gaussianos



Metric Inverted File

Giuseppe Amato and Pasquale Savino. Approximate similarity search in metric spaces using inverted les. In *Proceedings of the 3rd international conference on Scalable information systems*, InfoScale '08, pages 28:1–28:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social- Informatics and Telecommunications Engineering).

Indexando las permutaciones

Object	Permutation
o_1	(p_3, p_4, p_1, p_2)
o_2	(p_4, p_2, p_1, p_3)
o_3	(p_3, p_4, p_1, p_2)
o_4	(p_1, p_2, p_4, p_3)
o_5	(p_2, p_1, p_4, p_3)
o_6	(p_1, p_2, p_4, p_3)
o_7	(p_2, p_1, p_4, p_3)
o_8	(p_3, p_4, p_1, p_2)
o_9	(p_4, p_3, p_2, p_1)
q	(p_4, p_3, p_1, p_2)

Posting Lists

$p_1 \rightarrow (o_1,3), (o_2,3), (o_3,3), (o_4,1), (o_5,2), (o_6,1), (o_7,2), (o_8,3)$

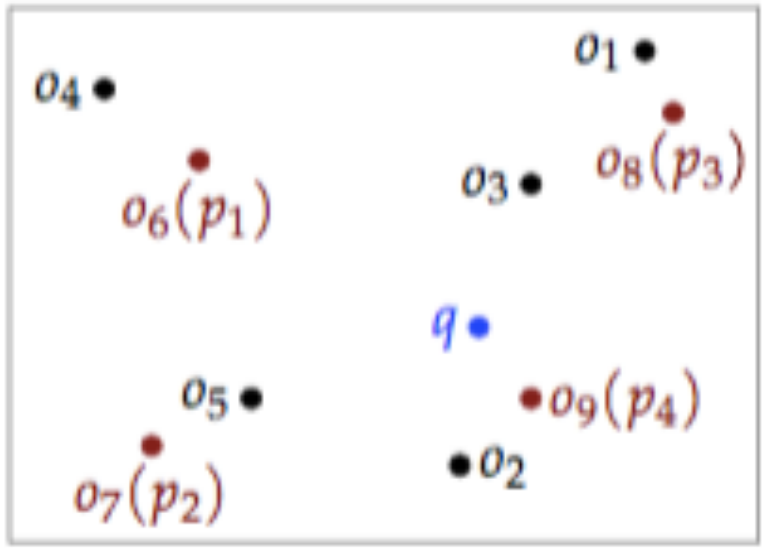
$p_2 \rightarrow (o_2,2), (o_4,2), (o_5,1), (o_6,2), (o_7,1), (o_9,3)$

$p_3 \rightarrow (o_1,1), (o_3,1), (o_8,1), (o_9,2)$

$p_4 \rightarrow (o_1,2), (o_2,1), (o_3,2), (o_4,3), (o_5,3), (o_6,3), (o_7,3), (o_8,2), (o_9,1)$

Object	Footrule(pivot_pos$_q$, pivot_pos$_{o_i}$)
o_9	$\lvert 2-2\rvert + \lvert 1-1\rvert = 0$
o_1	$\lvert 2-1\rvert + \lvert 1-2\rvert = 2$
o_3	$\lvert 2-1\rvert + \lvert 1-2\rvert = 2$
o_8	$\lvert 2-1\rvert + \lvert 1-2\rvert = 2$
o_2	$(m_i+1) + \lvert 1-1\rvert = 4$
o_4	$(m_i+1) + \lvert 1-3\rvert = 6$
o_5	$(m_i+1) + \lvert 1-3\rvert = 6$
o_6	$(m_i+1) + \lvert 1-3\rvert = 6$
o_7	$(m_i+1) + \lvert 1-3\rvert = 6$

