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/ – 2019

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:
    - Positividad estricta $\forall x, y \in \mathbb{X},\ x \neq y \Rightarrow \delta(x, y) > 0$
    - Simetría $\forall x, y \in \mathbb{X},\ \delta(x, y) = \delta(y, x)$
    - Reflexividad $\forall x \in \mathbb{X},\ \delta(x, x) = 0$
    - Desigualdad triangular $\forall x, y, z \in \mathbb{X},\ \delta(x, z) \leq \delta(x, y) + \delta(y, z)$

Espacios Métricos

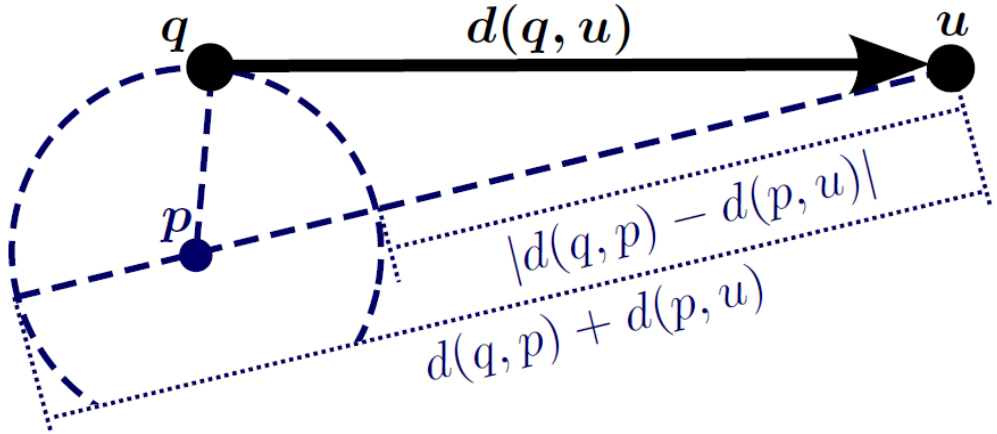
- Objetos de $\mathcal{D}$ son comparados utilizando la función de distancia
- La función d indica el grado de disimilitud entre dos objetos
- Ejemplo de espacio métrico:
  - Strings y distancia de edición
  - Vectores y una distancia de Minkowski L_p
  - Signatures y distancia EMD

Metric Access Methods

- Se tiene un conjunto de objetos *R*
- Un Metric Access Method (MAM) es una estructura de datos que permite resolver búsquedas por similitud en *R* reduciendo el número de veces que se evalúa la función de distancia *d*
  - Supuesto: la función de distancia es costosa de evaluar

Pivote

- Un pivote *p* es un objeto de la colección fijo
- Dados dos objetos *q* y *u*, si se conocen las distancias *d(q,p)* y *d(p,u)* entonces se puede estimar *d(q,u)*



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

Conjuntos de Pivotes

- El valor de la cota superior e inferior se acerca más al valor de *d* cuando se tienen más pivotes
- Para un conjunto de pivotes $\mathcal{P}$:
  - $UB_P(q,r)$ es la cota superior de $d(q,r)$:

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

  - $LB_P(q,r)$ es la cota inferior de $d(q,r)$:

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

Tablas de Pivotes

- AESA (Approximating and Eliminating Search Algorithm, 1986)
  - Usar todos los objetos de *R* como pivote
  - Requiere mantener una tabla de distancias entre todos los pares de objetos del dataset
    - Memoria $O(n^2)$
- LAESA (Linear AESA, 1994)
  - Seleccionar subconjunto de elementos de la colección como pivotes
  - ¿Cómo resolver una búsqueda eficientemente?
  - ¿Cómo seleccionar el conjunto de pivotes?

Creación de Tabla de Pivotes

- Para una colección de *n* objetos, se seleccionan *k* pivotes de la colección
- Calcular una tabla de $k \cdot n$ 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

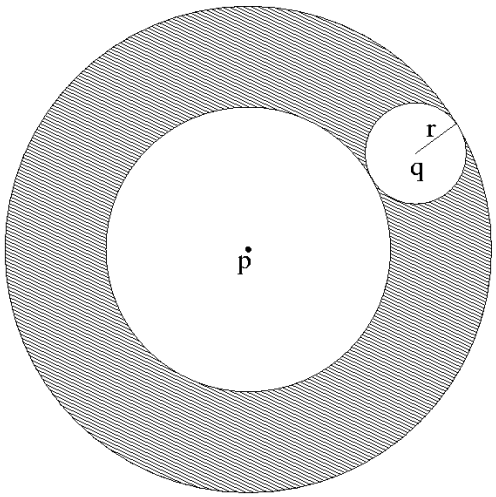
```
foreach p_i ∈ P do
    evaluar d(p_i, q) y guardar
end
queue ← ∅ ;
foreach u_i ∈ R do
    if LB_P(q, u_i) > r then
        continue;
    else if d(q, u_i) ≤ r then
        queue.Add(u_i) ;
    end
end
Print(queue);
```

