Recuperación de Información Multimedia

Índices Métricos

CC5213 – Recuperación de Información Multimedia
Departamento de Ciencias de la Computación
Universidad de Chile
Juan Manuel Barrios – https://juan.cl/mir/ – 2020

Espacios Métricos

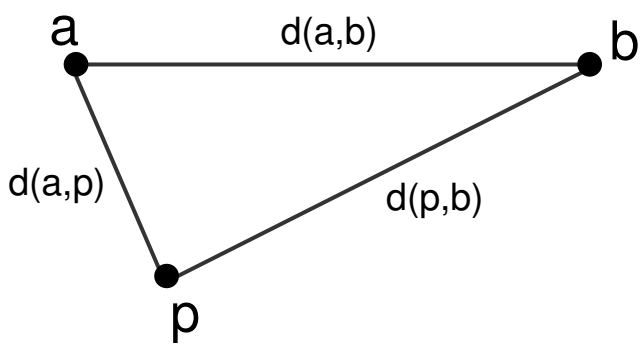
- Definición:
  - Universo de objetos válidos: $\mathcal{D}$
  - Función de distancia $d : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}$
  - El par $(\mathcal{D}, d)$ es un espacio métrico ssi d cumple con las propiedades métricas:
    - No negatividad $\forall x, y \in \mathcal{D} \quad d(x, y) \geq 0$
    - Reflexividad $\forall x \in \mathcal{D} \quad d(x, x) = 0$
    - Positividad $\forall x, y \in \mathcal{D}\ x \neq y \quad d(x, y) > 0$
    - Simetría $\forall x, y \in \mathcal{D} \quad d(x, y) = d(y, x)$
    - Desigualdad triangular $\forall x, y, z \in \mathcal{D} \quad d(x, z) \leq d(x, y) + d(y, z)$

Espacios Métricos

- La función d entrega el grado de disimilitud entre dos objetos cualquiera de $\mathcal{D}$
- $\mathcal{D}$ es el dominio y d es la métrica
- Ejemplos de espacio métrico:
  - Vectores y alguna distancia de Minkowski (L_p)
  - Signatures y distancia EMD
  - Strings y distancia de edición
  - Objetos multimedia con una distancia entrenada (metric learning)

Propiedades Métricas

- Dados tres objetos {a,b,p} la desigualdad triangular nos entrega una cota superior para d(a,b)
- Usando las propiedades métricas se puede calcular una cota inferior para d(a,b)



$$
\begin{aligned}
(1)\ \ d(a,b) &\leq d(a,p) + d(p,b) \\
(2)\ \ d(a,p) &\leq d(a,b) + d(b,p) \\
(3)\ \ d(b,p) &\leq d(b,a) + d(a,p)
\end{aligned}
$$

$$
\begin{aligned}
(2) \Rightarrow\ \ d(a,p) - d(b,p) &\leq d(a,b) \\
(3) \Rightarrow\ \ d(a,p) - d(b,p) &\geq -d(b,a)
\end{aligned}
$$

$$\Rightarrow\ \ -d(a,b) \leq d(a,p) - d(p,b) \leq d(a,b)$$

$$\Rightarrow\ \ |d(a,p) - d(p,b)| \leq d(a,b)$$

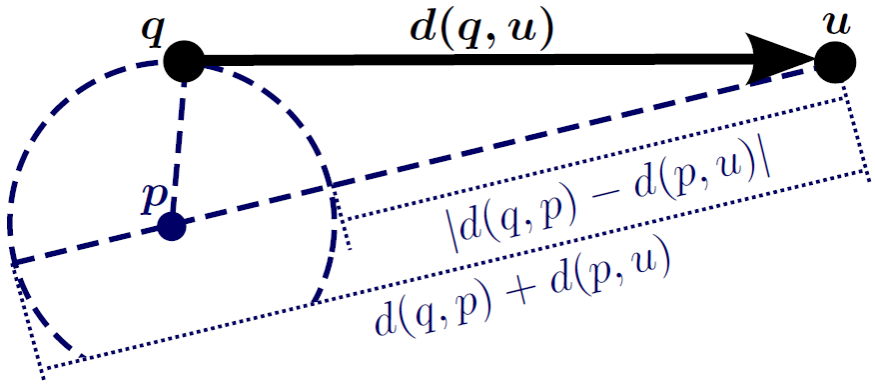
Metric Access Methods

- Se desea realizar búsquedas por similitud en un conjunto de objetos *R* usando una métrica *d*
- Metric Access Methods (MAM) son estructuras de datos que permiten resolver búsquedas por similitud en *R* explotando las propiedades métricas de la función de distancia
  - Objetivo: reducir el número de veces que se evalúa la función de distancia
  - Supuesto: la función de distancia es costosa de evaluar

Pivote

- Un pivote *p* es un objeto de la colección fijo
- Permite acotar el valor de *d* mediante el uso de las propiedades métricas:

$$|d(q,p) - d(p,u)| \;\leq\; d(q,u) \;\leq\; d(q,p) + d(p,u)$$



Conjuntos de Pivotes

- Las cotas se vuelven más ajustadas cuando se usan varios pivotes
- Dado un conjunto de pivotes $P = \{p_1,\ldots,p_k\}$:
  - $UB_P(q,u)$ es la cota superior (upper bound) de $d(q,u)$:

$$UB_{\mathcal{P}}(q,u) = \min_{p_i \in \mathcal{P}} \{d(q,p_i) + d(p_i,u)\}$$

  - $LB_P(q,u)$ es la cota inferior (lower bound) de $d(q,u)$:

$$LB_{\mathcal{P}}(q,u) = \max_{p_i \in \mathcal{P}} \{|d(q,p_i) - d(p_i,u)|\}$$

Tablas de Pivotes

- AESA (Approximating and Eliminating Search Algorithm)
  - Usar todos los objetos de *R* como pivote
  - Requiere mantener una tabla de distancias entre todos los pares de objetos del dataset
    - Memoria $O(n^2)$

Ver Vidal. 1986.

- LAESA (Linear AESA)
  - Seleccionar subconjunto de elementos de la colección como pivotes
  - ¿Cómo resolver una búsqueda eficientemente?
  - ¿Cómo seleccionar el conjunto de pivotes?

Ver Micó et al. 1994.

LAESA

- Para una colección de *n* objetos se seleccionan *k* pivotes (al azar o mediante algún algoritmo)
- Calcular una tabla de *n* x *k* con las distancias de cada objeto con cada pivote

	p_1	...	p_k
u_1	$d(p_1,u_1)$	...	$d(p_k,u_1)$
...	...	...	...
u_n	$d(p_1,u_n)$	...	$d(p_k,u_n)$

Consulta por Rango

- Calcular la distancia entre *q* y cada pivote
- Para cada objeto:
  - Calcular su cota inferior
  - Criterio de exclusión: Si la cota inferior es mayor que *r* el objeto no es relevante (se descarta)
    - Notar que basta un pivote para descartar el objeto, por lo que no es necesario evaluar todos los pivotes
  - Si no pudo ser descartado se evalúa su distancia real y se determina si es relevante o no

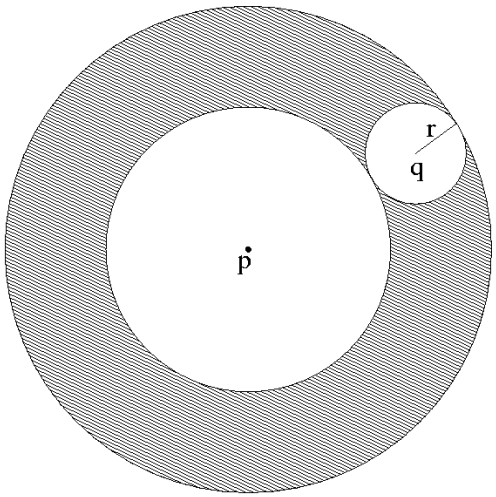
```
foreach p_i ∈ P do
    evaluar d(p_i, q) y guardar
end
queue ← ∅ ;
foreach u_i ∈ R do
    if LB_P(q, u_i) > r then
        continue;
    else if d(q, u_i) ≤ r then
        queue.Add(u_i) ;
    end
end
Print(queue);
```

