# 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*

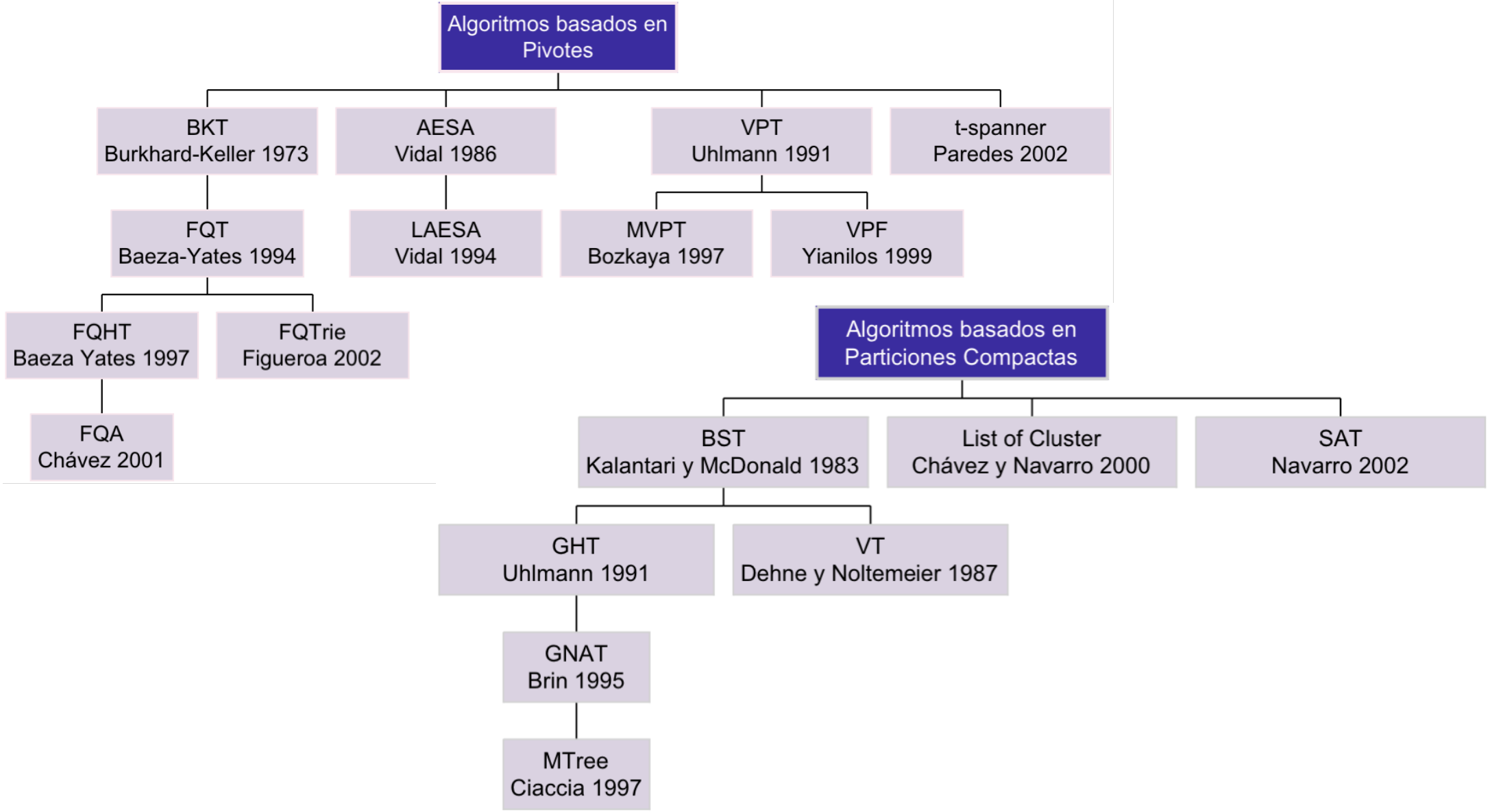
# Contenido del curso (1/2)

- 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.

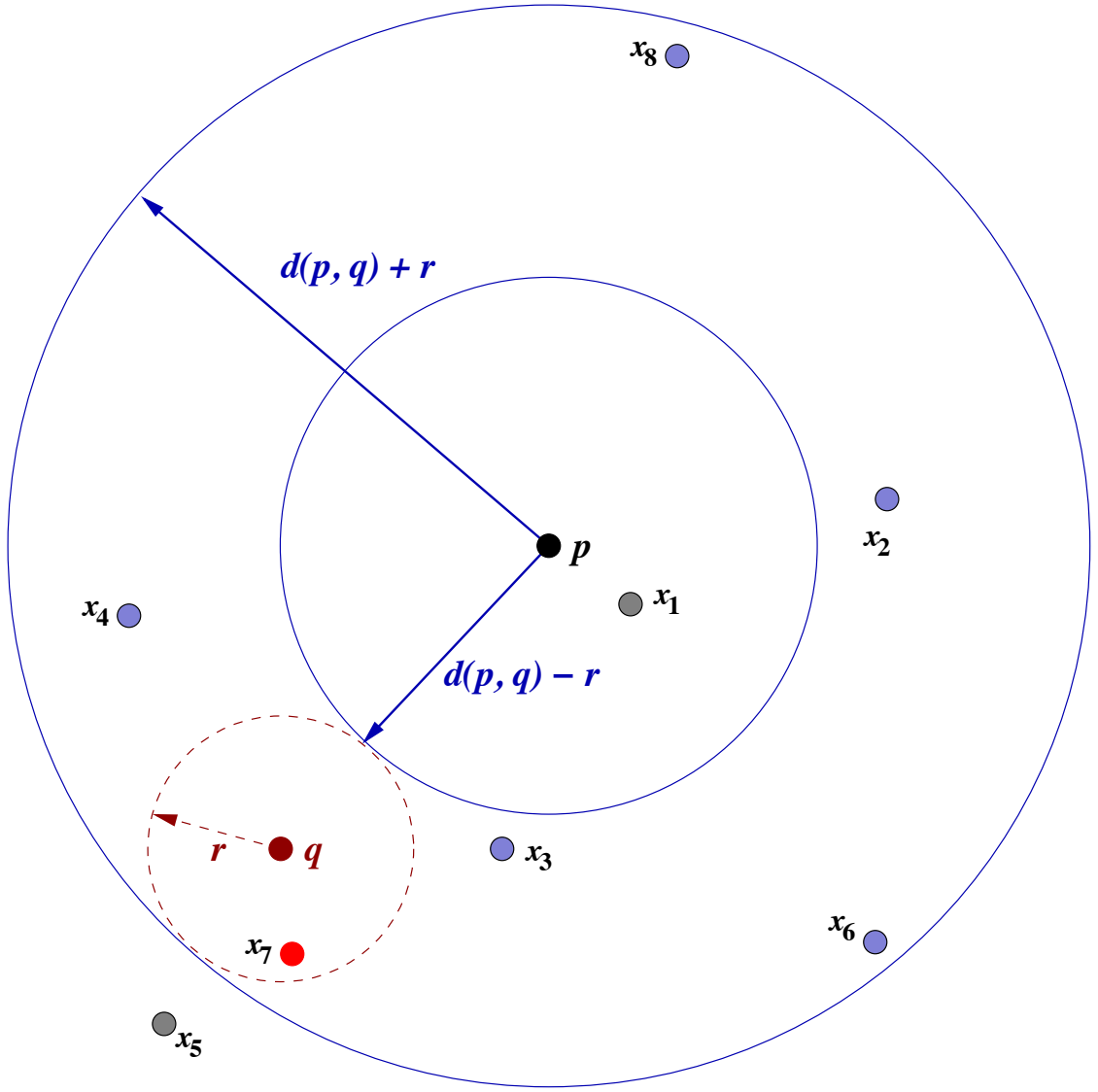
# Estado del arte



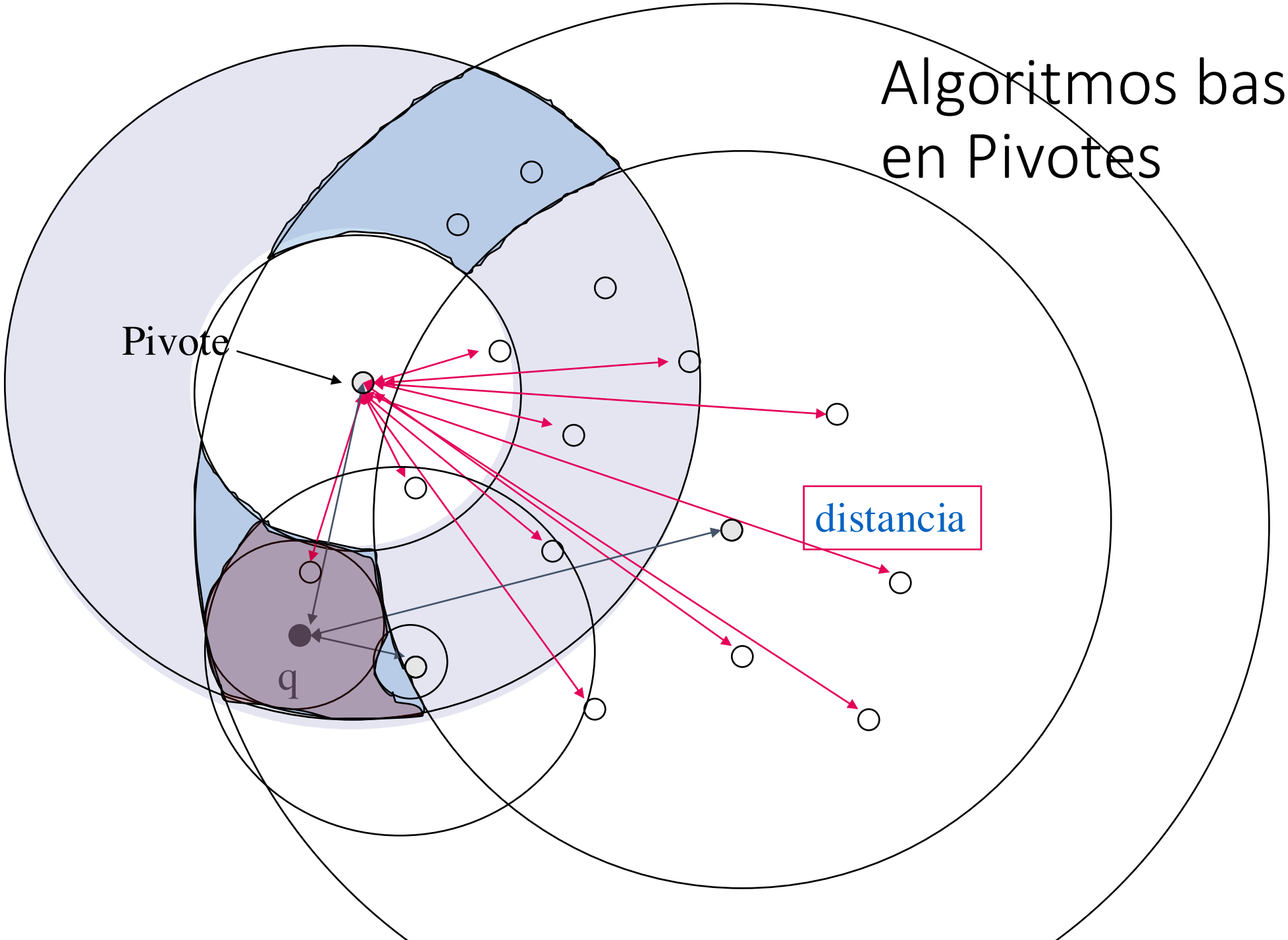
# Algoritmos basados en pivotes

- Cómo funcionan los pivotes?
- BKT
  - Burkhard Keller Tree
- FQT
  - Fixed Query Tree
- Vantage Point Tree
- Desempeño de los pivotes
- AESA

# Descarte utilizando un pivote.



# Algoritmos basados en Pivotes



# Definiciones

**Lema 1** *Dados tres objetos* $q \in \mathbb{X}$, $u \in \mathbb{U}, p \in \mathbb{P}$, *sabemos que* $|d(q,p) - d(p,u)| \leq d(q,u) \leq d(q,p) + d(p,u)$.