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

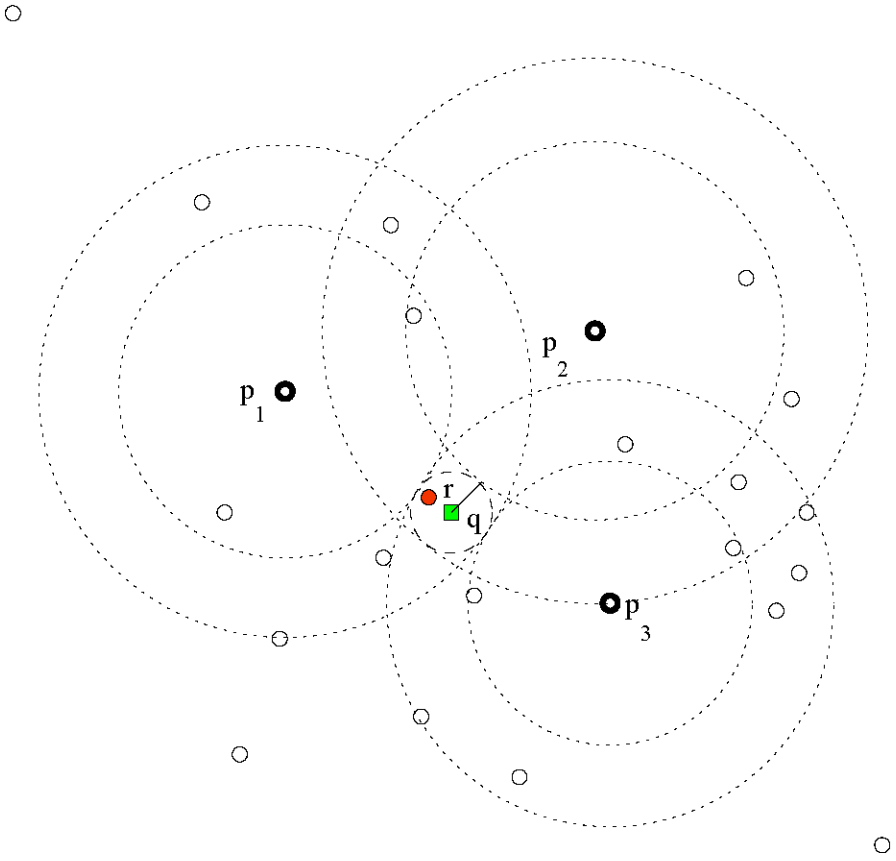


Tabla de Pivotes

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

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

Se tiene el siguiente conjunto de 16 descriptores (A-P) de 5 dims

Los descriptores se comparan 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 escoge al azar un conjunto de 3 pivotes (D, J, N)

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

Ejemplo

	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

Tabla de Pivotes

	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

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

Ejemplo

A	8	2	5	2	0
B	10	3	2	1	3
C	12	4	1	2	1
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
J	8	2	9	3	0
K	9	1	1	1	2
L	3	4	1	1	2
M	7	3	6	1	1
N	9	3	5	3	3
O	3	4	5	1	0
P	5	2	2	2	1
Q	**7**	**5**	**7**	**2**	**1**

Tabla de Pivotes

	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 todos los pivotes

Para cada elemento *x* se calcula su cota inferior LB:

$$\max_{p \in \mathcal{P}} \{|d(q,p) - d(p,x)|\}$$

Por cada pivote restar dos números de la Tabla de Pivotes y escoger la máxima diferencia

Si LB del objeto es mayor o igual a la distancia del candidato a NN entonces no puede ser relevante y se descarta

Ejemplo

A	8	2	5	2	0
B	10	3	2	1	3
C	12	4	1	2	1
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
J	8	2	9	3	0
K	9	1	1	1	2
L	3	4	1	1	2
M	7	3	6	1	1
N	9	3	5	3	3
O	3	4	5	1	0
P	5	2	2	2	1
Q	**7**	**5**	**7**	**2**	**1**

Tabla de Pivotes

	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 LB(q,..)	Distancia Real D(q,..)	Candidato
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~~	

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

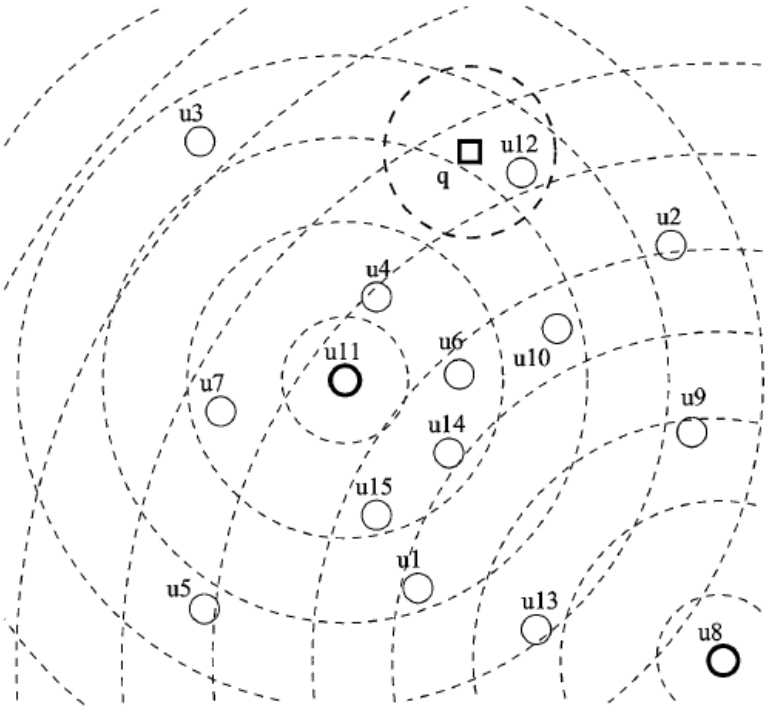
- Criterio de exclusión de la búsqueda por rango:

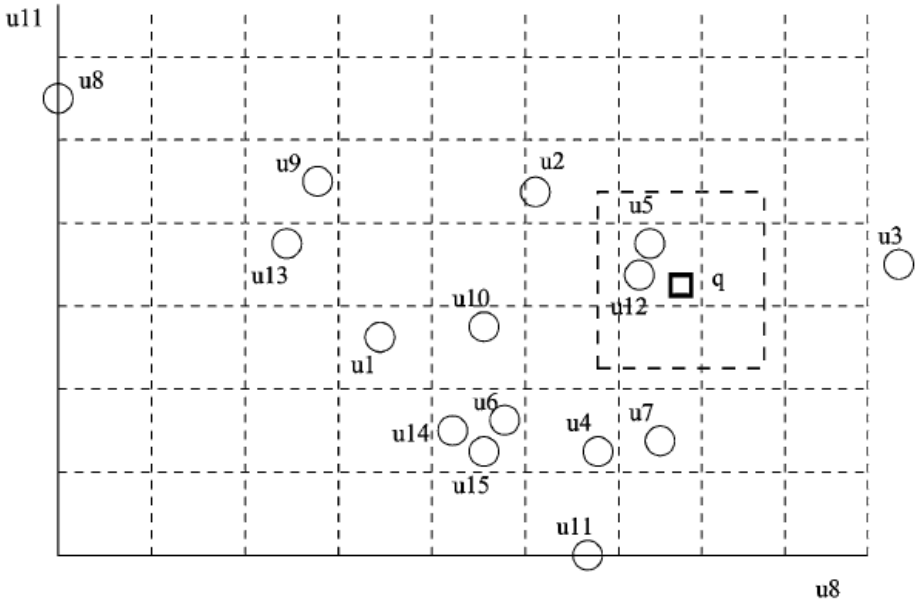
$$\mathrm{L}_{\max}(v_{\mathcal{P}}(q), v_{\mathcal{P}}(u_i)) > r$$