Candidate points

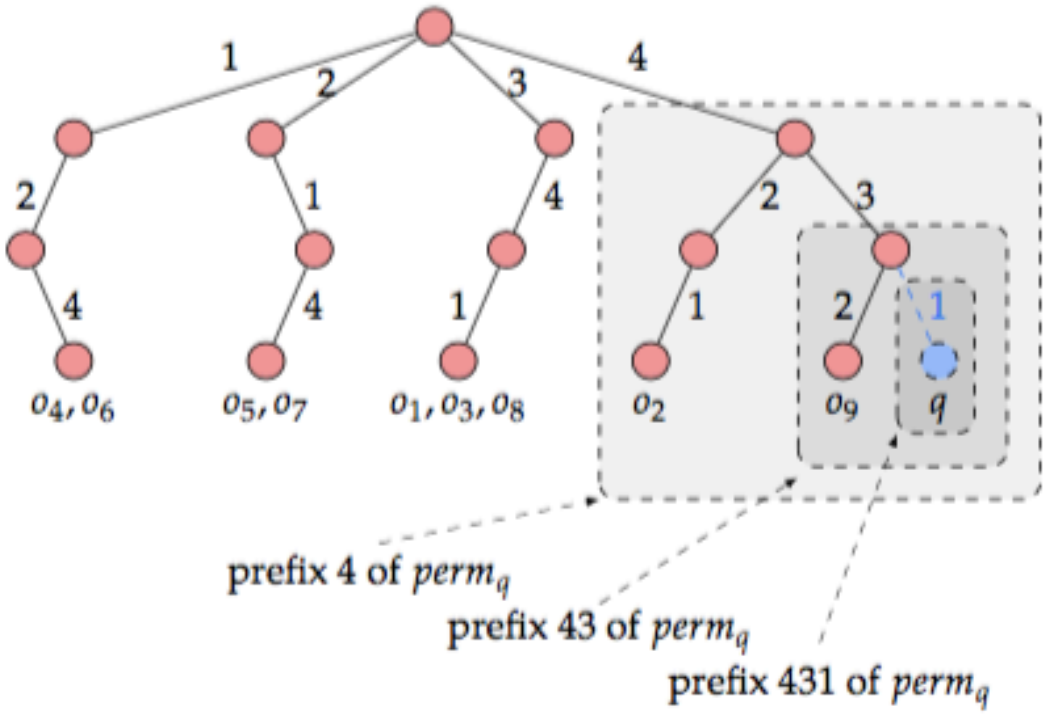


PP-INDEX

Andrea Esuli. PP-index: Using permutation prefixes for efficient and scalable approximate similarity search. In *Proceedings of LSDS-IR*, 2009.

Indexar la búsqueda de permutaciones

Object	Permutation
o_1	(p_3, p_4, p_1, p_2)
o_2	(p_4, p_2, p_1, p_3)
o_3	(p_3, p_4, p_1, p_2)
o_4	(p_1, p_2, p_4, p_3)
o_5	(p_2, p_1, p_4, p_3)
o_6	(p_1, p_2, p_4, p_3)
o_7	(p_2, p_1, p_4, p_3)
o_8	(p_3, p_4, p_1, p_2)
o_9	(p_4, p_3, p_2, p_1)
q	(p_4, p_3, p_1, p_2)



Aplicaciones


Capture Me File Edit Capture Window Help
Sat 11:39 AM
WebFaces Sistema de Reconocimiento Caras y Detección de Imágenes Faciales
http://www.ciw.cl/webfaces/
Octubre 6, 2007
ciw.cl
centrodeinvestigaciónde la web
Departamento de Ciencias de la Computación
Universidad de Chile
Inicio
Descripción
Reconocimiento Facial
Detección de Caras
Detección de Piel
Publicaciones
Investigadores
Preguntas
English Version
Reconocimiento | Base de Datos Yale »»
15 imagenes de referencia -- Seleccione imagen
FIMOSA

Capture Me File Edit Capture Window Help

WebFaces Sistema de Reconocimiento Caras y Detección de Imágenes Faciales

http://www.ciw.cl/webfaces/

paginas en la web

Gmail post my delicious Flickr Apple (73) News (1418) itesm C++

Gmail | Busqueda Google de i... | Download free clip art... | Clip Art Download – C... | WebFaces Sistema de ... | ruiz solar – Google Se...

Octubre 6, 2007

ciw.cl
centrodeinvestigaciónde la web

Departamento de Ciencias de la Computación
Universidad de Chile

Inicio | Descripción | Reconocimiento Facial | Detección de Caras | Detección de Piel | Publicaciones | Investigadores | Preguntas | English Version

Reconocimiento | Imagenes de Referencia Yale | **Buscar »»**

Indique número de imagenes a encontrar: 8

Buscar imagenes similares

FIMOSA

Capture Me File Edit Capture Window Help Sat 11:41 AM

WebFaces Sistema de Reconocimiento Caras y Detección de Imágenes Faciales

http://www.ciw.cl/webfaces/

Octubre 6, 2007

ciw.cl centrodeinvestigaciondelaweb

Departamento de Ciencias de la Computación Universidad de Chile

Inicio | Descripción | Reconocimiento Facial | Detección de Caras | Detección de Piel | Publicaciones | Investigadores | Preguntas | English Version

Reconocimiento | Imagenes de Referencia Yale | Buscar | **Resultados Busqueda »»**

Busqueda por aproximacion...