**Demostración** El límite superior se obtiene directamente de la desigualdad triangular,

$$d(q,u) \leq d(q,p) + d(p,u).$$

En el caso del límite inferior, de acuerdo a la desigualdad triangular, se tiene que:

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

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

Estas desigualdades implican $|d(p,u) - d(p,q)| \leq d(q,u)$. ∎

# BKT. Burkhard-Keller Tree

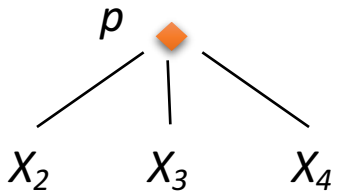
[Burkhard and Keller, 1973] Burkhard, W. A. and Keller, R. M. (1973). **Some approaches to best-match file searching**. *Communications of the ACM (CACM1973),* 16(4): 230-236. ACM Press.

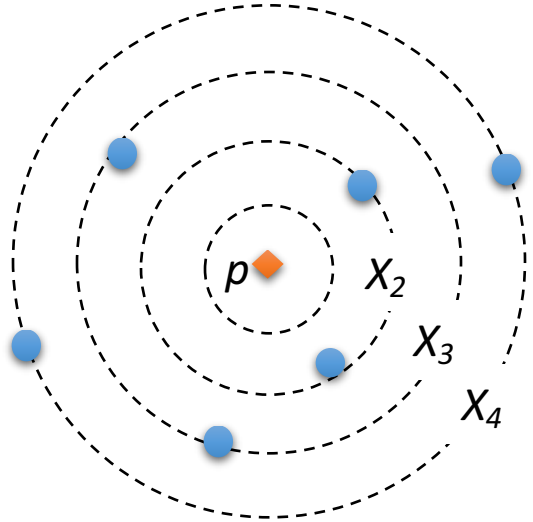
# Burkhard-Keller Tree (BKT) 1973

- Aplicable a distancias discretas o discretizadas
- Se divide recursivamente el conjunto $U$ en subconjuntos, usando en cada paso un pivote $p \in U$:

  $X_i = \{o \in U,\ d(o,p) = i\}$,  para cada distancia $i \geq 0$.

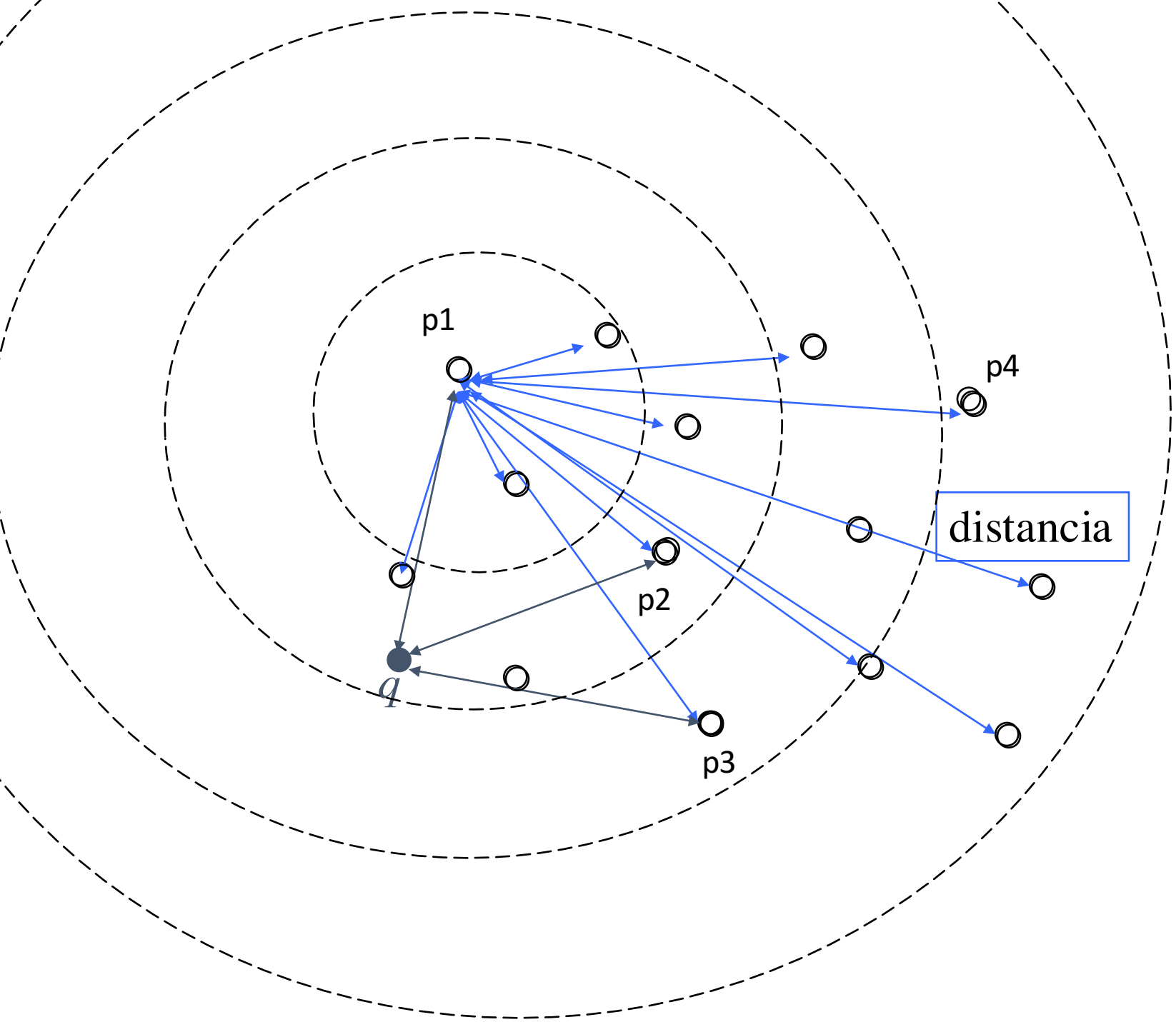
- Para cada $X_i$ se crea un subárbol de $p$, ignorando los subconjuntos vacíos.

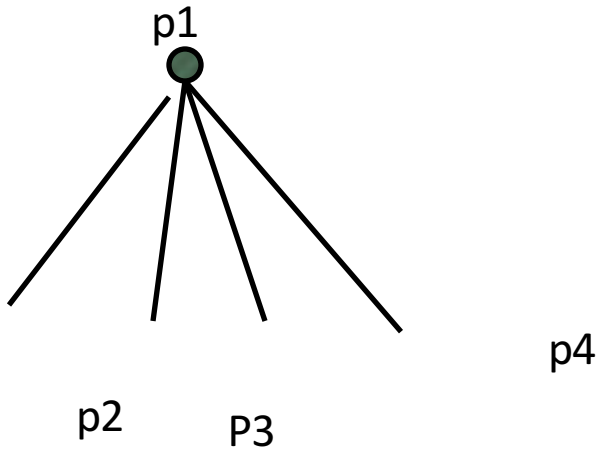


# BKT

## Burkhard Keller Tree



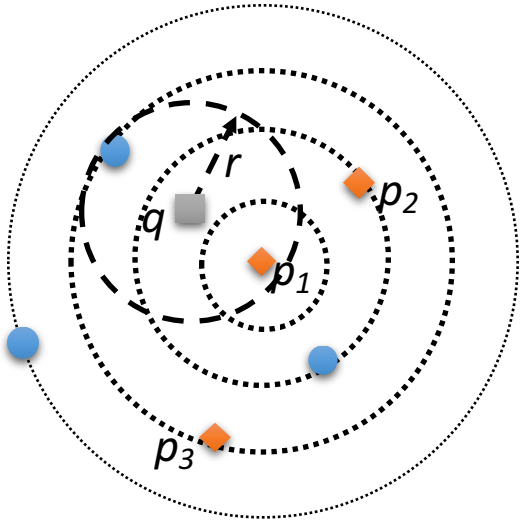



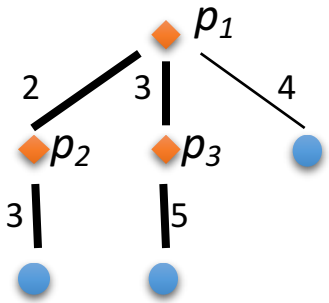

**Espacio *O(n)***

# BKT: Búsqueda

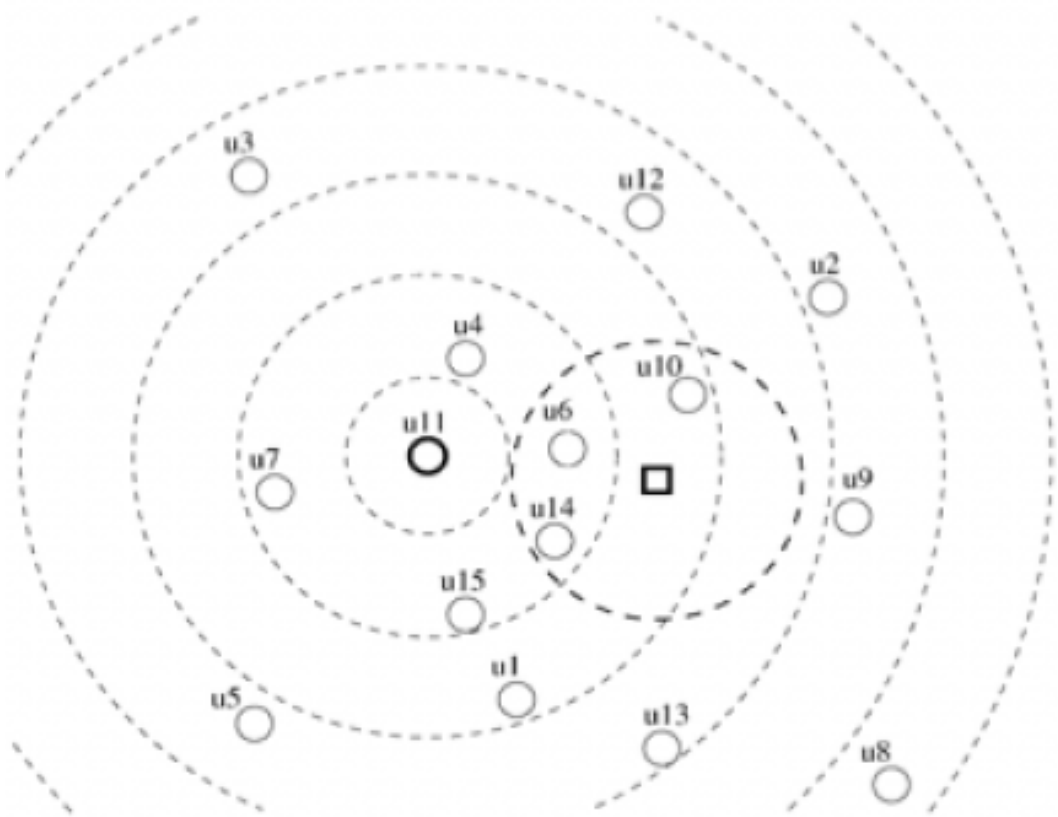
Dada una consulta por rango *(q,r)*:

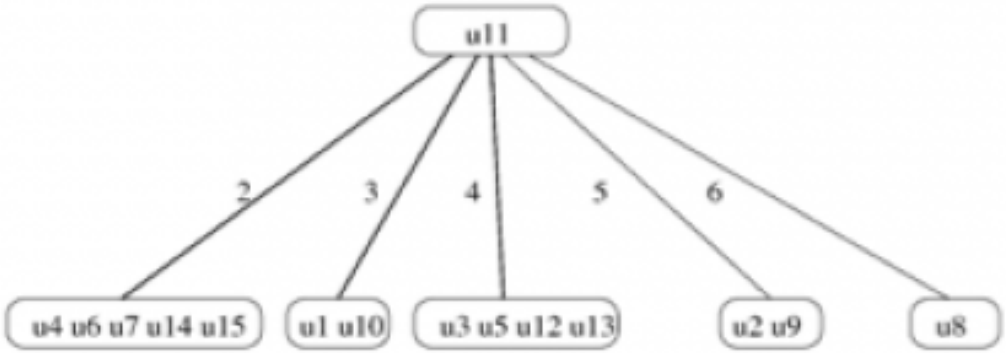
- Se comienza atravesando el árbol desde la raíz.
- En cada nodo interno $p_j$ :
  - reportar en la salida a $p_j$ si $d(q,p_j) \leq r$
  - entrar al hijo *i* si $\max\{d(q,p_j) - r, 0\} \leq i \leq d(q,p_j) + r$



# BKT


u3
u12
u2
u4
u10
u11
u6
u7
u9
u14
u15
u1
u5
u13
u8

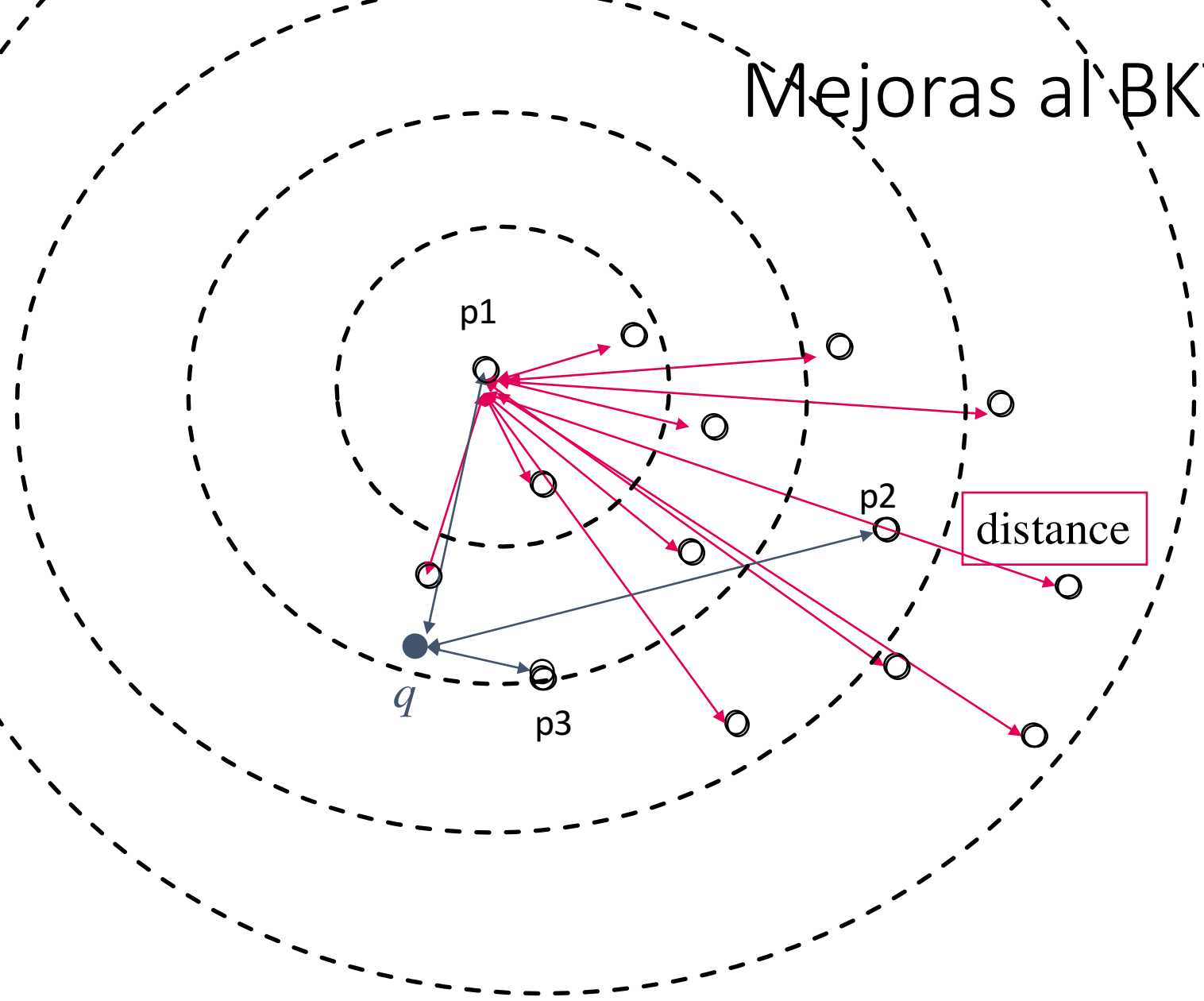

u11
2
3
4
5
6
u4 u6 u7 u14 u15
u1 u10
u3 u5 u12 u13
u2 u9
u8

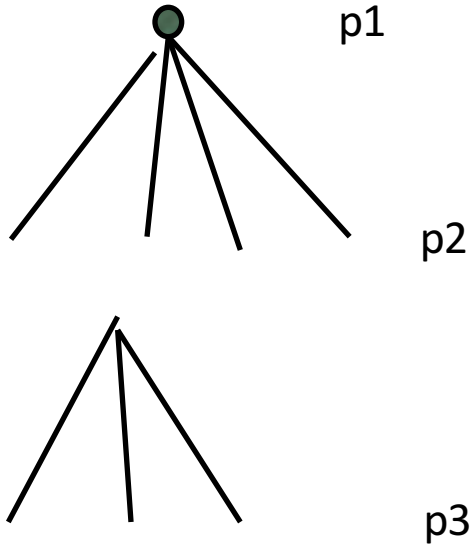
# Fixed Query Tree

[Baeza-Yates et al., 1994] Baeza-Yates, R. A., Cunto, W., Manber, U., and Wu, S. (1994). **Prox- imity matching using fixed-queries trees**. In Crochemore, M. and Gusfield, D., editors. *Pro- ceedings of the 5th Annual Symposium on Combinatorial Pattern Matching (CPM 1994), Asilomar, California, USA, June 5-8, 1994,* volume 807 of *Lecture Notes in Computer Sci- ence,* pages 198-212. Springer, Berlin.

# Mejoras al BKT: FQT (1994)

## Fixed Query Tree




Espacio: $O(n)....O(nh)$

$h = \log n$

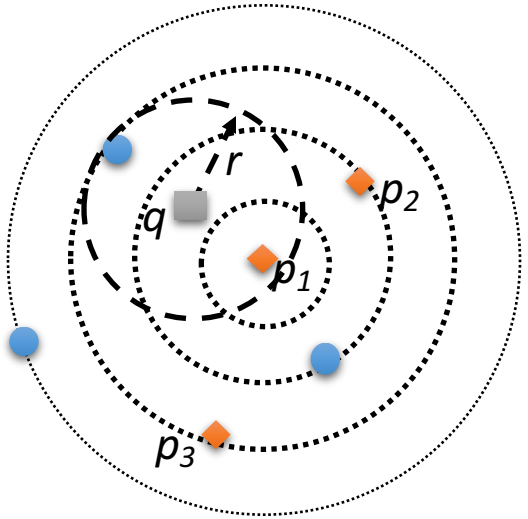
# FHQT. Fixed Height Query Tree
# FQA. Fixed Query Array

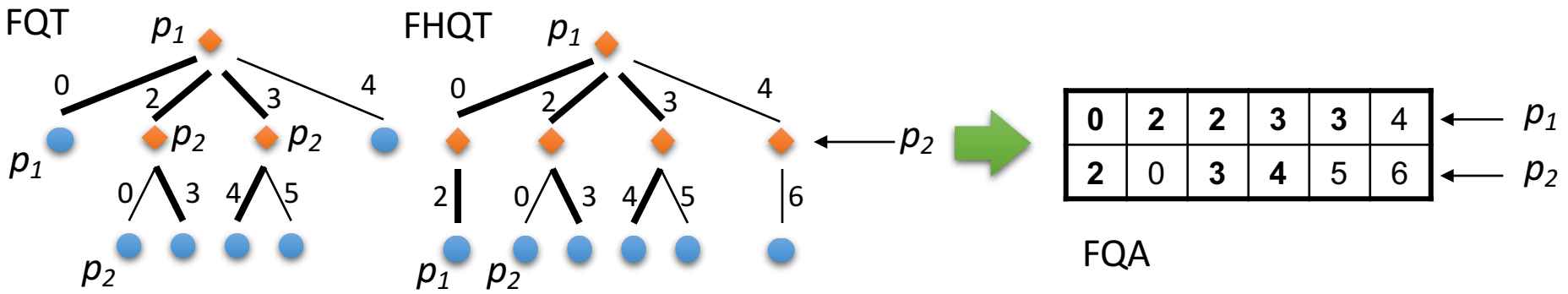
[Baeza-Yates, 1997] Baeza-Yates, R. A. (1997). Searching: an algorithmic tour. In Kent, A. and Wilhams, J. G., editors. *Encyclopedia of Computer Science and Technology,* volume 37, pages 331-359. Marcel Dekker, Inc.