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

Espacio de pivotes

- Convertir espacio métrico al espacio de pivotes





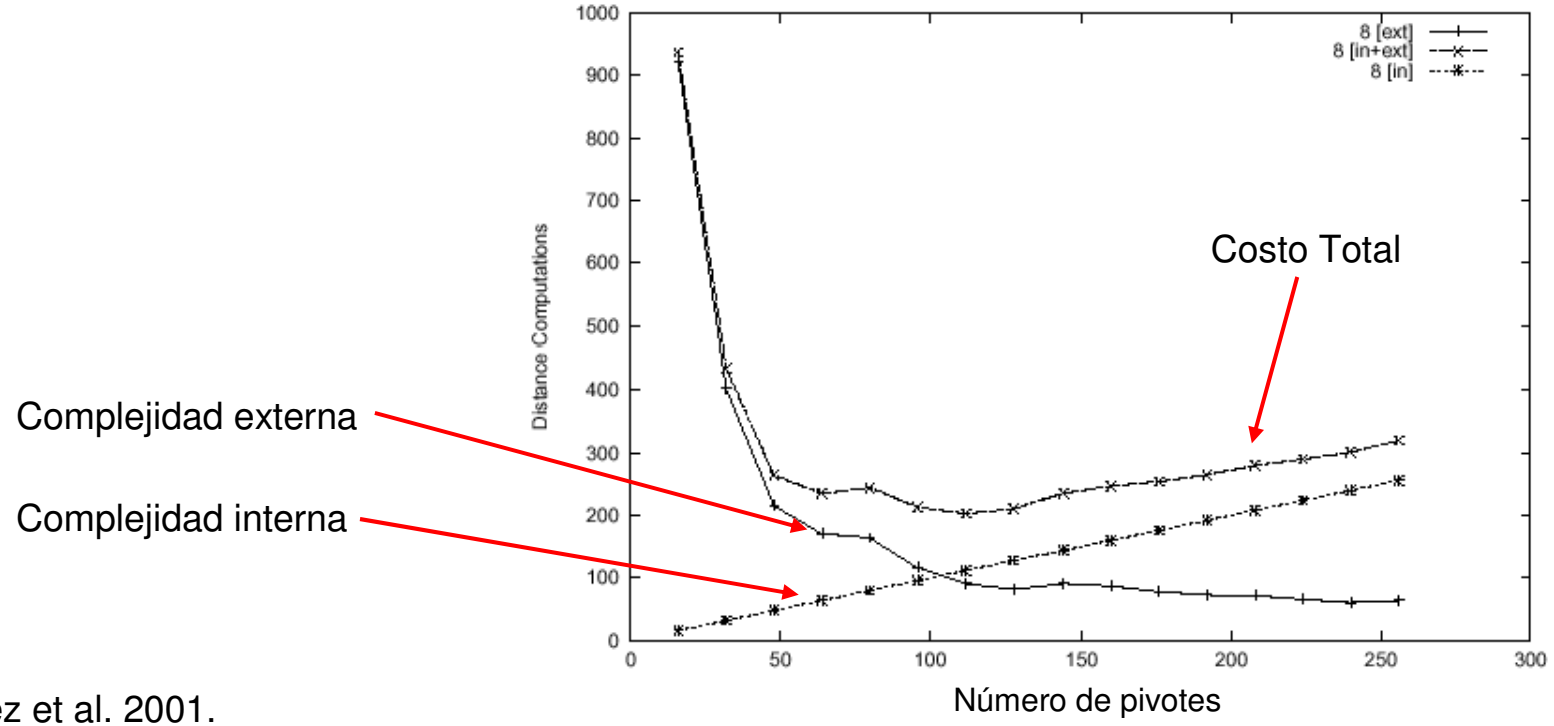
Ver Chavez et al. 2001.

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:
  - Disminuye la complejidad externa
  - Aumenta la complejidad interna (linealmente)
- Existe un número óptimo de pivotes
  - Comparar performance del óptimo contra no usar índice

Complejidad versus Pivotes

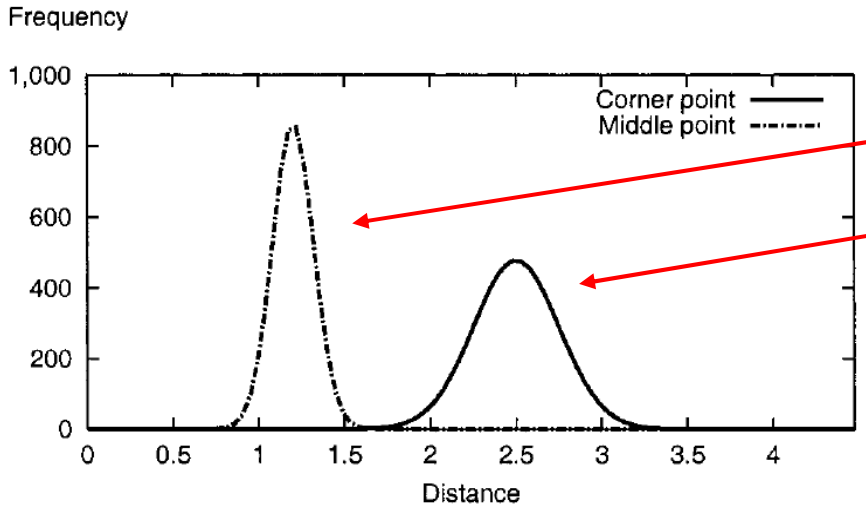
- Al aumentar el número de pivotes:
  - Disminuye la complejidad externa (se descartan más objetos)
  - Aumenta la complejidad interna (crece el trabajo realizado para descartar un objeto)