Criterio de Exclusión

- El criterio de exclusión consiste en descartar todos los objetos *u* que cumplan:

$$|d(q,p) - d(p,u)| > r$$

- Gráficamente en el plano, usando un pivote se descartan todos los objetos que están fuera de un anillo centrado en p



Ejemplo consulta por rango

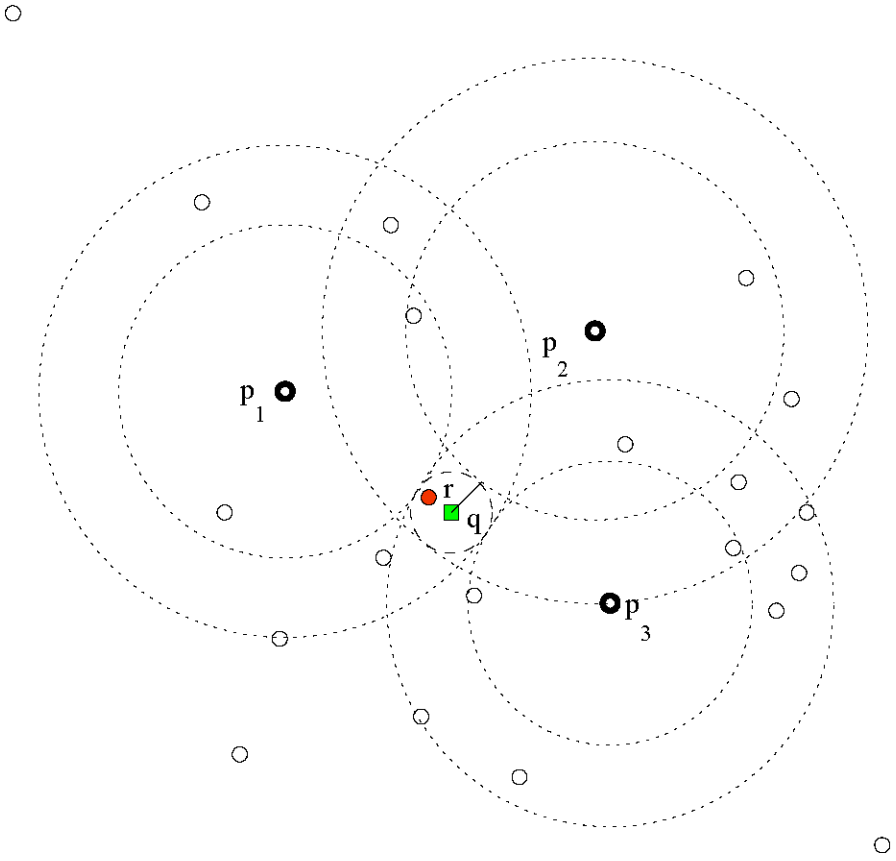


Tabla de Pivotes

$d(p_1,u_1)$	$d(p_2,u_1)$	$d(p_3,u_1)$
$d(p_1,u_2)$	$d(p_2,u_2)$	$d(p_3,u_2)$
…	…	…
$d(p_1,u_n)$	$d(p_2,u_n)$	$d(p_3,u_n)$

Criterio de exclusión:

$$LB_P(q,u_i) > r$$

Consulta k-NN

- Similar a una consulta por rango donde *r* es la distancia al candidato actual
- Calcular la distancia entre *q* y cada pivote
- Para cada objeto:
  - Calcular su cota inferior
  - Si la cota inferior es mayor que el candidato actual el objeto no es relevante (se descarta)
  - Si no pudo ser descartado se evalúa su distancia real y se determina si es mejor que el candidato actual o no

```
foreach p_i ∈ P do
    evaluar d(p_i, q) y guardar
end
candidate ← null ;
candidate_dist ← +∞ ;
foreach u_i ∈ R do
    if LB_P(q, u_i) ≥ candidate_dist then
        continue;
    end
    dist ← d(u_i, q) ;
    if dist < candidate_dist then
        candidate ← u_i ;
        candidate_dist ← dist;
    end
end
Print(candidate);
```

Ejemplo (1)

Sea ***R*** un conjunto de 16 objetos (a–p) (vectores de cinco dimensiones, comparados con distancia L_1)

a	8	2	5	2	0	
b	10	3	2	1	3	
c	12	4	1	2	1	
d	2	3	0	1	0	← **P1**
e	6	3	2	1	7	
f	9	1	0	4	3	
g	7	3	5	3	3	
h	10	0	1	2	3	
i	9	1	1	3	1	
j	8	2	9	3	0	← **P2**
k	9	1	1	1	2	
l	3	4	1	1	2	
m	7	3	6	1	1	
n	9	3	5	3	3	← **P3**
o	3	4	5	1	0	
p	5	2	2	2	1	

Se escogen 3 objetos como pivotes P={d, j, n}

La construcción del índice consiste en calcular la distancia de cada objeto con cada pivote

Ejemplo (2)

	a	8	2	5	2	0
	b	10	3	2	1	3
	c	12	4	1	2	1
P1 →	d	2	3	0	1	0
	e	6	3	2	1	7
	f	9	1	0	4	3
	g	7	3	5	3	3
	h	10	0	1	2	3
	i	9	1	1	3	1
P2 →	j	8	2	9	3	0
	k	9	1	1	1	2
	l	3	4	1	1	2
	m	7	3	6	1	1
P3 →	n	9	3	5	3	3
	o	3	4	5	1	0
	p	5	2	2	2	1

	P1	P2	P3
a	13	5	6
b	13	15	6
c	14	16	11
d	0	18	17
e	13	19	12
f	15	15	8
g	15	9	2
h	16	16	9
i	13	11	8
j	18	0	9
k	12	14	9
l	5	19	14
m	12	8	7
n	17	9	0
o	7	13	12
p	8	12	11

Tabla de Pivotes:

Matriz de distancias entre cada objeto y cada pivote (usando distancia L_1)

A continuación se desea buscar el vecino más cercano de un nuevo descriptor ***q***…

Ejemplo (3)

	a	8	2	5	2	0
	b	10	3	2	1	3
	c	12	4	1	2	1
P1 →	d	2	3	0	1	0
	e	6	3	2	1	7
	f	9	1	0	4	3
	g	7	3	5	3	3
	h	10	0	1	2	3
	i	9	1	1	3	1
P2 →	j	8	2	9	3	0
	k	9	1	1	1	2
	l	3	4	1	1	2
	m	7	3	6	1	1
P3 →	n	9	3	5	3	3
	o	3	4	5	1	0
	p	5	2	2	2	1
	q	**7**	**5**	**7**	**2**	**1**

	P1	P2	P3
a	13	5	6
b	13	15	6
c	14	16	11
d	0	18	17
e	13	19	12
f	15	15	8
g	15	9	2
h	16	16	9
i	13	11	8
j	18	0	9
k	12	14	9
l	5	19	14
m	12	8	7
n	17	9	0
o	7	13	12
p	8	12	11
q	**16**	**8**	**9**

Primero calcular la distancia de ***q*** a cada pivote

Búsqueda NN:
Para cada elemento ***u*** se calcula la cota inferior LB(***q***,***u***)

$$LB_{\mathcal{P}}(q,u) = \max_{p_i \in \mathcal{P}} \{|d(q,p_i) - d(p_i,u)|\}$$

Números en la tabla, escoger la máxima diferencia por fila

Si LB(***q***,***u***) es mayor o igual a la distancia del candidato actual a NN se descarta ***u***, si no se calcula d(**q**,**u**)…