[Chavez et al., 2001a] Chavez, E., Marroquin, J. L., and Navarro, G. (2001a). Fixed Queries Array: A fast and economical data structure for proximity searching. *Multimedia Tools and Applications,* 14(2): 113-135. Kluwer Academic Pubhshers.

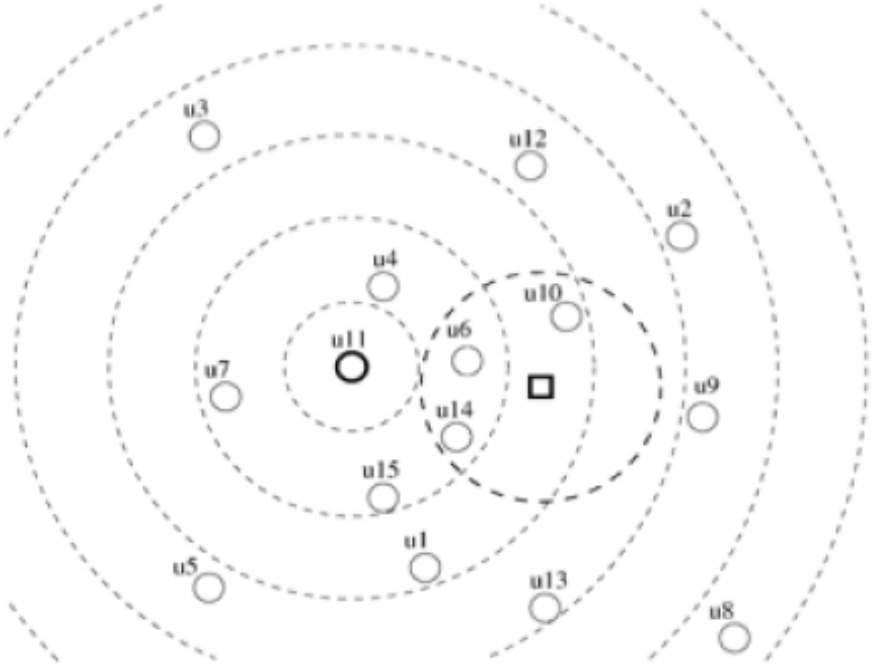
# Mejoras a la idea del BKT

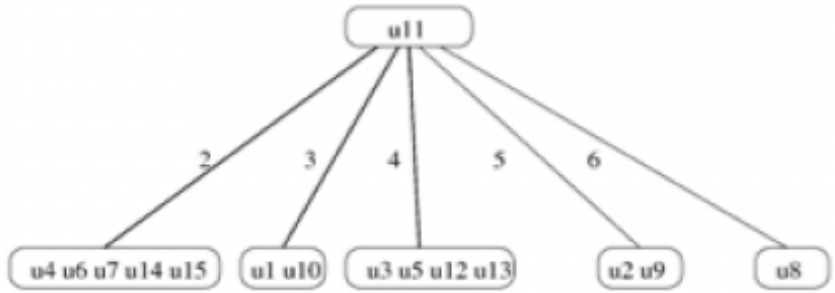
- En cada nivel del árbol se considera un pivote fijo: FQT
- Todos los elementos guardan las distancias a todos los pivotes de los distintos niveles: FHQT
- Se guardan sólo las distancias: FQA

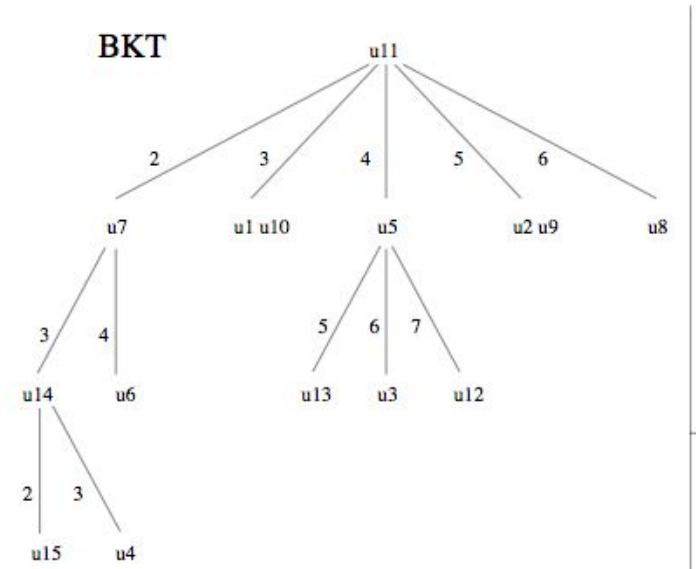


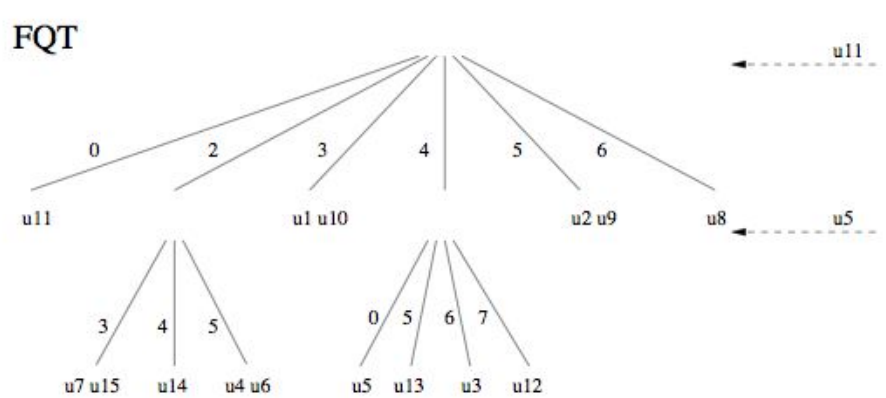



# Resumen



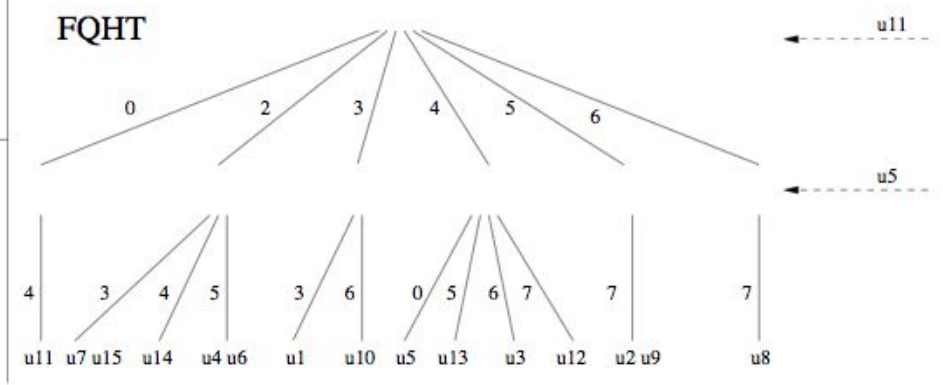



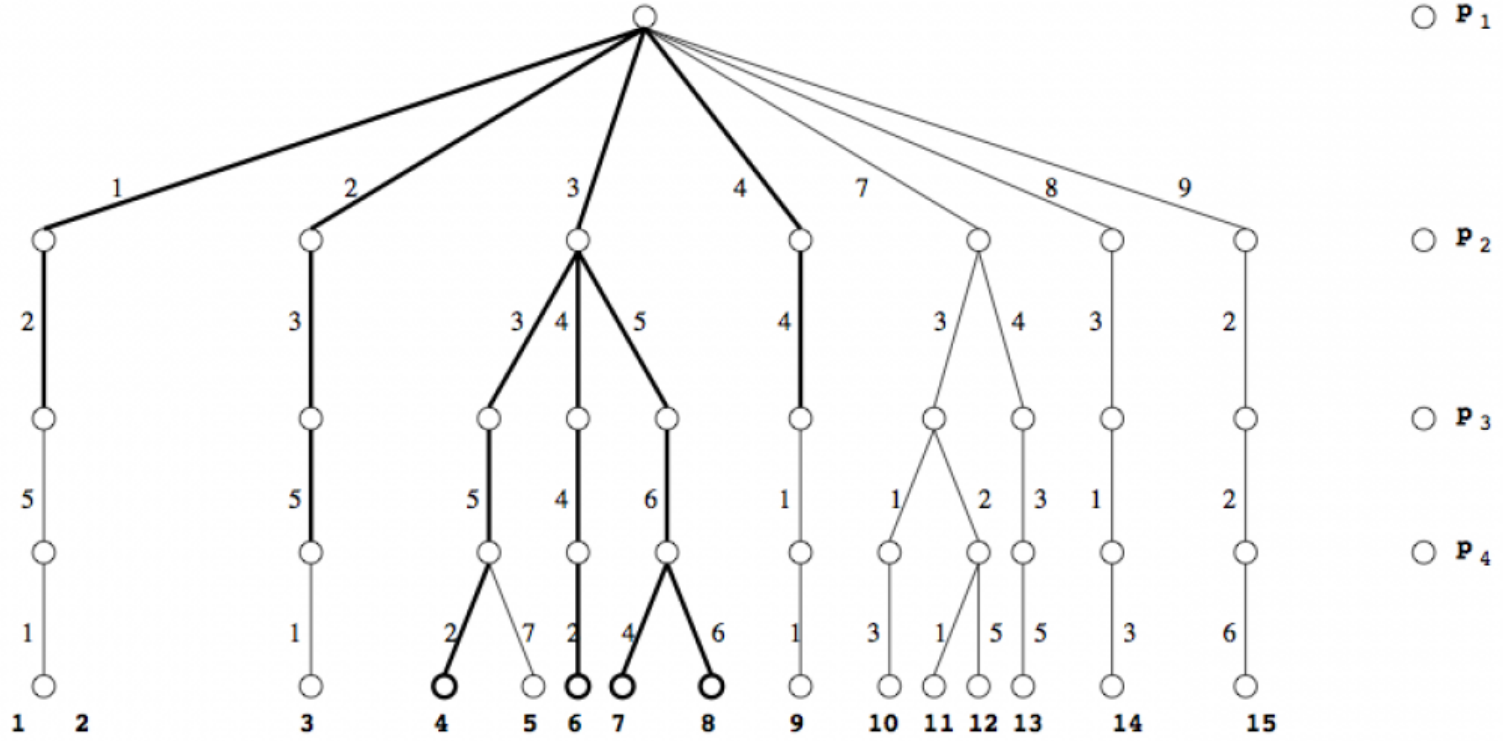




FQA

| u11 | u7 | u15 | u14 | u4 | u6 | u1 | u10 | u5 | u13 | u3 | u12 | u2 | u9 | u8 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 4 | 5 | 5 | 6 |
| 4 | 3 | 3 | 4 | 5 | 5 | 3 | 6 | 0 | 5 | 6 | 7 | 7 | 7 | 7 |