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

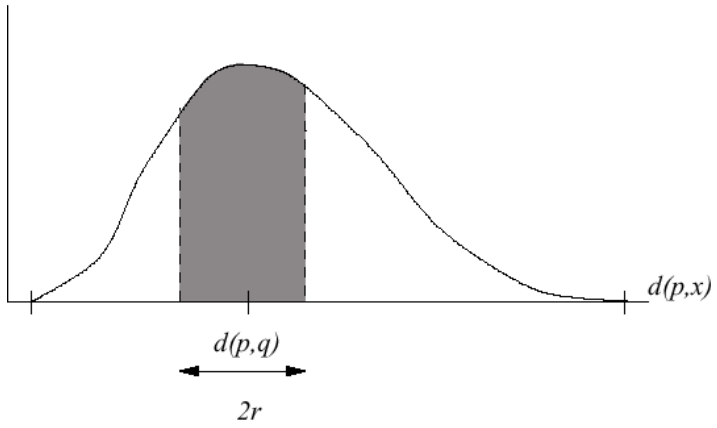
Selección de pivotes

- 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!

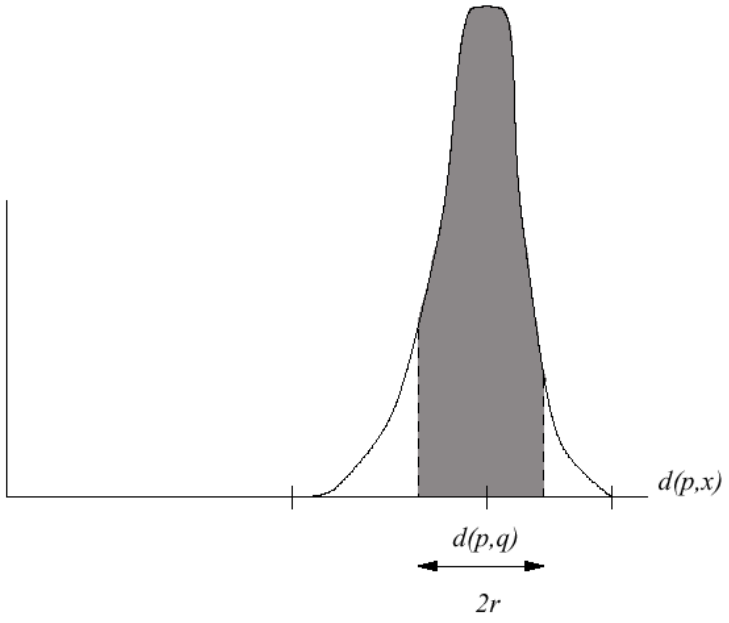
Dimensión Intrínseca

- 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: escoger pivotes en forma aleatoria
  - Definir un número de pivotes y escogerlos al azar
- ¿Se obtendrá el mismo rendimiento al usar cualquier conjunto de pivotes al azar?
- ¿Existirán mejores o peores conjuntos de pivotes?
- ¿Cómo escoger un conjunto de pivotes que logre la mejor performance en búsquedas?
- Notar que durante la creación del índice no se conocen los objetos de consulta
  - Asumir que las consultas tendrán una distribución similar a los datos conocidos

Evaluación de pivotes

- Criterio de evaluación: Un buen conjunto de pivotes *P* debe calcular una cota inferior lo más cercana a la distancia real
- Sea $\mu_{P\Delta}$ el promedio de la diferencia entre $LB_P(x,y)$ y $d(x,y)$ para todo x, y
- Si se tienen *N* conjuntos de pivotes, se debe escoger el conjunto que minimiza $\mu_{P\Delta}$

Evaluación de pivotes

- Elegir al azar *m* pares de puntos (a_i,b_i)
- Estimar $\mu_{P\Delta}$ para cada conjunto de pivotes *P:*
  - Para cada par (a_i,b_i) calcular $\Delta_i = |LB_P(a_i,b_i)-d(a_i,b_i)|$
  - El valor estimado de $\mu_{P\Delta}$ es el promedio de los *m* valores Δ_i
  - Pero el valor de $d(a_i,b_i)$ es el mismo para *N* los conjuntos de pivotes que se están evaluando!
- Finalmente, el criterio de evaluación consiste en escoger el conjunto *P* que maximice LB_P(a,b)
- Costo de selección para *N* conjuntos de k pivotes: *Nm2k* evaluaciones de la distancia

Selección de pivotes

- Idea:
  - Dos pivotes muy cercanos entre sí no mejoran mucho el valor de *LB*
  - Se deben evitar pivotes cercanos y preferir los pivotes que están lejos entre sí
- Método de selección 2:
  - Dado un parámetro de distancia mínima crear una "zona de exclusión" alrededor de cada pivote

Selección de pivotes

- SSS: Sparse Spatial Selection
  - Realizar un recorrido aleatorio de los objetos de la colección y elegir objetos distantes entre sí.
- Parámetro de exclusión $M\alpha$:
  - M: máxima distancia en el espacio
  - α: factor (típicamente 0.4)
- Seleccionar con SSS distintos conjuntos reduciendo $M\alpha$ hasta obtener varios conjuntos con el tamaño deseado y luego quedarse con el mejor