Ejemplo (4)

	a	8	2	5	2	0
	b	10	3	2	1	3
	c	12	4	1	2	1
P1 →	d	2	3	0	1	0
	e	6	3	2	1	7
	f	9	1	0	4	3
	g	7	3	5	3	3
	h	10	0	1	2	3
	i	9	1	1	3	1
P2 →	j	8	2	9	3	0
	k	9	1	1	1	2
	l	3	4	1	1	2
	m	7	3	6	1	1
P3 →	n	9	3	5	3	3
	o	3	4	5	1	0
	p	5	2	2	2	1
	q	**7**	**5**	**7**	**2**	**1**

	P1	P2	P3
a	13	5	6
b	13	15	6
c	14	16	11
d	0	18	17
e	13	19	12
f	15	15	8
g	15	9	2
h	16	16	9
i	13	11	8
j	18	0	9
k	12	14	9
l	5	19	14
m	12	8	7
n	17	9	0
o	7	13	12
p	8	12	11
q	**16**	**8**	**9**

Cota Inferior	Distancia Real	Candidato
LB(*q*,*u*)	**d(*q*,*u*)**	**NN**
3	**7**	**a**
7	~~13~~	
8	~~12~~	
16	~~16~~	
11	~~15~~	
7	~~17~~	
7	~~7~~	
8	~~16~~	
3	**13**	**a**
8	~~8~~	
6	**14**	**a**
11	~~13~~	
4	**4**	**m**
9	~~9~~	
9	~~9~~	
8	~~10~~	

Costo: LB se evaluó 16 veces y d se evaluó 7 veces: 3 x d(q,p) + 4 x d(q,u)

Espacio de pivotes

- Espacio *k*-dimensional, donde cada coordenada es la distancia entre el objeto y cada pivote:

$$v_{\mathcal{P}}(u) = (d(p_1, u) \dots d(p_k, u))^T$$

- Notar que:

$$LB_{\mathcal{P}}(q, u_i) = \mathrm{L}_{\max}(v_{\mathcal{P}}(q), v_{\mathcal{P}}(u_i))$$

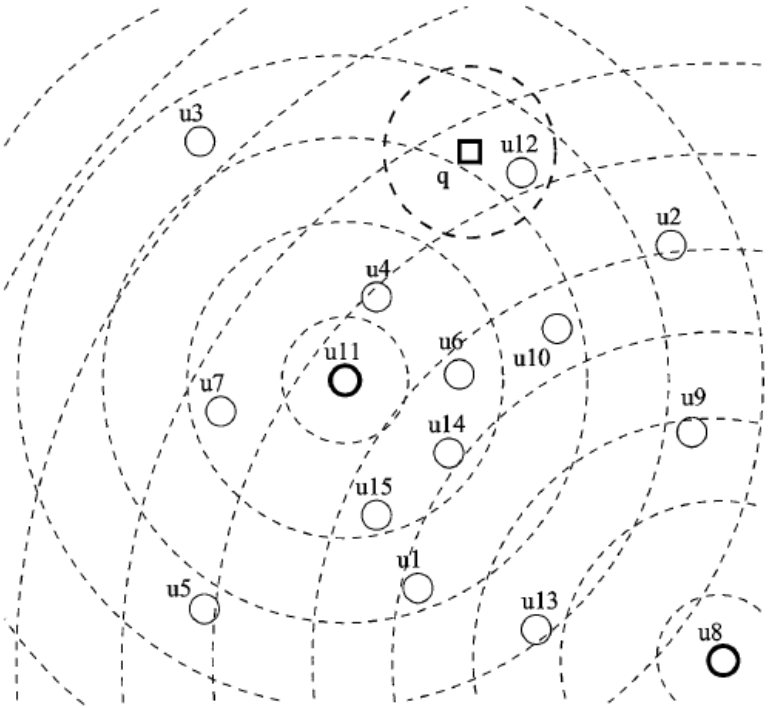
- Criterio de exclusión de la búsqueda por rango:

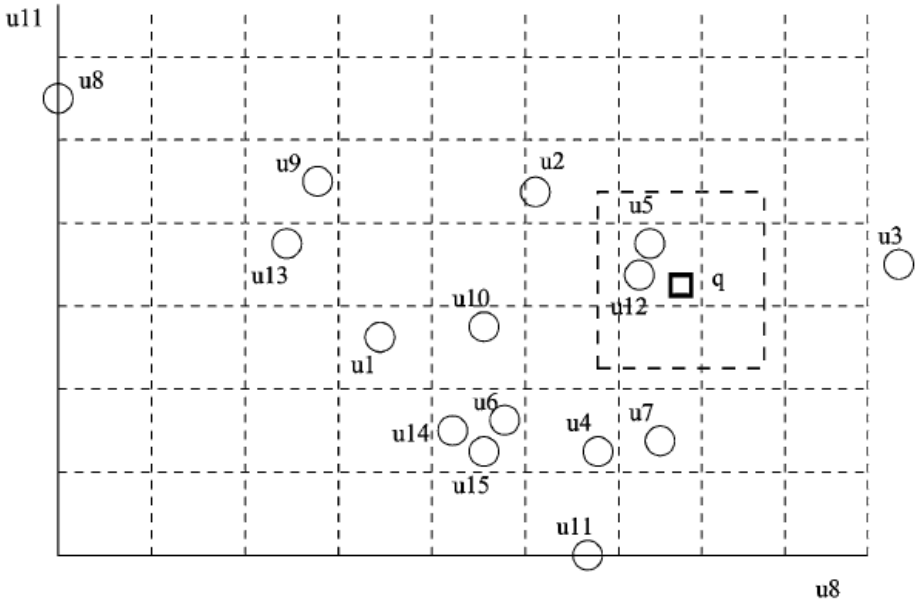
$$\mathrm{L}_{\max}(v_{\mathcal{P}}(q), v_{\mathcal{P}}(u_i)) > r$$