# FQHT vs FQA




(1,5)

| | ( | a | ) |
|---|---|---|---|
| **1** | **2** | **5** | **1** |
| **2** | **3** | **5** | **1** |
| **3** | **3** | **5** | **2** |
| **3** | **3** | **5** | **7** |
| **3** | **4** | **4** | **2** |
| **3** | **5** | **6** | **4** |
| **3** | **5** | **6** | **6** |
| **4** | **4** | **1** | **1** |
| 7 | 3 | 1 | 3 |
| 7 | 3 | 2 | 1 |
| 7 | 3 | 2 | 5 |
| 7 | 4 | 3 | 5 |
| 8 | 3 | 1 | 3 |
| 9 | 2 | 2 | 6 |

(2,6)

| | ( | b | ) |
|---|---|---|---|
| **1** | **2** | **5** | **1** |
| **2** | **3** | **5** | **1** |
| **3** | **3** | **5** | **2** |
| **3** | **3** | **5** | **7** |
| **3** | **4** | **4** | **2** |
| **3** | **5** | **6** | **4** |
| **3** | **5** | **6** | **6** |
| **4** | **4** | **1** | **1** |
| 7 | 3 | 1 | 3 |
| 7 | 3 | 2 | 1 |
| 7 | 3 | 2 | 5 |
| 7 | 4 | 3 | 5 |
| 8 | 3 | 1 | 3 |
| 9 | 2 | 2 | 6 |

(3,7)

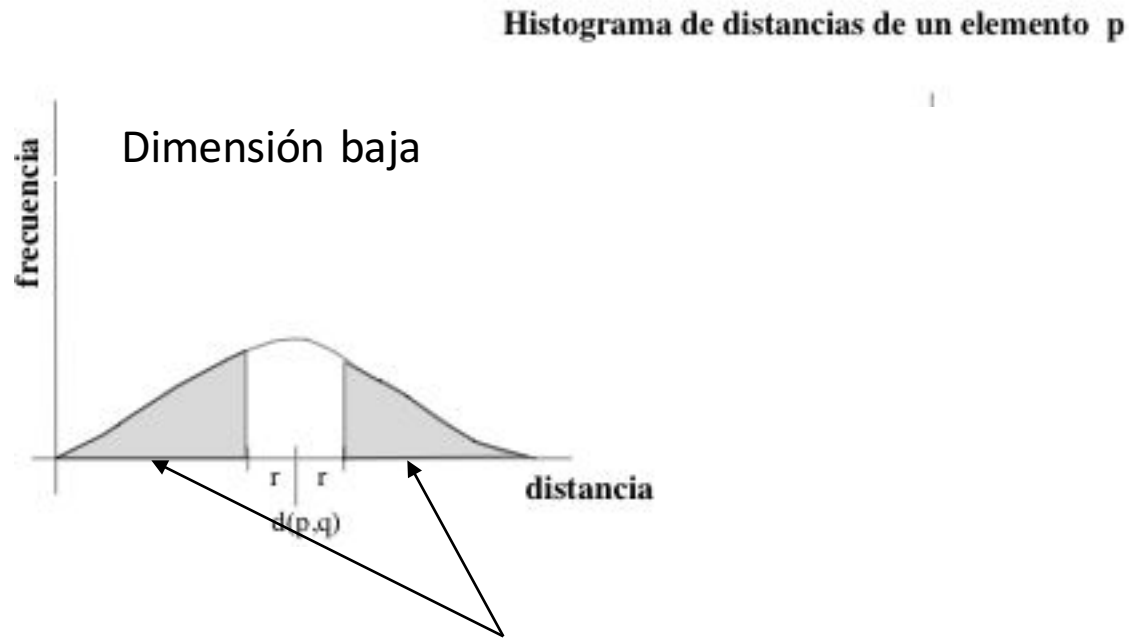
| | ( | c | ) |
|---|---|---|---|
| **1** | **2** | **5** | **1** |
| **2** | **3** | **5** | **1** |
| **3** | **3** | **5** | **2** |
| **3** | **3** | **5** | **7** |
| **3** | **4** | **4** | **2** |
| **3** | **5** | **6** | **4** |
| **3** | **5** | **6** | **6** |
| **4** | **4** | 1 | 1 |
| 7 | 3 | 1 | 3 |
| 7 | 3 | 2 | 1 |
| 7 | 3 | 2 | 5 |
| 7 | 4 | 3 | 5 |
| 8 | 3 | 1 | 3 |
| 9 | 2 | 2 | 6 |

(2,6)

| | ( | d | ) |
|---|---|---|---|
| **1** | **2** | **5** | 1 |
| **2** | **3** | **5** | 1 |
| **3** | **3** | **5** | **2** |
| **3** | **3** | **5** | 7 |
| **3** | **4** | **4** | **2** |
| **3** | **5** | **6** | **4** |
| **3** | **5** | **6** | **6** |
| **4** | **4** | 1 | 1 |
| 7 | 3 | 1 | 3 |
| 7 | 3 | 2 | 1 |
| 7 | 3 | 2 | 5 |
| 7 | 4 | 3 | 5 |
| 8 | 3 | 1 | 3 |
| 9 | 2 | 2 | 6 |

Mejoras del FQA:
FSFQA. Dividir en b partes iguales de distancias.
FQFQA. Dividir en quantiles

# Problema:
## La dimensión de los datos

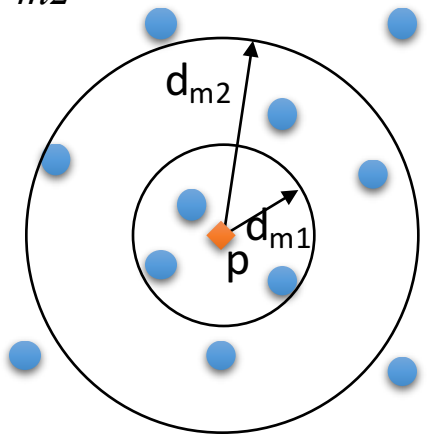



Elementos descartados en una consulta usando *p*

# Particionamiento por pivotes

- Algunos índices basados en pivotes clasifican de diferente manera los elementos de acuerdo a distintas maneras de particionar. Ejemplos: VPT, VPF, D-Index

  - El conjunto interno $d(p, x) \leq d_{m1}$
  - El conjunto central $d(p, x) > d_{m1}$ ^ $d(p, x) \leq d_{m2}$
  - El conjunto externo $d(p, x) > d_{m2}$

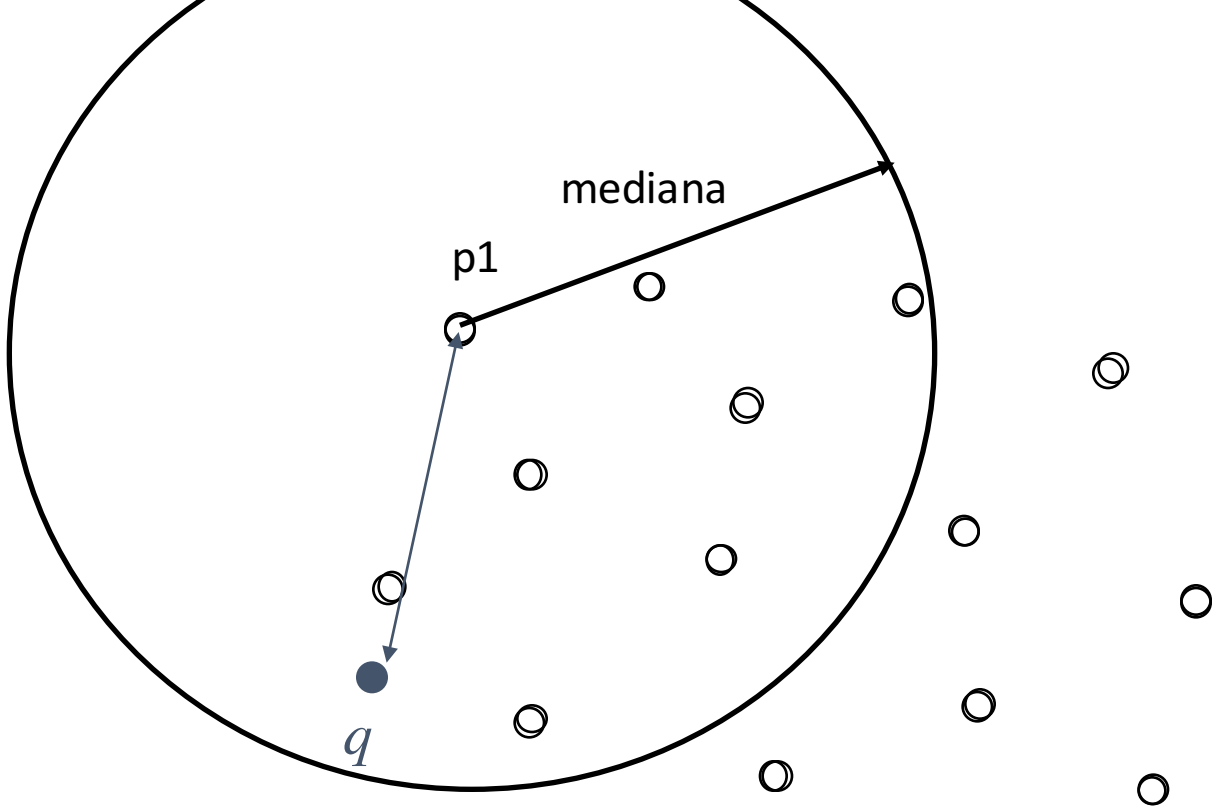


# VPT. Vantage Point Tree

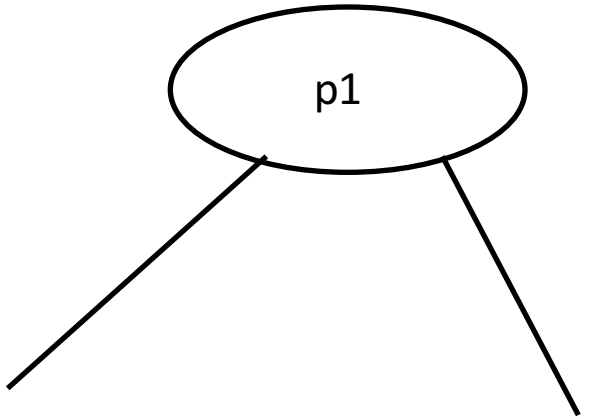
[Yianilos, 1993] Yianilos, P. N. (1993). **Data structures and algorithms for nearest neighbor search in general metric spaces**. In *Proceedings of the 4th Annual ACM Symposium on Discrete Algorithms (SODA 1993), Austin, Texas, USA, January 25-27, 1993,* pages 311- 321. ACM Press.