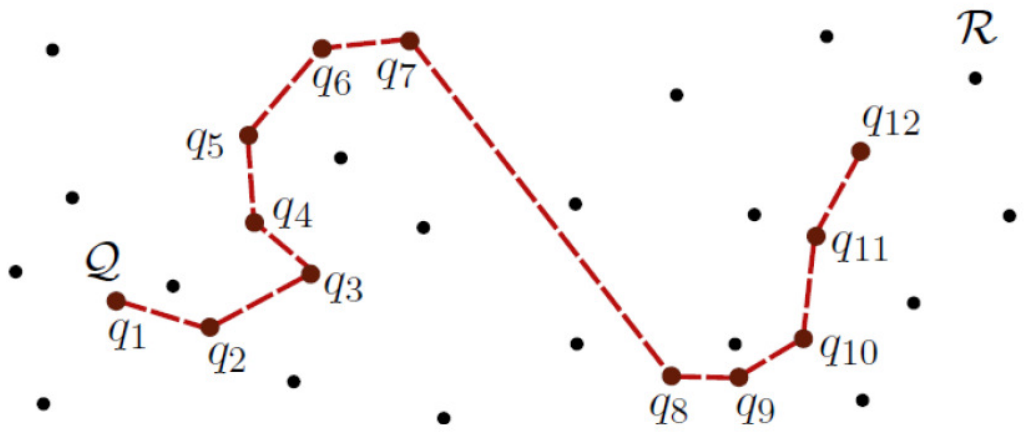
$PIVOTS \leftarrow \{x_1\}$
for all $x_i \in \mathbb{U}$ **do**
  if $\forall\, p \in PIVOTS,\ d(x_i, p) \geq M\alpha$ **then**
    $PIVOTS \leftarrow PIVOTS \cup \{x_i\}$
  end if
end for

Optimización de tabla de pivotes

- Se debe notar que el valor de *LB* 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

- Criterio de selección: Seleccionar pivotes en forma dinámica, según se resuelven consultas
  - Utilizar como pivote el objeto de consulta previo



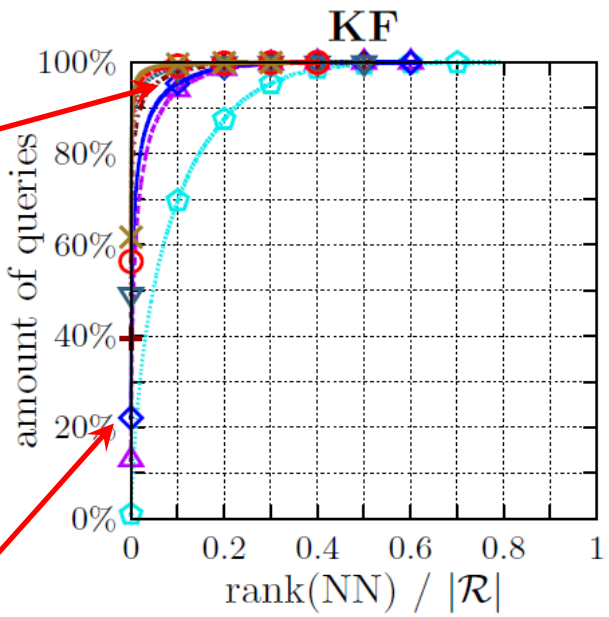
Búsqueda Aproximada con Pivotes

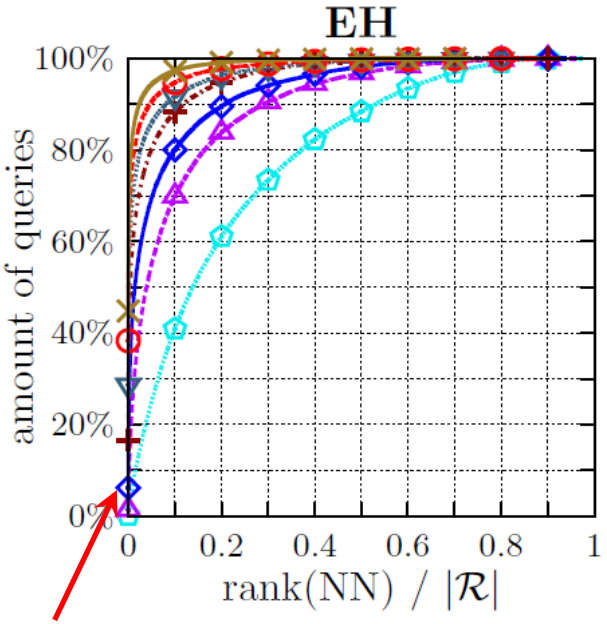
- Idea: La función LB_P puede ser usada como una estimación rápida de la distancia real
- Parámetro de aproximación T (entre 0 y 1):
  - Calcular LB_P para todos los objetos y seleccionar los $T\%$ menores valores
  - Calcular la búsqueda por rango o los k-NN solo entre esos los $T\%$ objetos seleccionados
    - 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