0.71472	0.713184	0.707258	0.69639	0.690334	0.680642	0.680642	0.680408

FIMOSA

Capture Me File Edit Capture Window Help Sat 11:41 AM

WebFaces Sistema de Reconocimiento Caras y Detección de Imágenes Faciales

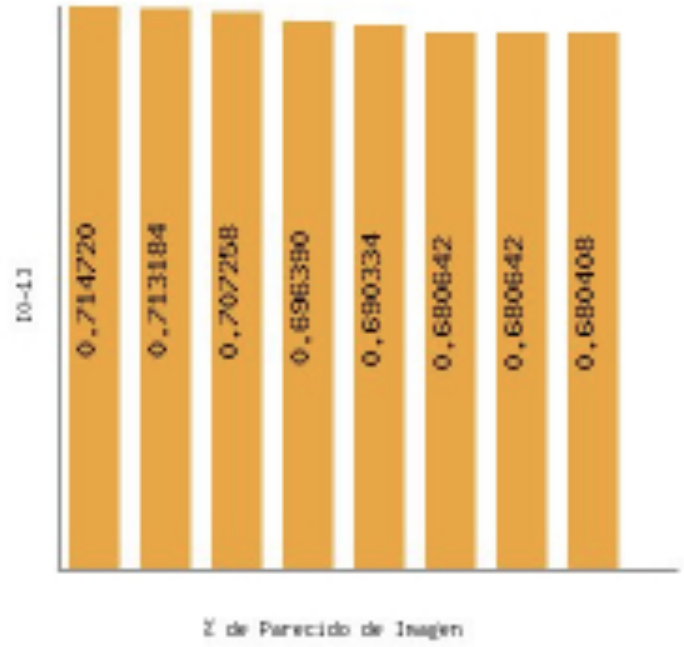
http://www.ciw.cl/webfaces/

Octubre 6, 2007

ciw.cl centrodeinvestigaciondelaweb

Departamento de Ciencias de la Computación Universidad de Chile

Inicio | Descripción | Reconocimiento Facial | Detección de Caras | Detección de Piel | Publicaciones | Investigadores | Preguntas | English Version



RESULTADOS

- Tiempo de búsqueda: 0.051 segundos
- Base de datos: Yale
- Componentes de base de datos: 150
- Imagenes encontradas: 8
- Seleccione Imágenes de resultado para ampliar
- Cada imagen tiene un índice de parecido