# VPT

Vantage Point Tree

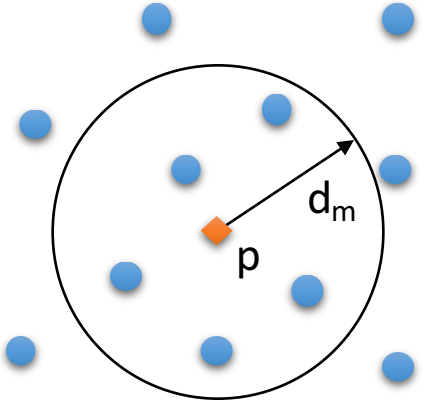


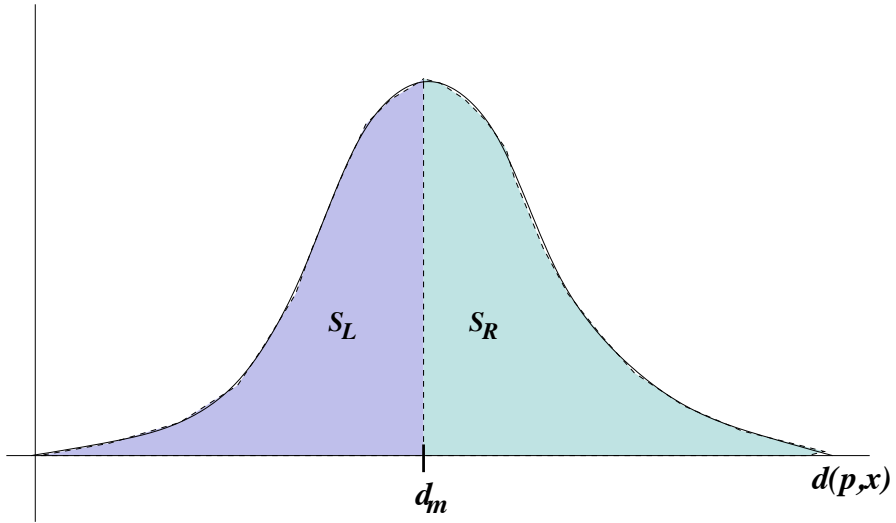

Espacio: $O(n)$

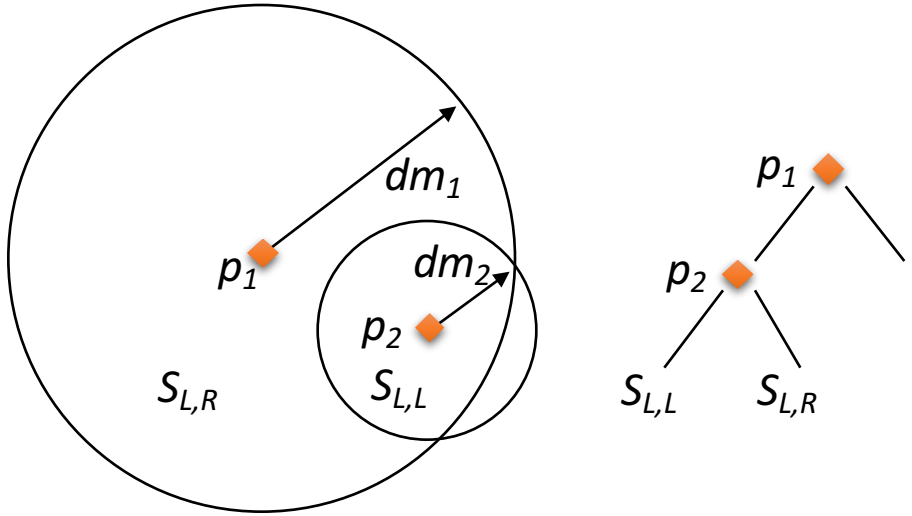


# Vantage Point Tree (VPT)

- Particiona el espacio con pivotes y la mediana $d_m$ de las distancias al pivote.
- En cada nodo guarda el pivote utilizado y la mediana.
- Es un árbol balanceado.

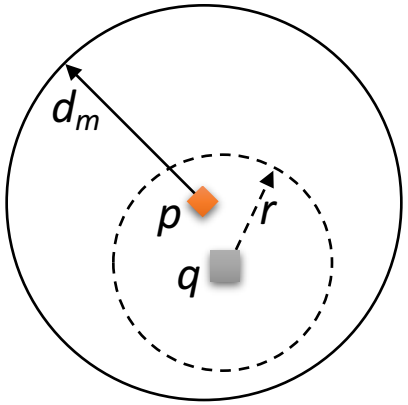


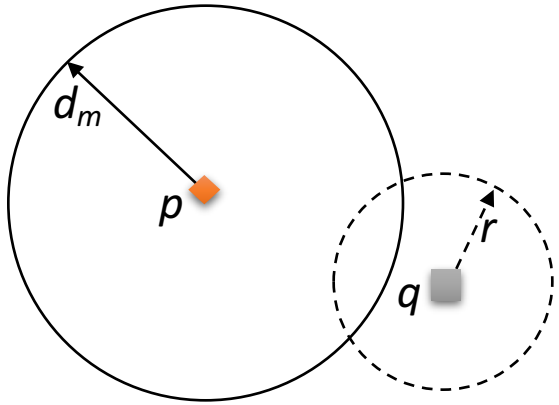



# VPT: búsqueda

- Para $(q, r)$ se calcula $d = d(q,p)$.
  - Si $d - r \leq d_m$ entramos en el subárbol izquierdo
  - Si $d + r > d_m$ entramos en el subárbol derecho.



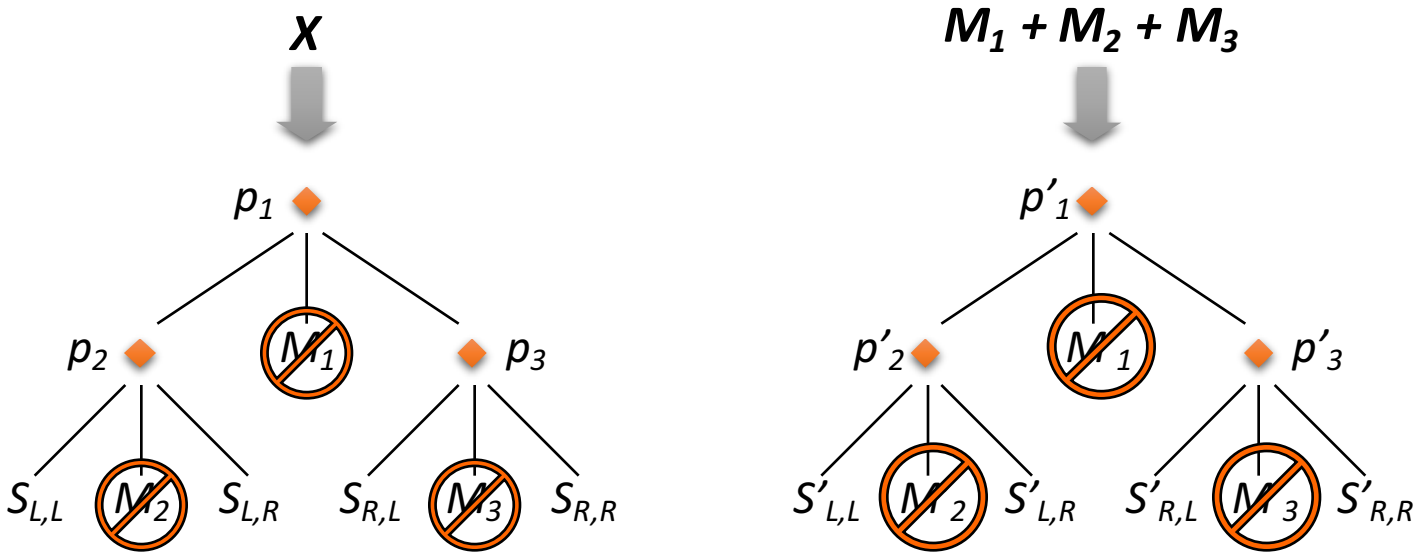



# VPF. Vantage Point Forest

[Yianilos, 1999] Yianilos, P. N. (1999). Excluded middle vantage point forests for nearest neigh- bor search. In *Proceedings of the 6th DIMACS Implementation Challenge: Near Neighbor Searches (ALENEX1999), Baltimore, Maryland, USA, January 15-16, 1999.*

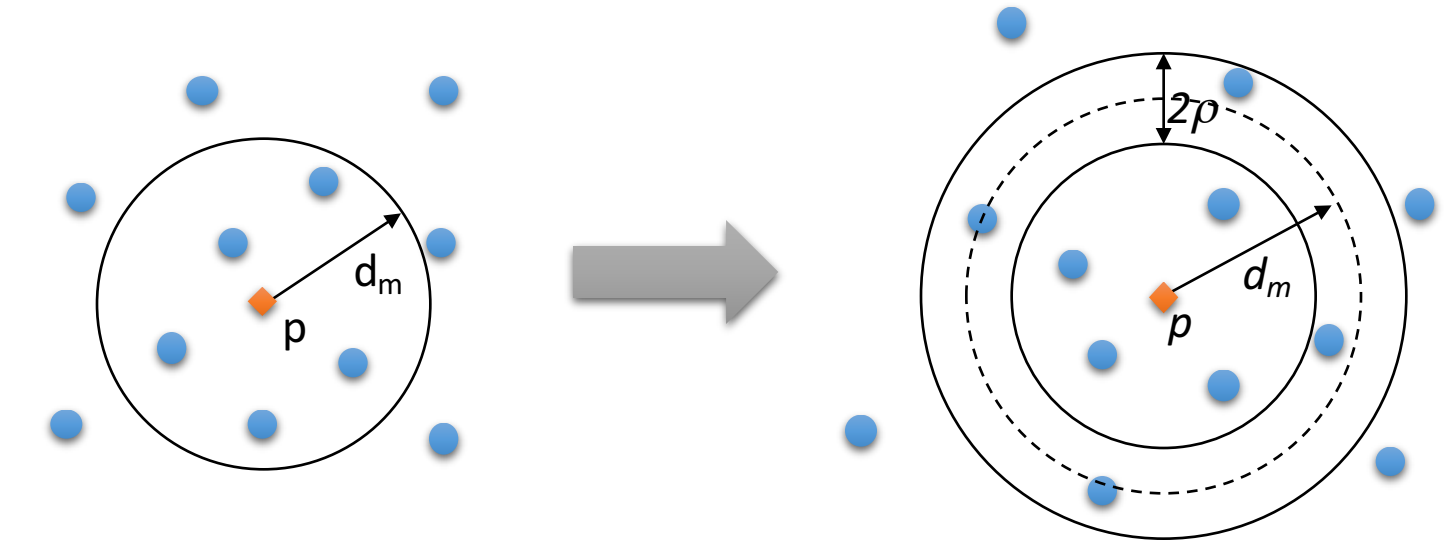
# Vantage Point Forest (VPF)

- Es una foresta de VPT's.
- En cada nivel excluye los elementos a distancia intermedia de su pivote y con los elementos excluidos del primer árbol se construye otro VPT.
- Se elimina backtracking cuando se busca con radio *r* menor que el "radio" de la zona descartada.