En un 21% de las consultas el objeto que era el NN fue también el de menor LB_P

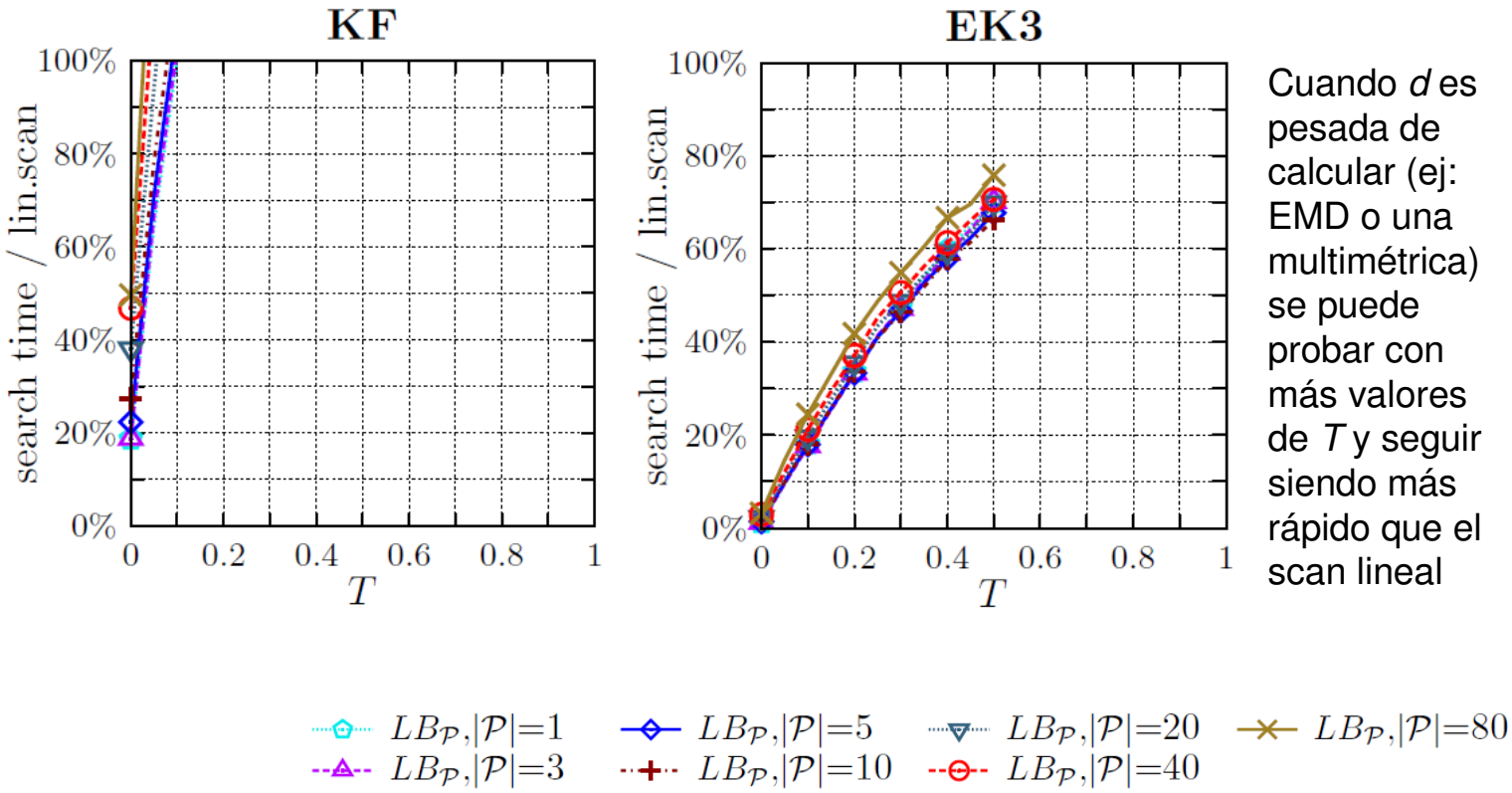
En este dataset usando 5 pivotes, sólo un 6% de las veces el NN fue también el de menor LB_P y un 80% de las veces estuvo dentro del 10% menor

$LB_{\mathcal{P}},|\mathcal{P}|=1$ $LB_{\mathcal{P}},|\mathcal{P}|=3$ $LB_{\mathcal{P}},|\mathcal{P}|=5$ $LB_{\mathcal{P}},|\mathcal{P}|=10$ $LB_{\mathcal{P}},|\mathcal{P}|=20$ $LB_{\mathcal{P}},|\mathcal{P}|=40$ $LB_{\mathcal{P}},|\mathcal{P}|=80$

Búsqueda Aproximada con Pivotes

- Reducción de los tiempos de búsqueda:

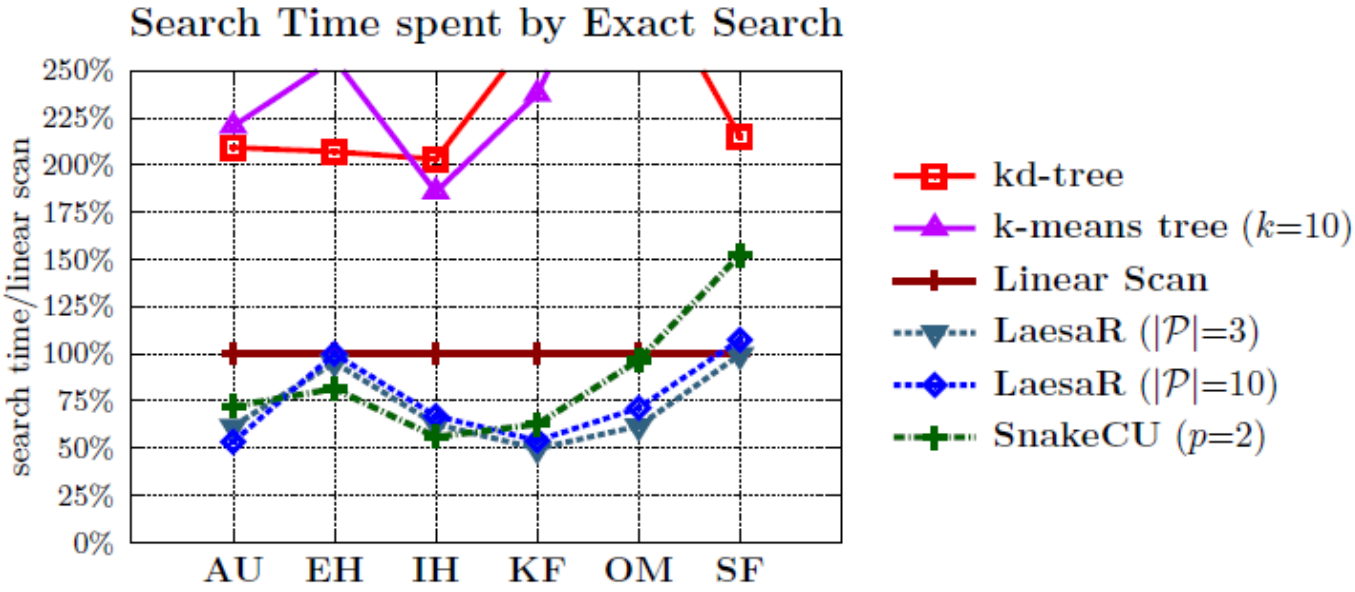
Cuando *d* es muy rápida de calcular (ej: L_1) el valor de *T* no puede ser muy alto si no la búsqueda aproximada se vuelve más lenta que el scan lineal