<<-- Estadísticas de los índices de parecido

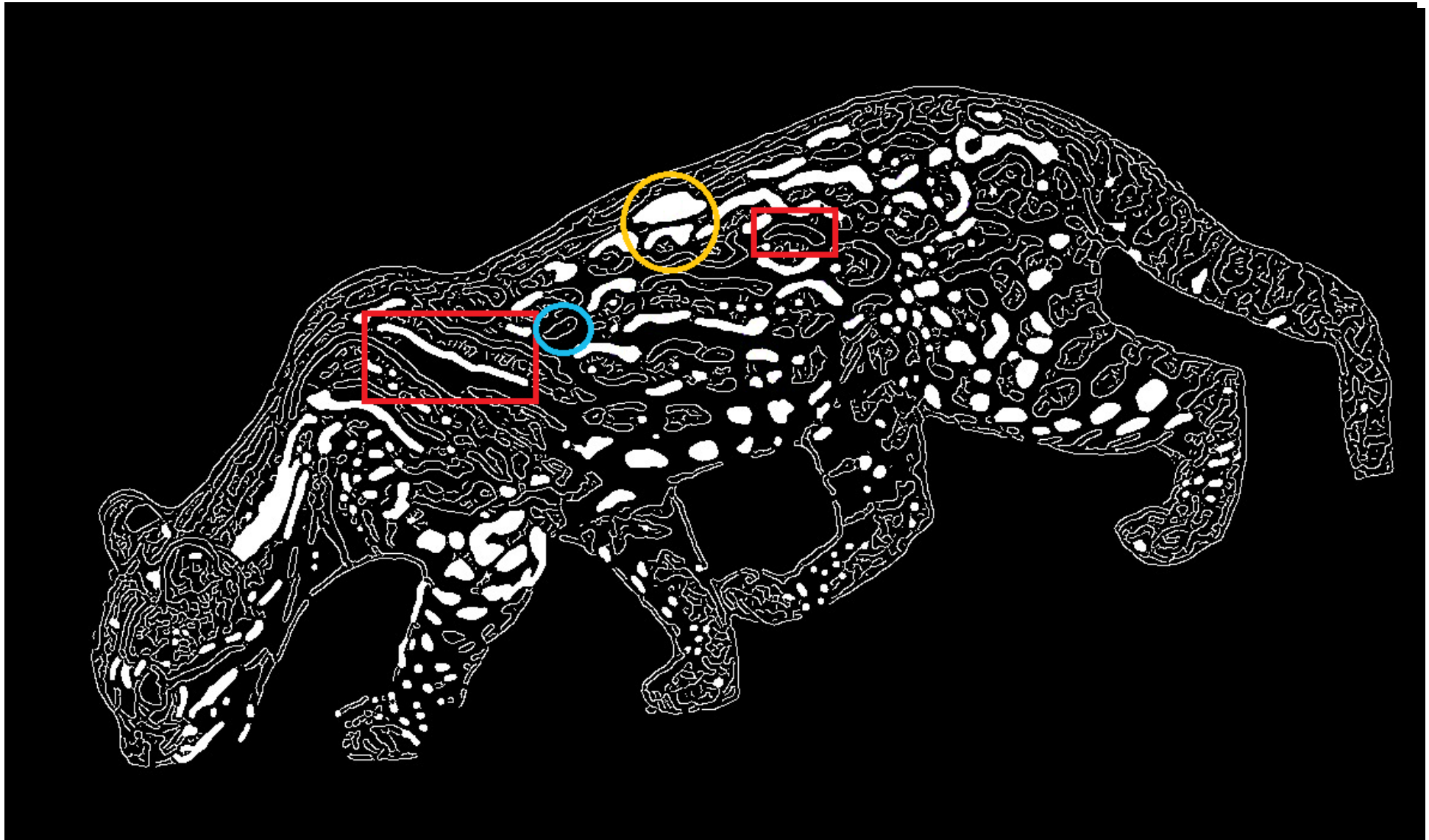
FIMOSA

Reconocimiento de Animales silvestres

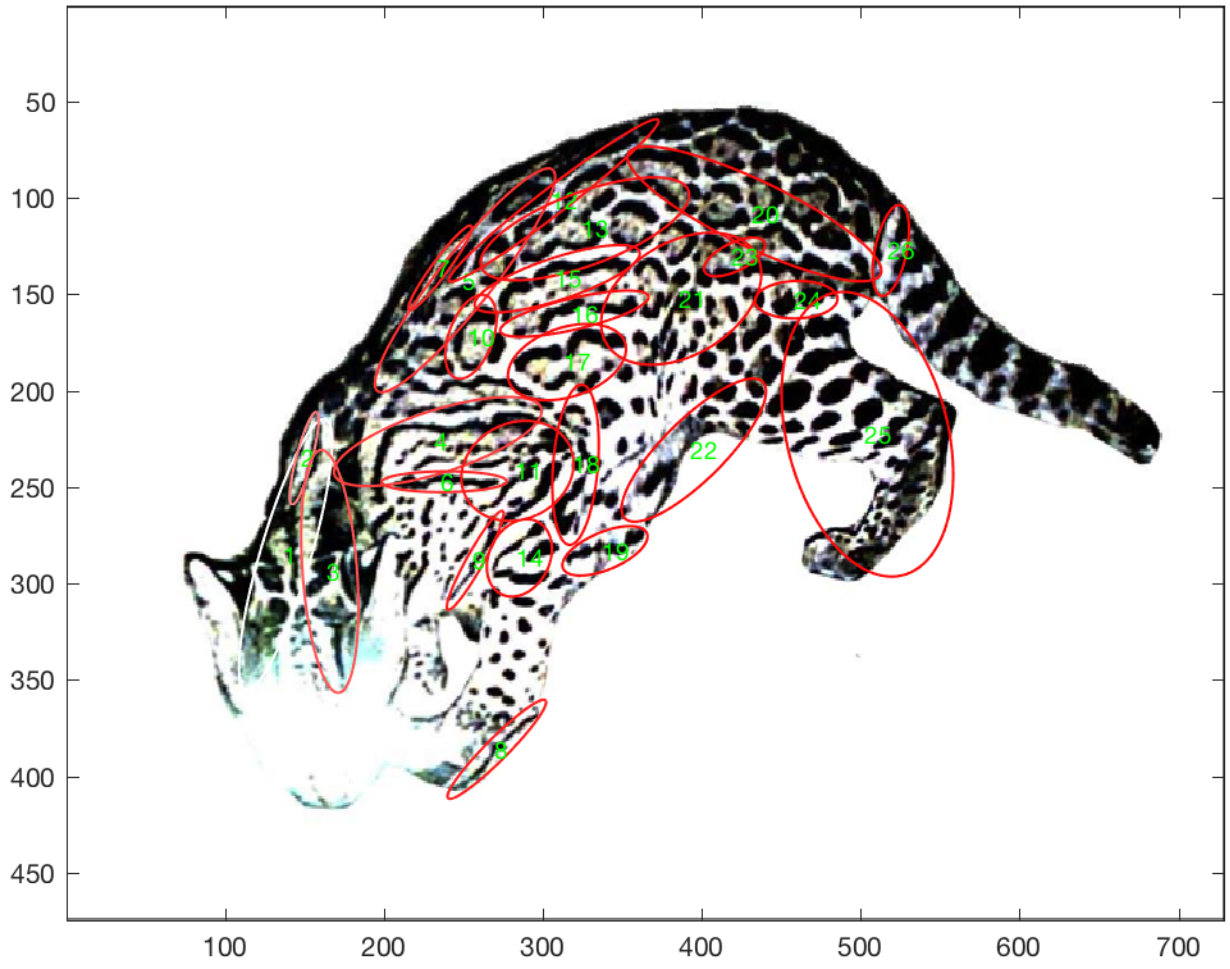
Segmentación

Segmentación

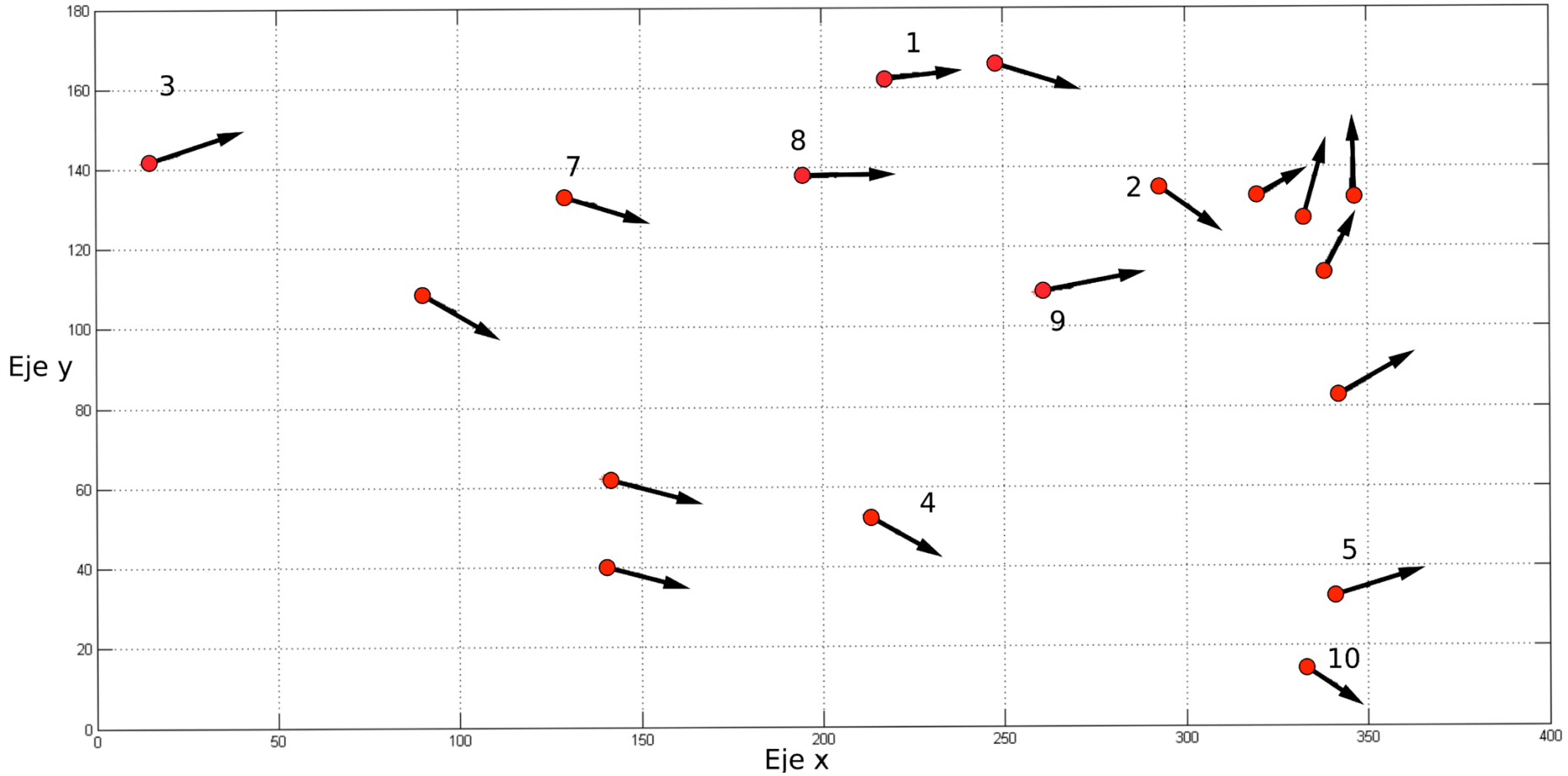
Identificación de manchas



Caracterizado



Esta historia continuará...



Otro escenario

Librería de espacios métricos

Karina Figueroa, Gonzalo Navarro, and Edgar Chavez. Metric spaces library, 2010. http://www.sisap.org/Metric_Space_Library.html.

Estructura

- Se puede bajar en http://www.sisap.org. Los directorios principales son:
- Readme.txt: ;)
- Copyright.txt: Copyright information. The code is licensed under a
- GNU Public License.
- src: Códigos fuentes de:
  - índices (subdirectorio indeces)
  - Espacios métricos (subdirectorio de espacios)
  - Y otros programas.
- dbs: Bases de datos para pruebas.
- lib: Módulos compilados.
- bin: Ejecutables compilados.
- doc: Manual.

In a nutshell

Preprocesamiento

Index **build**

(char *dbname, int n, int *argc, char ***argv)

void **saveIndex**

(Index S, char *fname)

void **freeIndex**