- En búsqueda k-NN considerar *r* como la distancia al k-ésimo candidato

Espacio de pivotes

- Convertir espacio métrico al espacio de pivotes





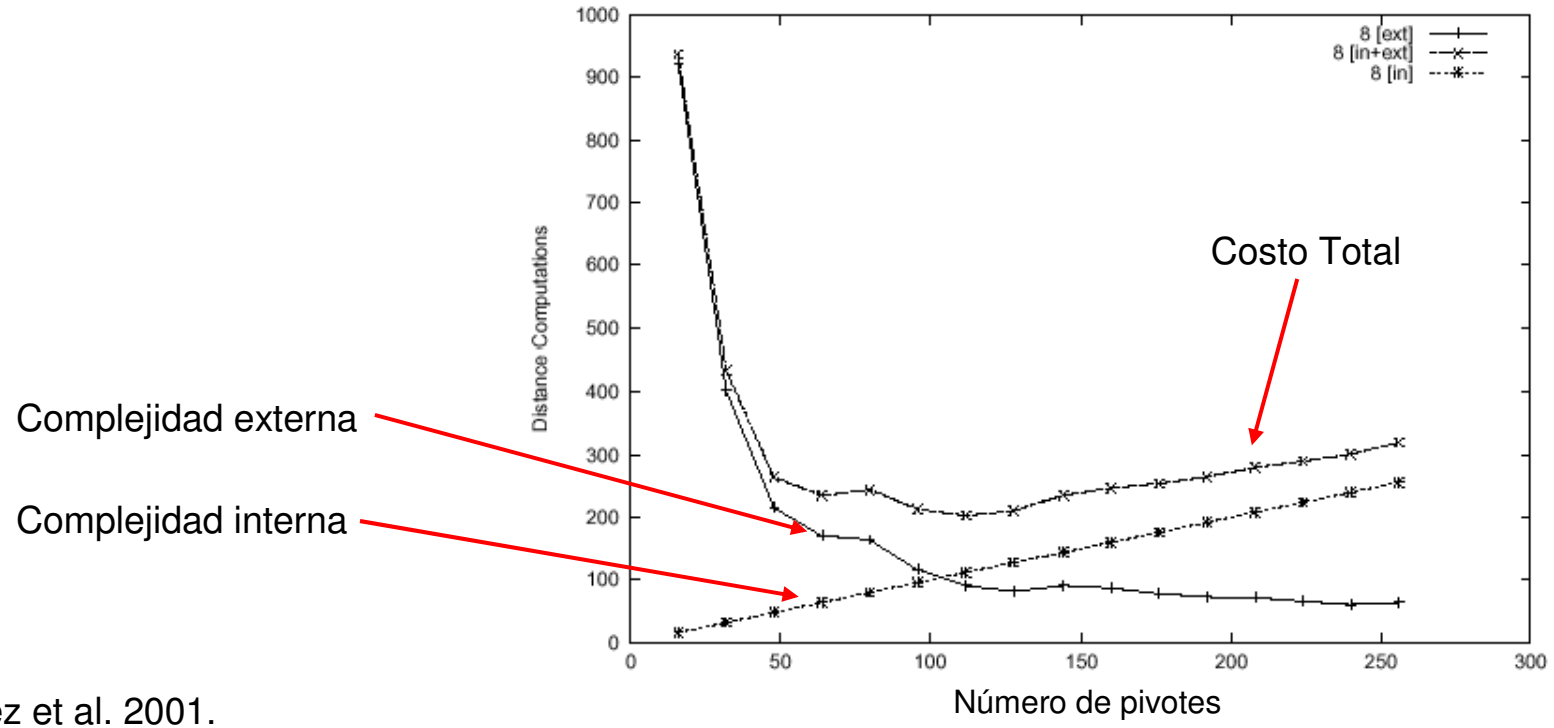
Ver Chavez et al. 2001.

Complejidad Interna y Externa

- Complejidad externa:
  - Cómputos de distancia entre *q* y objetos no descartados
- Complejidad interna:
  - Cómputos de distancia entre *q* y pivotes
  - Cómputos de *LB* entre *q* y todos los objetos
- Al aumentar el número de pivotes:
  - Aumenta la complejidad interna
  - Disminuye la complejidad externa
- Existe un número óptimo de pivotes
  - Comparar performance del óptimo contra no usar índice

Complejidad versus Pivotes

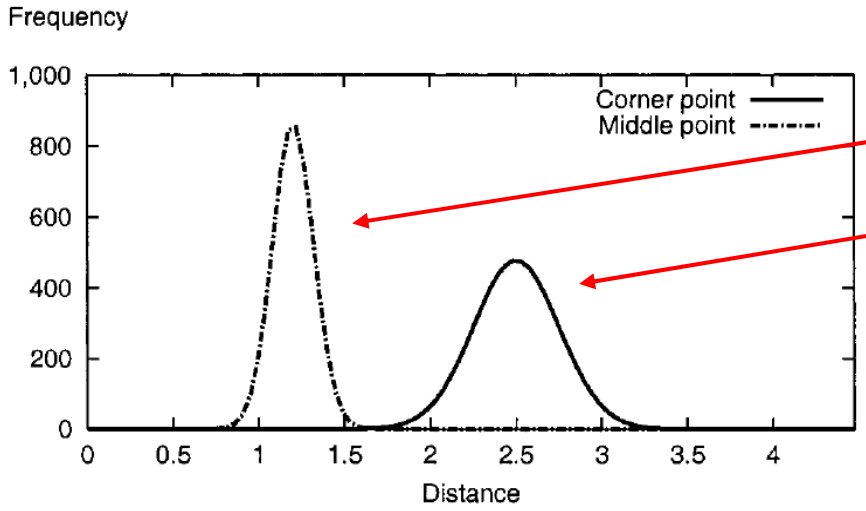
- Al aumentar el número de pivotes:
  - Aumenta el costo de evaluar d(q,p) y LB(q,u) (linealmente)
  - Disminuye la cantidad de veces que se evalúa d(q,u) (no lineal)



Ver Chavez et al. 2001.

Selección de pivotes

- Dependiendo del dataset, hay objetos que son mejores pivotes que otros
  - Mejor pivote → Descarta más distancias → Cotas inferiores lo más altas posible
- Ejemplo: si tenemos datos en un cubo unitario de 20 dimensiones:



Distancias entre los datos y el punto central

Distancias entre los datos y una esquina

Ver Zezula cap 10.5

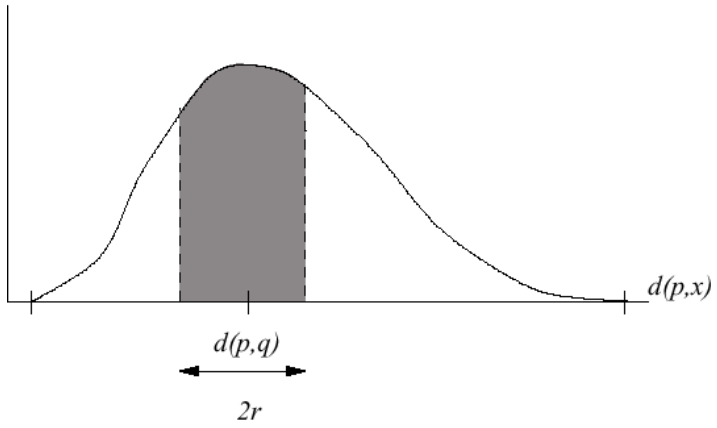
Selección de pivotes

- Una baja varianza en el histograma de distancias implica que al restar la distancia entre dos objetos probablemente será cercano a cero
  - El punto central es un mal pivote porque todos los objetos están casi a una misma distancia de él
  - Puntos en la esquinas obtienen una mayor varianza en las distancias
- Sin embargo, en un espacio métrico genérico no hay geometría!

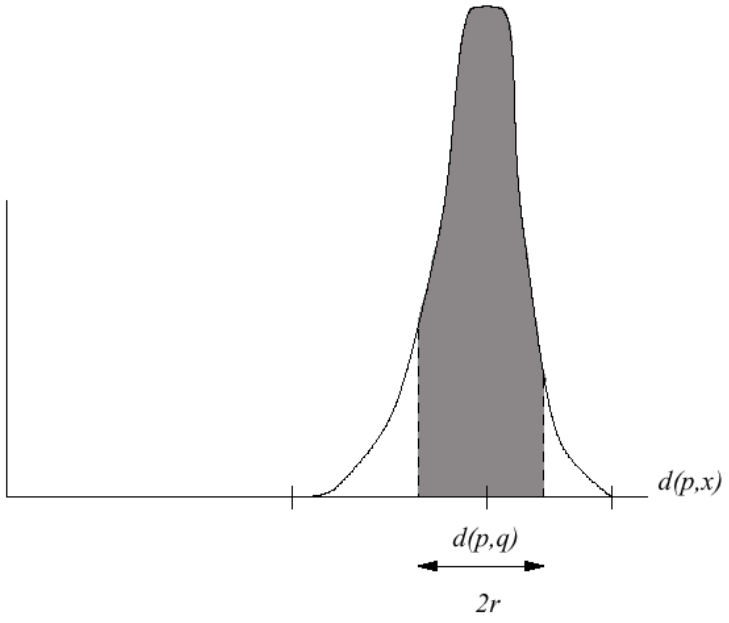
Dimensión Intrínseca

- Concepto de alta dimensión en espacios métricos:

$$\rho = \frac{\mu^2}{2\sigma^2}$$



ρ bajo



ρ alto

Ver Chavez et al. 2001.