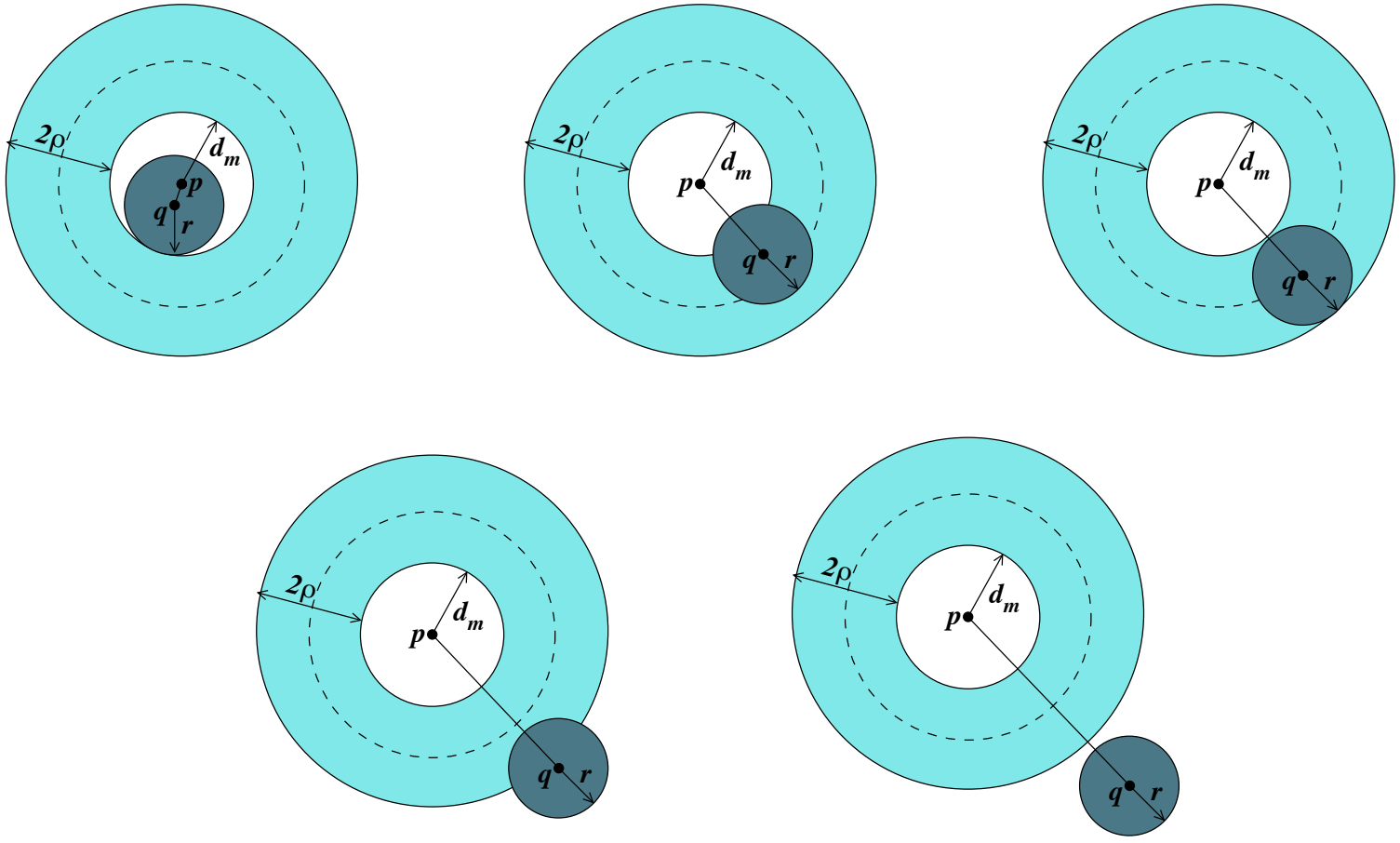


# VPF: Construcción

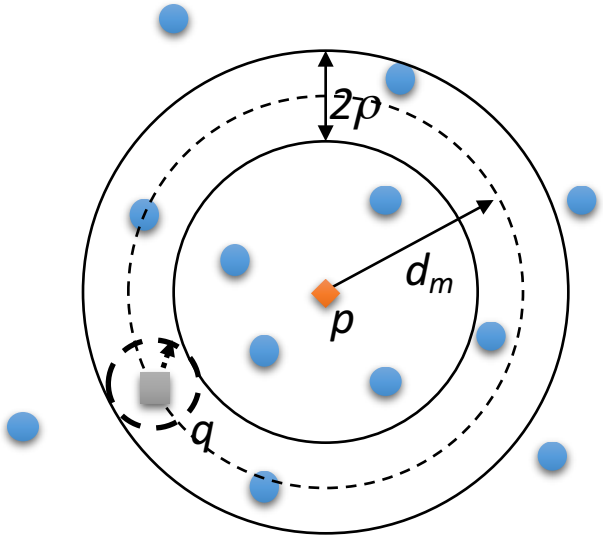
- Divide el espacio en más zonas que el VPT, para evitar en lo posible revisar más de una zona en las búsquedas.