(Index S, bool closedb)

Búsquedas

int **search**

(Index S, Obj obj, Tdist r, bool show)

Tdist **searchNN**

(Index S, Obj obj, int k, bool show)

Index **loadIndex**

(char *fname)

```
/**
   This program is free software; you can redistribute it and/or modify
   it under the terms of the GNU General Public License as published by
   the Free Software Foundation; either version 3 of the License, or
   (at your option) any later version.

   This program is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU General Public License for more details.

   You should have received a copy of the GNU General Public License
   along with this program.  If not, see <http://www.gnu.org/licenses/>.
**/

#include"pivots.h"

void prnstats (Index S);

Index build (char *dbname, int n, int *argc, char ***argv)

   {...}

void freeIndex (Index S, bool libobj)

   {...}

void saveIndex (Index S, char *fname)

   {...}

Index loadIndex (char *fname)

   {...}

int search (Index S, Obj obj, Tdist r, int show)

   {...}

Tdist searchNN (Index S, Obj obj, int k, bool show)

   {...}

void prnstats (Index S)

   {...}
```

```
Karina-Figueroas-MacBook-Air:bin karina$ ./build-pivots-strings diccionario.txt -1 i.inx 4
indexing 10 objects out of 10...
number of elements: 10
number of pivots: 4
finished... 40 distances computed
user: 0.00, system: 0.00


saving...
saved... 184 bytes
```

```
Karina-Figueroas-MacBook-Air:bin karina$ cat diccionario.txt
hola
casa
cama
carro
cosa
caro
cana
cala
caso
cata
```

```
Karina-Figueroas-MacBook-Air:bin karina$ ./query-pivots-strings i.inx
reading index...
read 184 bytes
-7,casa
casa
cama
cana
cata
cala
cosa
caso
kNNs at distance 1
1,casa
casa
cama
cosa
cana
cala
caso
cata
7 objects found
0
Total distances per query: 9.000000
freeing...
done
```

Gracias

Referencias

- Además de los artículos mencionados
- http://www.nmis.isti.cnr.it/amato/similarity-search-book/SAC-07-tutorial.pdf
- Similarity search: The metric space approach. P Zezula, G Amato, V Dohnal, M Batko. Springer-Verlag New York Inc.
- Searching in metric spacesE Chávez, G Navarro, R Baeza-Yates, JL Marroquín. ACM computing surveys (CSUR) 33 (3), 273-321