Selección de pivotes

- **Método de selección 1**:
  - Definir un número de pivotes y escogerlos al azar
- Existen muchos posibles conjuntos de pivotes
  - ¿Se obtendrá el mismo rendimiento al usar cualquier conjunto de pivotes al azar?
  - ¿Existirán mejores o peores conjuntos de pivotes?
- Se necesita alguna medida que permita estimar la performance que logrará un conjunto de pivotes para búsquedas futuras
  - Durante la creación del índice no se conocen los objetos de consulta
  - Se asume que las consultas tendrán una distribución similar a los datos conocidos

Criterio de Evaluación

- Un buen conjunto de pivotes *P* debe calcular una cota inferior lo más cercana a la distancia real
  - Los valores que entregue LB_P deben ser cercanos a los valores de *d*
- Definamos $\Delta P(x,y) = d(x,y) - LB_P(x,y)$
- Sea $\mu_{\Delta P}$ el promedio de $\Delta P(x,y)$ para todo par x, y en ***R***
- Dados *N* conjuntos de pivotes, se debe escoger el conjunto que obtiene un menor $\mu_{\Delta P}$
- Notar que d(x,y) es fijo para los *N* conjuntos que se están evaluando
- Por tanto es equivalente a escoger el conjunto de pivotes que obtenga un mayor valor promedio de $LB_P(x,y)$

Algoritmo de Evaluación

- Dados N conjuntos de pivotes con *k* pivotes cada uno
- Elegir al azar *m* pares de objetos (a_i,b_i)
- Para cada conjunto de pivotes *P*:
  - Calcular y promediar los m valores $LB_P(a_i,b_i)$
- Escoger el conjunto que logra el mayor valor promedio de LB_P
- Costo: *N*2*m*k* cálculos de distancia

Selección de pivotes

- **Método de selección 2**:
  - Dado un parámetro de distancia mínima crear una “zona de exclusión” alrededor de cada pivote
- Evita seleccionar pivotes cercanos y prefier pivotes que están lejos entre sí
  - Dos pivotes muy cercanos entre sí no mejoran mucho el valor de LB

Sparse Spatial Selection (SSS)

- Realizar un recorrido aleatorio de los objetos y elegir objetos distantes entre sí
- Parámetro de exclusión $M\alpha$:
  - M: máxima distancia en el espacio
  - α: factor (típicamente 0.4)

$$
\begin{array}{l}
PIVOTS \leftarrow \{x_1\} \\
\textbf{for all } x_i \in \mathbb{U} \textbf{ do} \\
\quad \textbf{if } \forall\, p \in PIVOTS,\ d(x_i, p) \geq M\alpha \textbf{ then} \\
\quad\quad PIVOTS \leftarrow PIVOTS \cup \{x_i\} \\
\quad \textbf{end if} \\
\textbf{end for}
\end{array}
$$

Ver Pedreira et al. 2007

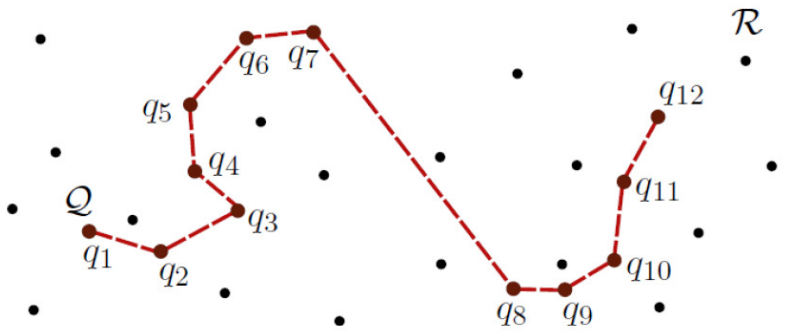
- Seleccionar distintos conjuntos reduciendo $M\alpha$ hasta obtener varios conjuntos con el tamaño deseado y luego quedarse con el mejor

Optimización de tabla de pivotes

- Se debe notar que el valor de LB_P finalmente depende de un solo pivote
  - Opción 1: En la tabla de pivotes, se puede ordenar cada fila para probar primero el mejor pivote de cada objeto (el más cercano o más lejano)
  - Opción 2: La tabla de pivotes se puede reducir a una sola columna, dejando sólo el mejor pivote por objeto
    - Reducir el espacio en memoria

Snake Table

- **Método de selección 3**:
  - Utilizar como pivote el objeto de consulta previo
  - Selección dinámica de pivotes (la tabla de pivotes se llena mientras se resuelven consultas)
  - Útil cuando se realizan varias consultas consecutivas (ej. frames de videos)



Búsqueda Aproximada con Pivotes

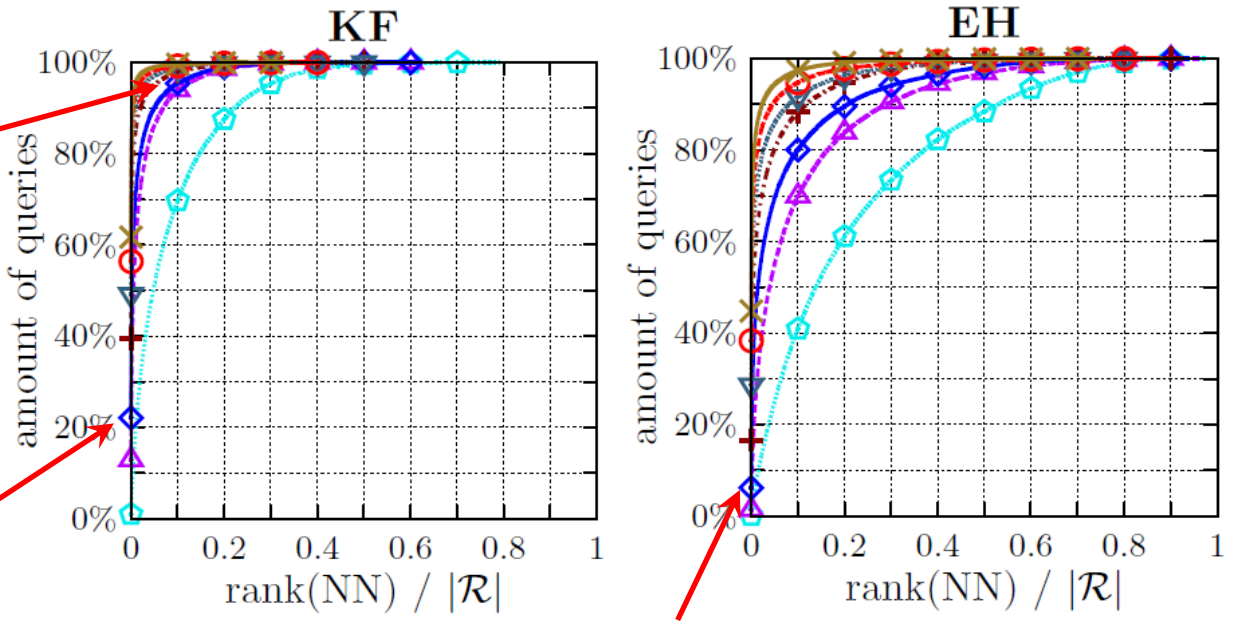
- La función LB_P puede ser usada como una estimación rápida de la distancia real
- Parámetro de aproximación T (entre 0 y 1):
  - Calcular LB_P para todos los objetos y seleccionar los $T\%$ menores valores
  - Calcular la búsqueda por rango o los k-NN solo entre los $T\%$ objetos seleccionados
    - La distancia real d se evalúa solo para un $T\%$ de los objetos y los restantes son descartados
    - Para que sea más rápido que la búsqueda lineal, el tiempo de evaluar LB_P debe ser al menos T veces más rápido que d
    - Requisito: Los objetos u_i con menor $d(q,u_i)$ deben tener un valor bajo de $LB_P(q,u_i)$