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

Indices Métricos vs Multidimensional

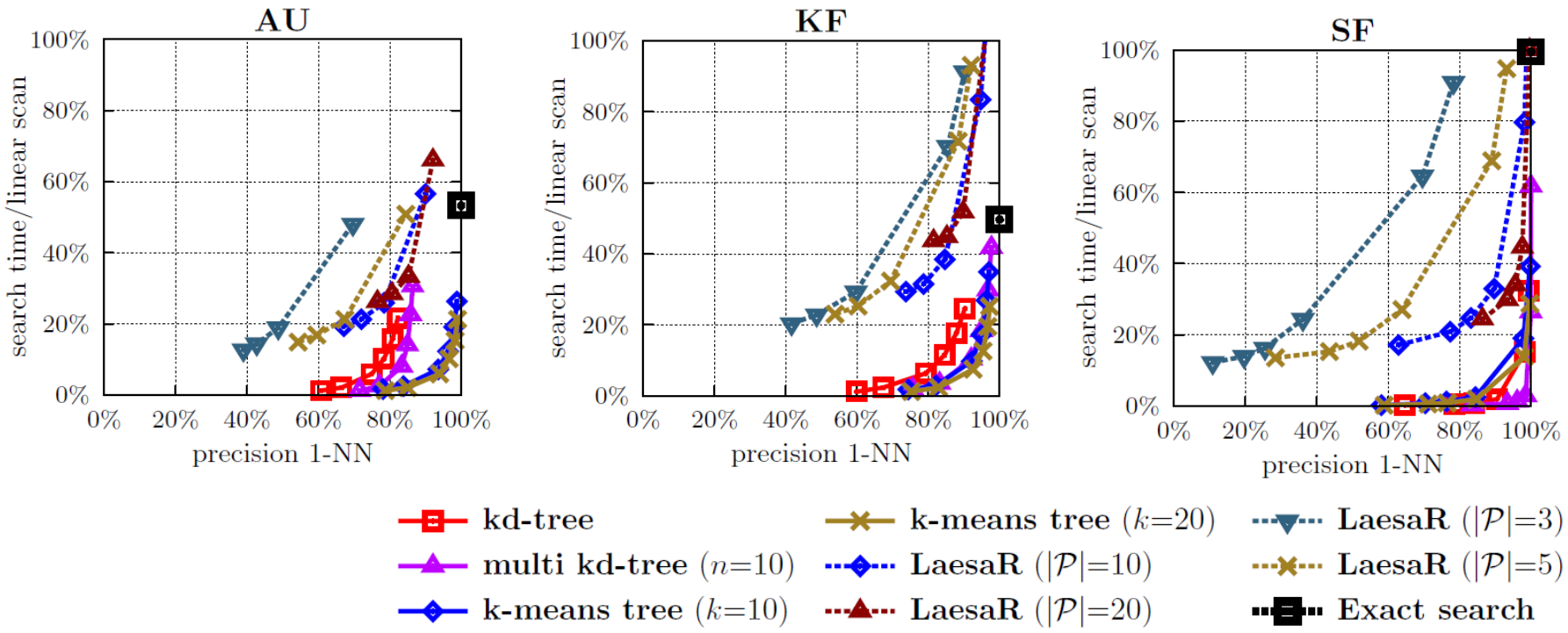
- Búsqueda Exacta:



Para búsqueda exacta, los índices métricos usualmente son más rápidos que la búsqueda lineal, en cambio los índices multidimensionales usualmente son más lentos que búsqueda lineal

Indices Métricos vs Multidimensional

- Búsqueda Aproximada:



Para búsqueda aproximada, los índices multidimensionales logran un mucho mejor balance de efectividad vs tiempo de búsqueda

ESPACIOS MULTIMÉTRICOS

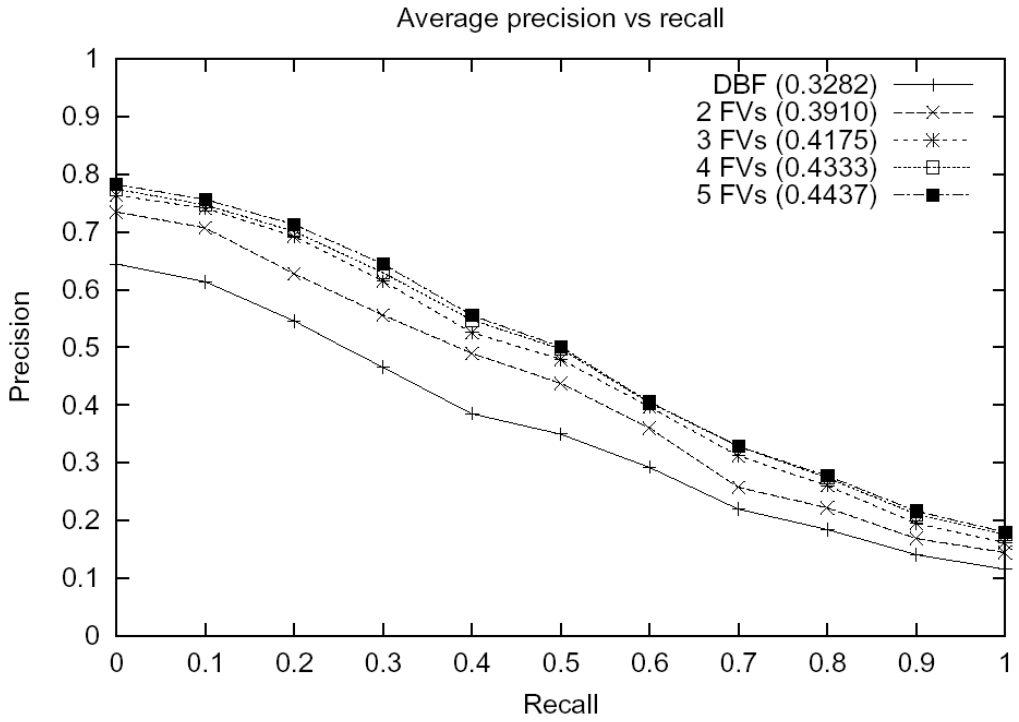
Combinación de distancias

- 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, sumando cada distancia multiplicada por un peso w_i:

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

Combinación de distancias

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

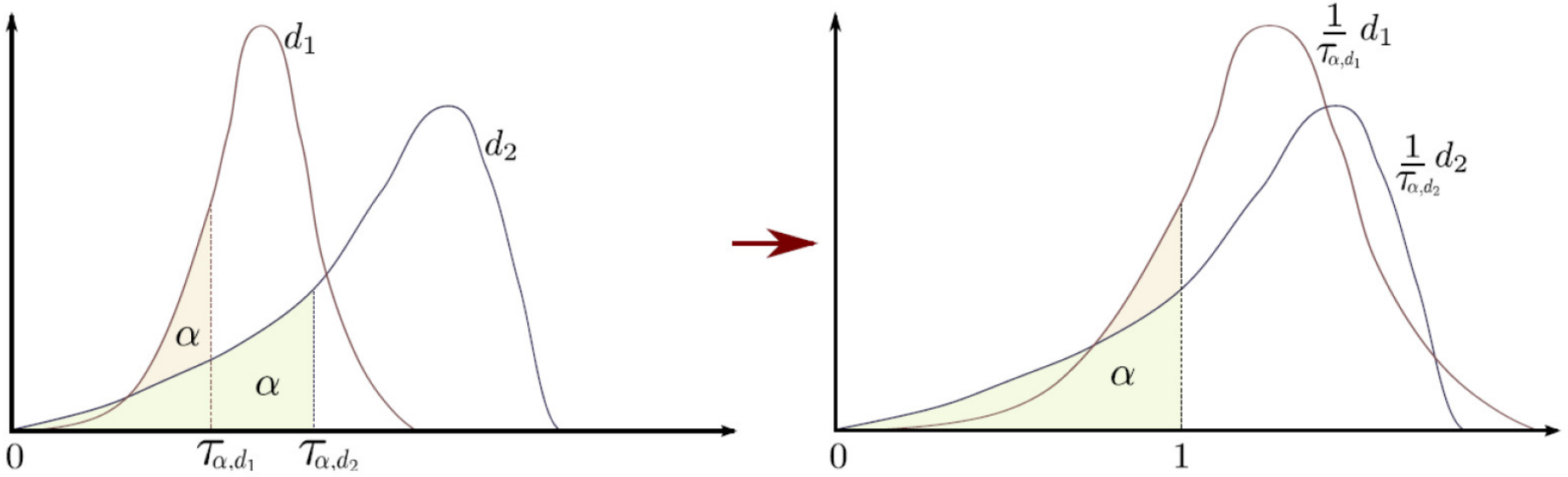


Combinación de distancias

- 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

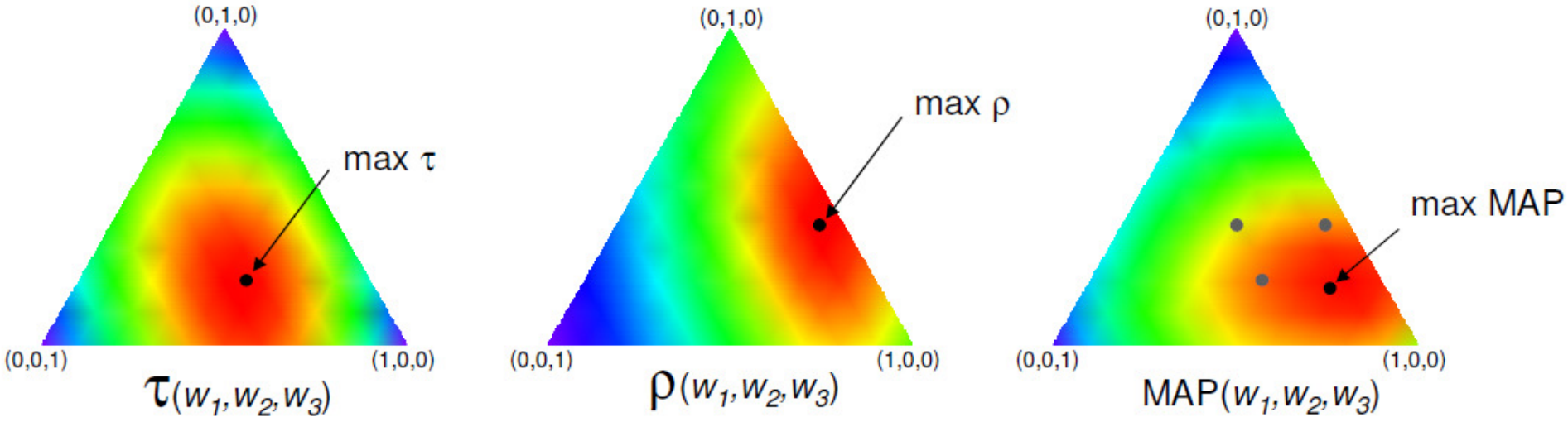
Normalización

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



Pesos estáticos

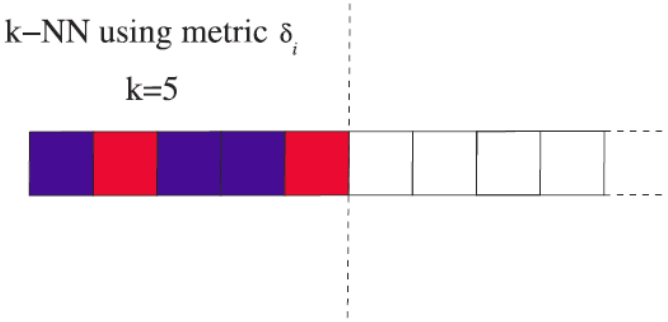
- Cálculo automático de pesos estáticos:



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

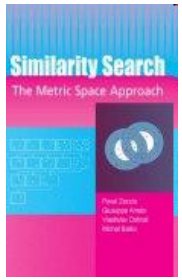
P_{ω_i} : fraction of objects that belong to model class i

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

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

- Chávez, Navarro, Marroquín, and Baeza-Yates. **Searching in metric spaces**. ACM Computing Surveys, 2001.
- Pedreira and Brisaboa. **Spatial Selection of Sparse Pivots for Similarity Search in Metric Spaces**. In SOFSEM, 2007.
- Bustos, Keim, Saupe, Schreck, and Vranic. **Automatic selection and combination of descriptors for effective 3D similarity search**. In ISMSE, 2004.
- Barrios, Bustos, and Skopal. **Analyzing and dynamically indexing the query set**. Information Systems, 2014.
- Barrios and Bustos. **Competitive content-based video copy detection using global descriptors**. Multimedia Tools and Applications, 2013.

Librerías

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