Búsqueda Aproximada con Pivotes

- Distribución del valor de LB_P para los vecinos más cercanos (efectividad):

Para este dataset usando 5 pivotes, en un 92% de las consultas el objeto que era el NN tuvo un valor de LB_P dentro del 10% de menor → Si usamos T=10 un 92% de las veces obtendremos el NN correcto

Solo en un 21% de las consultas el objeto de menor LB_P era el NN

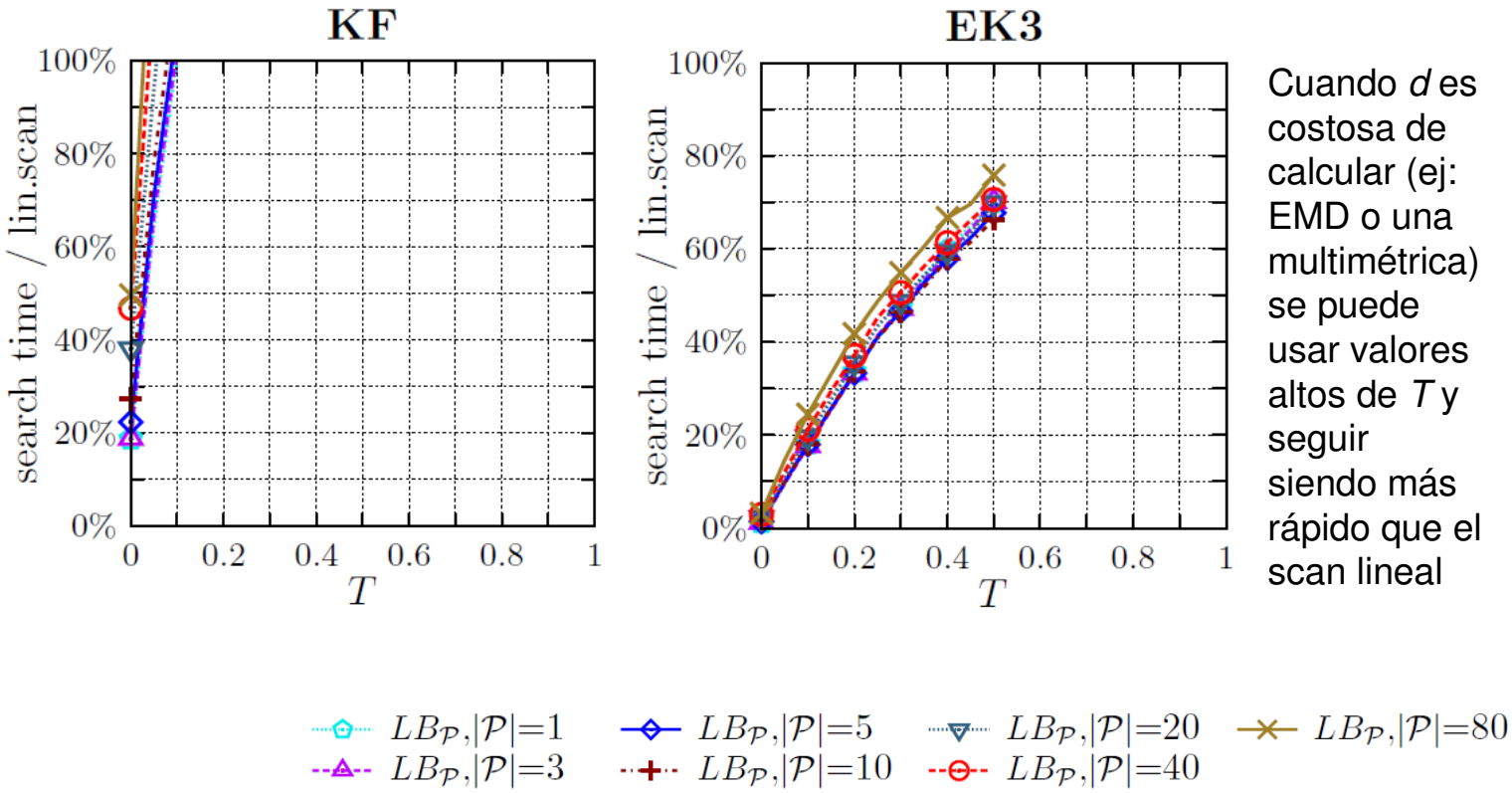


En este dataset usando 5 pivotes, sólo un 6% de las veces el de menor LB_P fue también el NN y un 80% de las veces estuvo dentro del 10% menor

Búsqueda Aproximada con Pivotes

- Reducción de los tiempos de búsqueda:

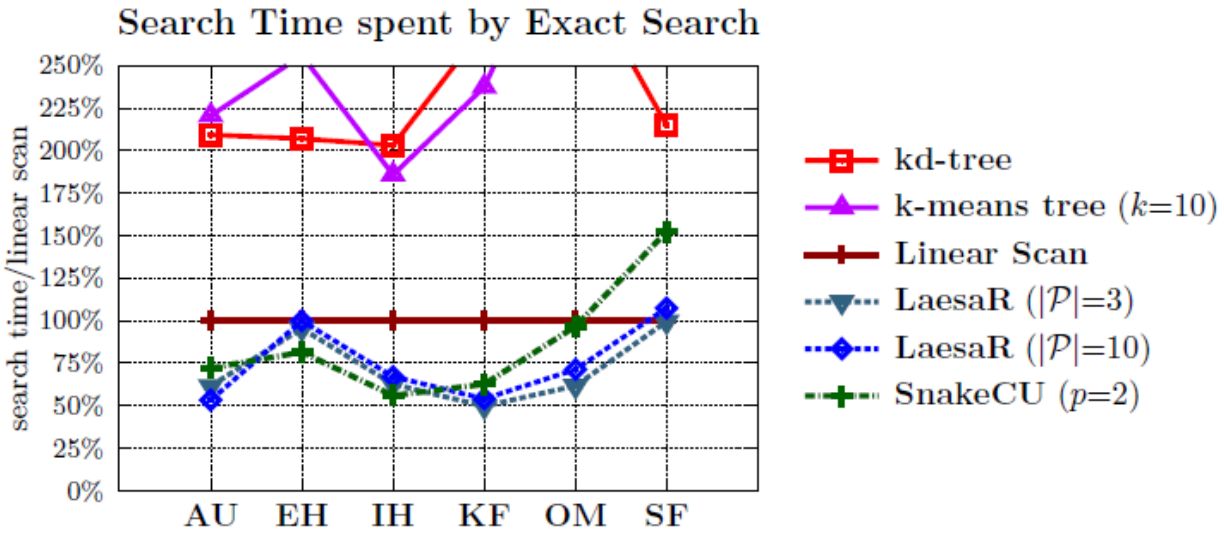
Cuando *d* es muy rápida de calcular (ej: L_1) el valor de *T* no puede ser muy alto si no la búsqueda aproximada se vuelve más lenta que el scan lineal



Cuando *d* es costosa de calcular (ej: EMD o una multimétrica) se puede usar valores altos de *T* y seguir siendo más rápido que el scan lineal

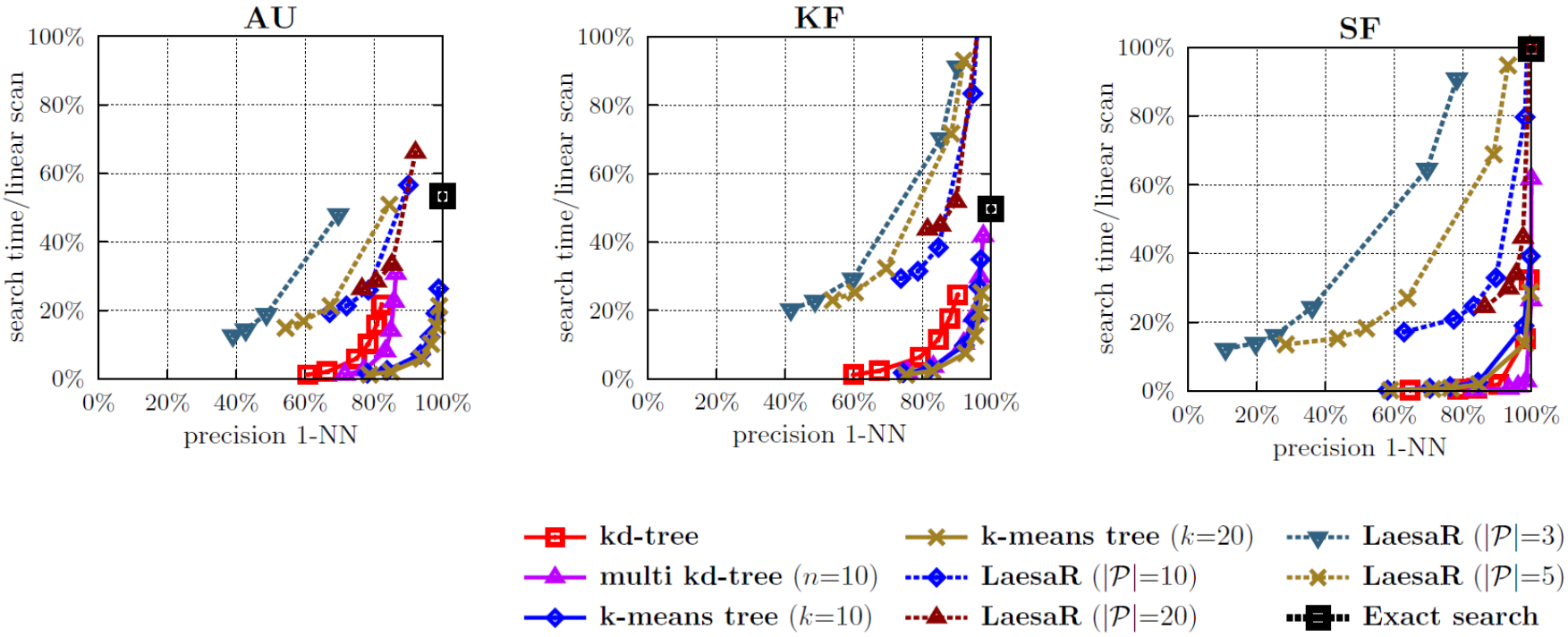
Indices Métricos vs Multidimensional

- Búsqueda exacta en distintos conjuntos:
  - Índices multidimensionales usualmente más lentos que búsqueda lineal
  - Índices métricos usualmente más rápidos que la búsqueda lineal



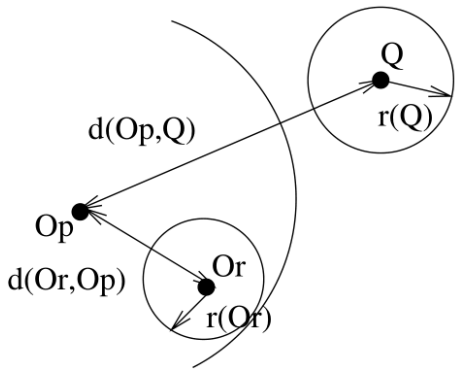
Indices Métricos vs Multidimensional

- Búsqueda aproximada en distintos conjuntos:
  - Índices multidimensionales logran un mucho mejor balance de efectividad vs tiempo de búsqueda



Árboles Métricos

- Los objetos se pueden agrupar en una jerárquía de zonas
  - Búsquedas por similitud calculan la intersección entre la bola de consulta y el radio de una zona (criterio del radio cobertor)
  - Búsquedas por rango y del NN usando MINDIST y cola de prioridad
- Ejemplos:
  - M-tree (Ciaccia et al. 1997)
  - Voronoi Tree (Dehne et al. 1987)
  - Vantage Point Tree (Yianilos. 1993)
  - GNAT (Brin. 1995)
  - List of Clusters (Chávez et al. 2005)



Espacios Multimétricos

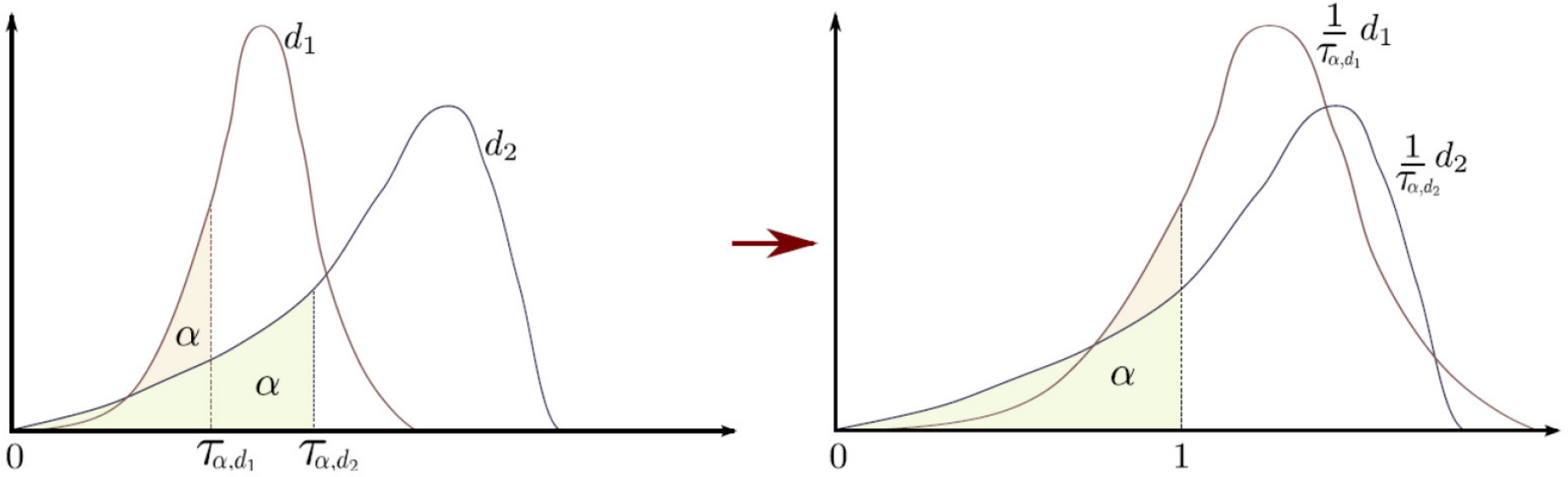
Combinación de distancias

- Sean δ_1, ... δ_m diferentes métricas para un mismo universo de objetos
  - Se puede definir una nueva función de distancia como la combinación lineal de las *m* métricas, es decir, sumar cada distancia ponderada por un peso w_i:

$$\Delta(q,o) = \sum_{i=1}^{m} w_i \cdot \frac{\delta_i(q,o)}{normFactor_i}$$

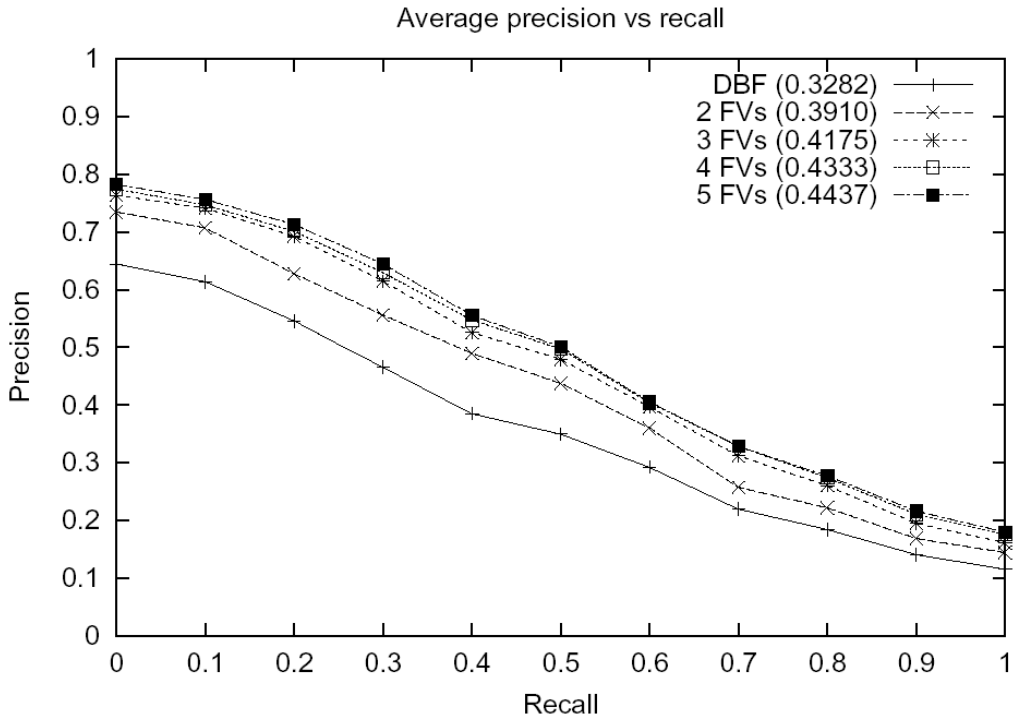
Normalización

- Normalizar por la distancia máxima o por una distancia de probabilidad α



Combinación de distancias

- Combinando más distancias (i.e. usando más descriptores) usualmente se mejora la calidad de la respuesta

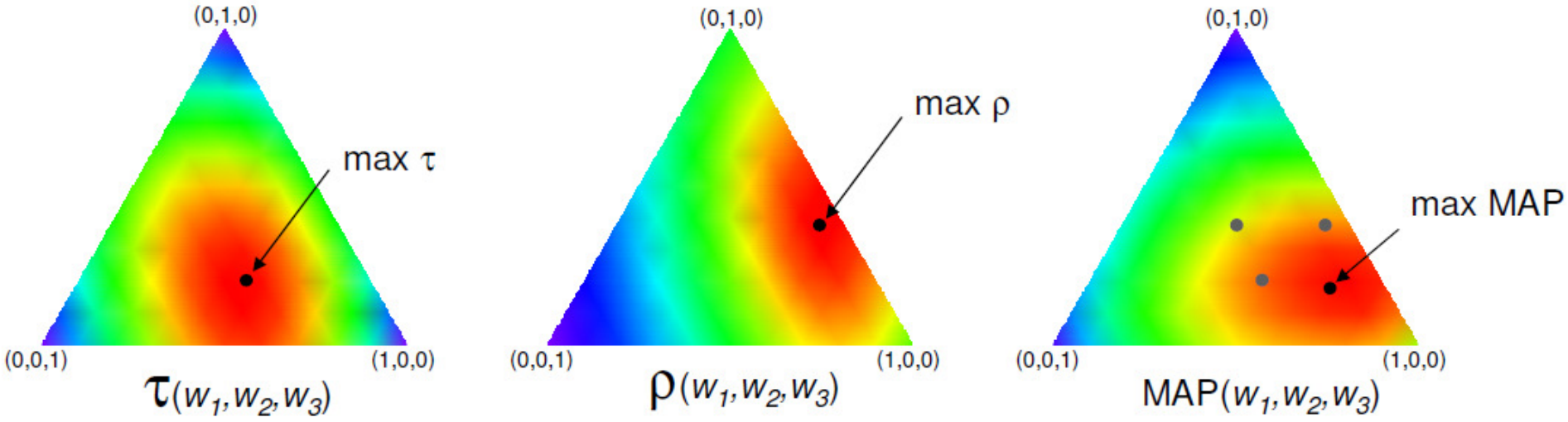


Combinación de distancias

- Al combinar funciones de distancia se construye una nueva función que (usualmente) logra mejor efectividad
- Pesos estáticos: función con pesos fijos
  - La distancia combinada también es métrica
  - Índices pueden indexar la distancia combinada
- Pesos dinámicos: función puede cambiar sus pesos dependiendo del objeto de consulta
  - Usualmente logra mejor efectividad que pesos fijos
  - La distancia combinada no es métrica
  - Se deben indexar las distancias por separado

Pesos estáticos

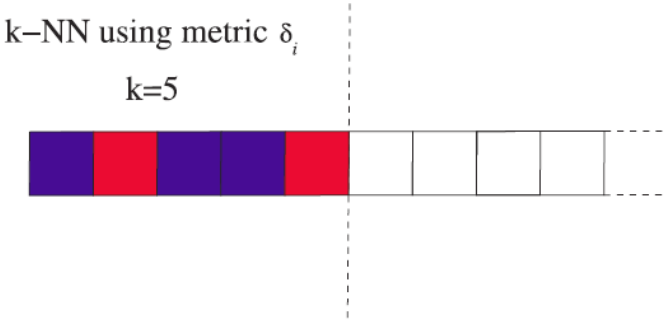
- Cálculo automático de pesos estáticos:



Pesos dinámicos

- Entropy Impurity

I. Perform k-NN in training dataset



Three objects belong to the blue class and two objects belong to the red class.

II. Entropy impurity

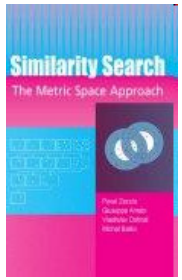
P_{ω_i} : fraction of objects that belong to model class i

$$entropy(\delta_i) = -\sum_{i=1}^{|\#classes|} \begin{cases} P_{\omega_i} \cdot \log_2(P_{\omega_i}) & \text{if } P_{\omega_i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

The entropy impurity of metric δ_i is equal to 0 if all objects belong to the same class, and has a maximum value (log(*k*)) if each object belongs to a different class.

Bibliografía

- **Similarity Search: The Metric Space Approach**. Zezula et al. 2006.
  - Capítulo 1, Secciones 1-4.



Papers

- Vidal. **An algorithm for finding the nearest neighbours in (approximately) constant average time**. 1986.
- Dehne and Noltemeier. **Voronoi trees and clustering problems**. 1987.
- Yianilos. **Data structures and algorithms for nearest neighbor search in general metric spaces**. 1993.
- Micó, Oncina, and Vidal. **A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements**. 1994.
- Brin. **Near neighbor search in large metric spaces**. 1995
- Ciaccia, Patella, and Zezula. **M-tree: An Efficient Access Method for Similarity Search in Metric Spaces**. 1997.
- Chávez and Navarro. **A compact space decomposition for effective metric indexing**. 2005.
- Chávez, Navarro, Marroquín, and Baeza-Yates. **Searching in metric spaces**. 2001.
- Pedreira and Brisaboa. **Spatial Selection of Sparse Pivots for Similarity Search in Metric Spaces**. 2007.
- Bustos, Keim, Saupe, Schreck, and Vranic. **Automatic selection and combination of descriptors for effective 3D similarity search**. 2004.
- Barrios and Bustos. **Competitive content-based video copy detection using global descriptors**. 2013.
- Barrios, Bustos, and Skopal. **Analyzing and dynamically indexing the query set**. 2014.

Librerías

- Metric Space Library
  - http://www.sisap.org/metricspaceslibrary.html
- MetricKnn: Fast Similarity Search using the Metric Space Approach
  - https://juan.cl/metricknn_org/