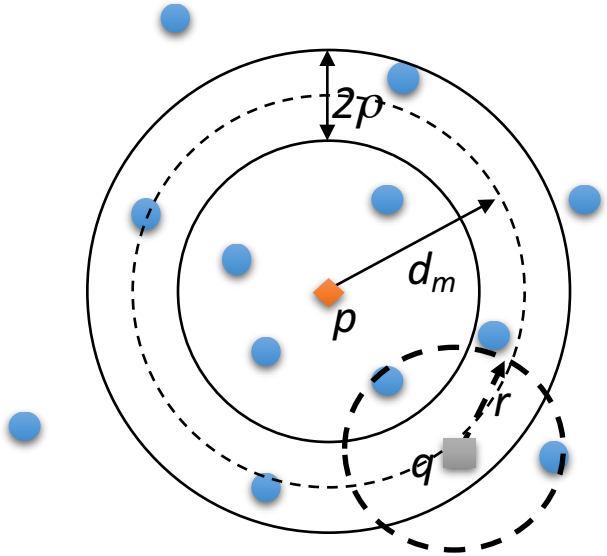


# VPF: búsqueda



# VPF: búsqueda

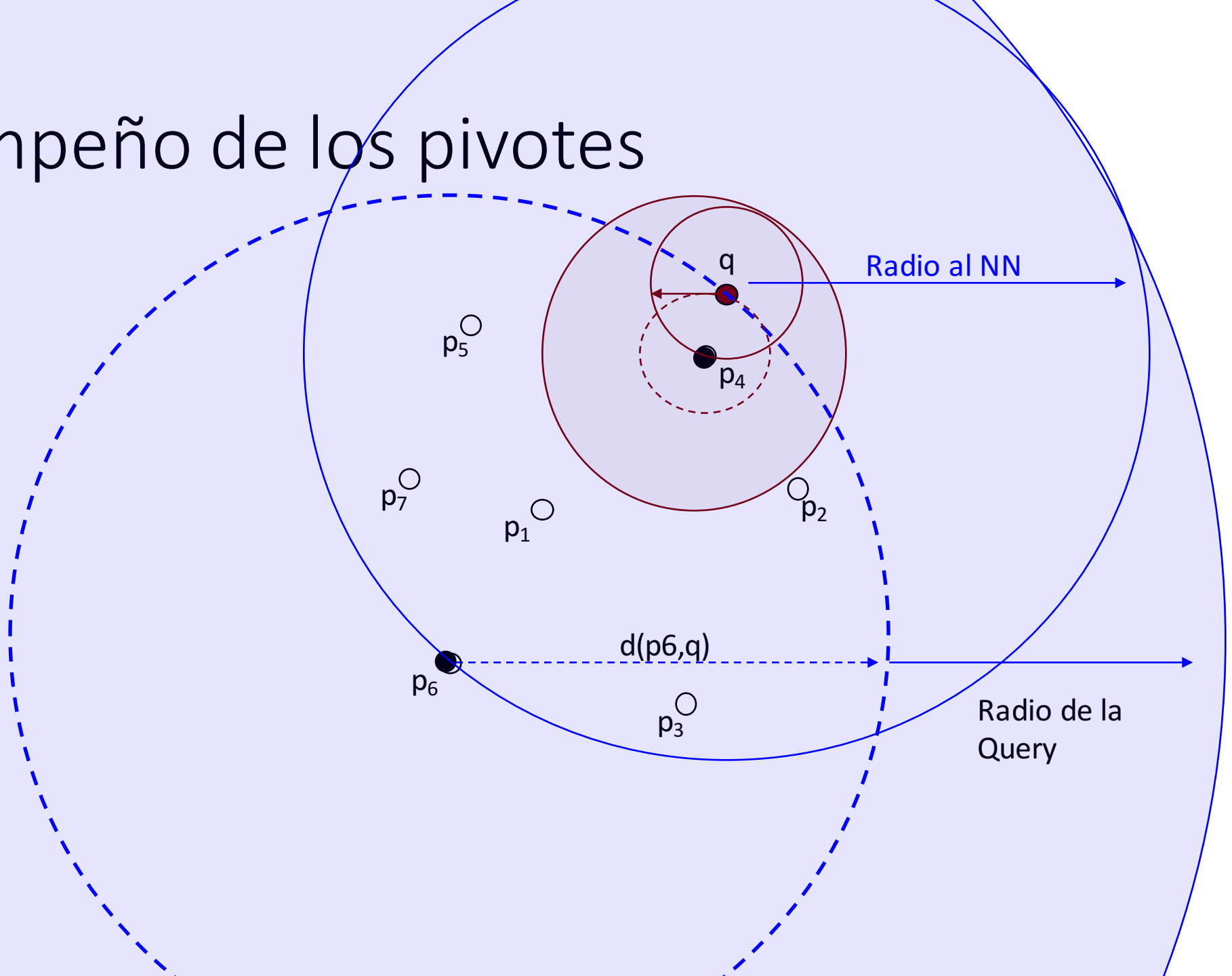


# AESA. Approximating and Eliminating Searching Algorithm

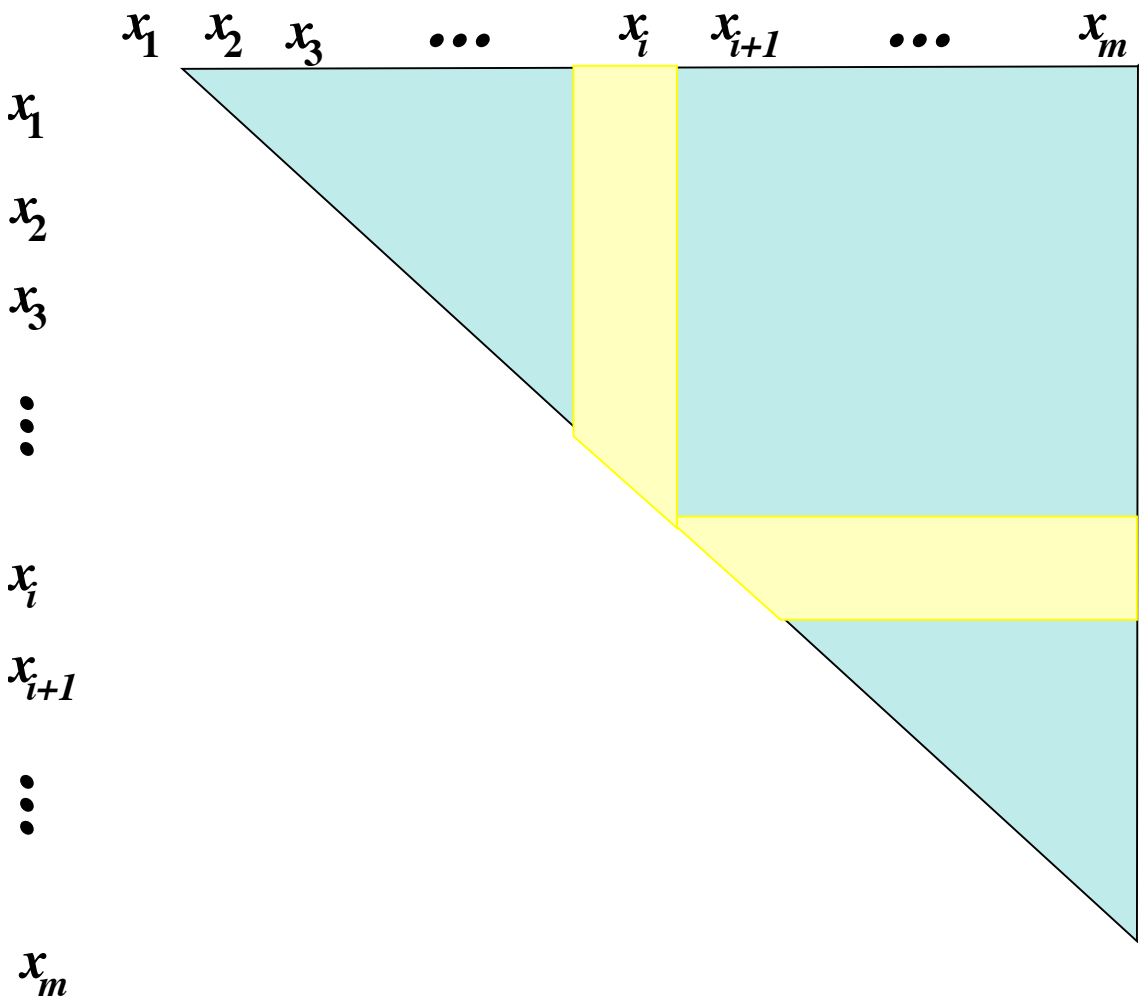
[Vidal, 1986] Vidal, E. (1986). **An algorithm for finding nearest neighbors in (approximately) constant average time**. *Pattern Recognition Letters,* 4(3): 145-157. Elsevier.

# Desempeño de los pivotes



# AESA

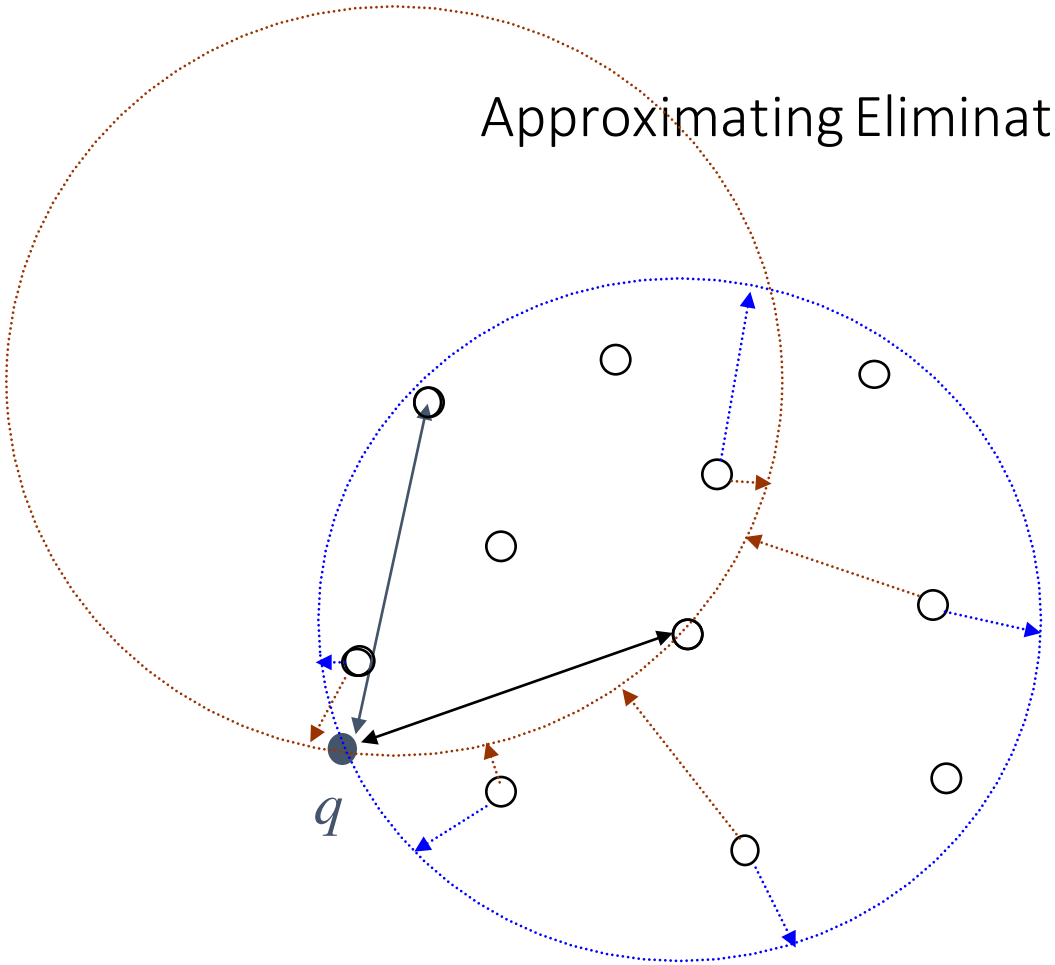
- Approximating Eliminating Search Algorithm




Se guardan las distancias de todos contra todos (sólo la matriz triangular por la simetría).

# AESA

Approximating Eliminating Search Algorithm

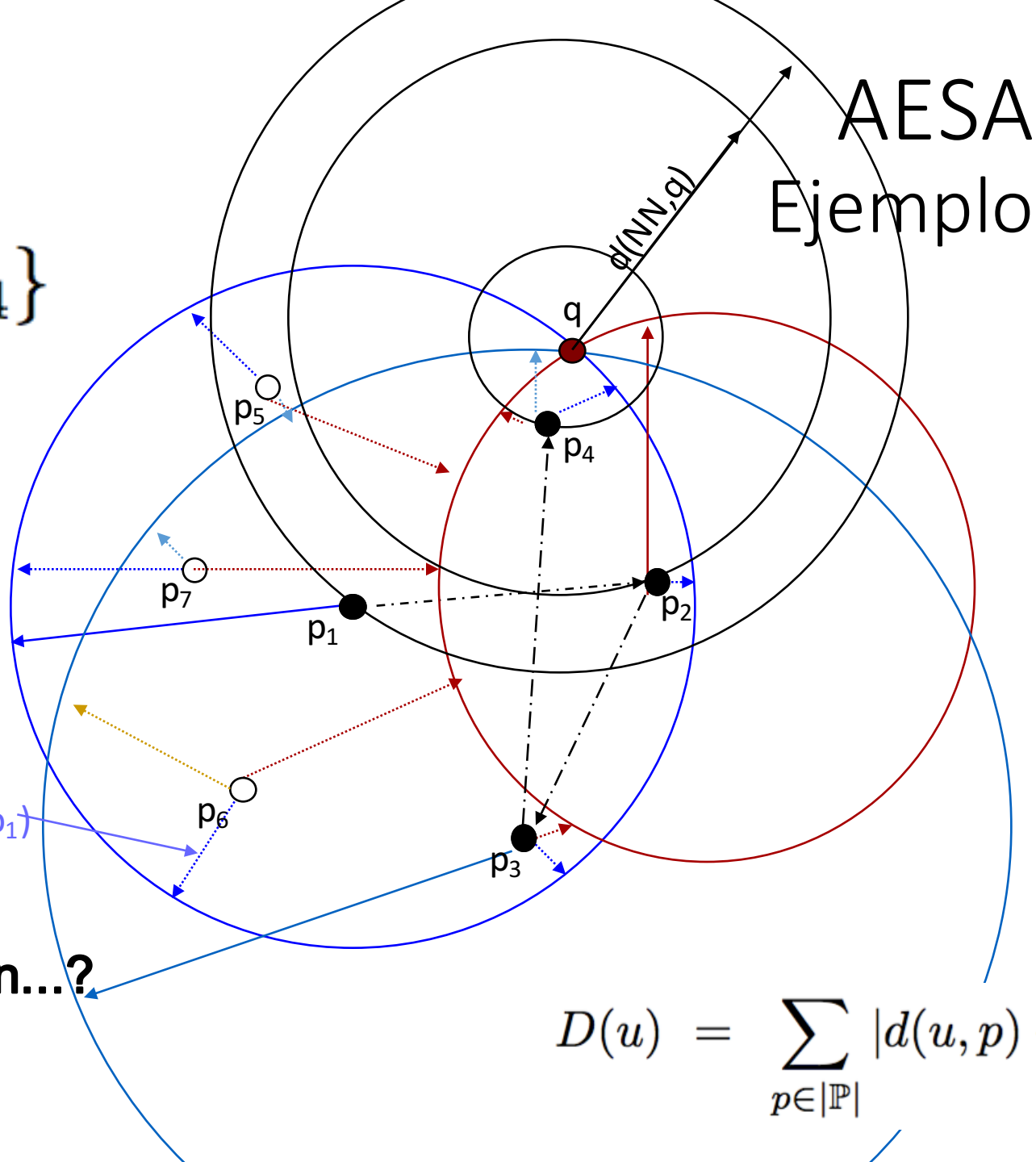



Espacio: $O(n^2)$

$$D(u) = \sum_{p \in |\mathbb{P}|} |d(u,p) - d(p,q)|,$$

# AESA Ejemplo

$$NN = \{p_4\}$$




**En qué orden...?**

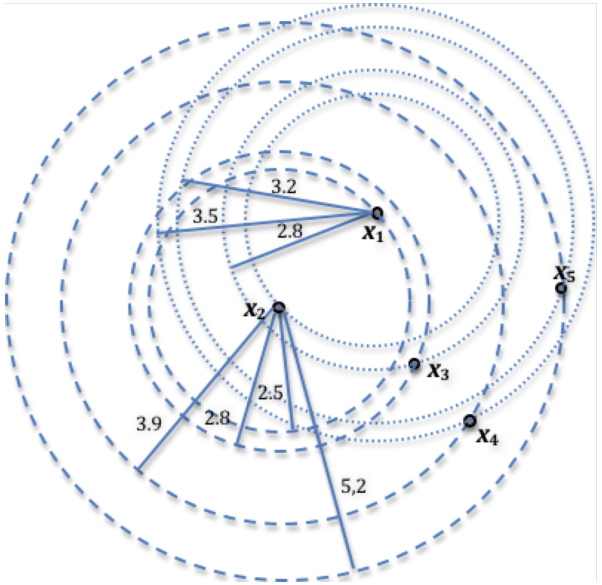
$$D(u) = \sum_{p \in |\mathbb{P}|} |d(u,p) - d(p,q)|,$$

# Mejoras de AESA

- LAESA. Linear AESA [Micó et al. 1992]
  - Almacena solo k filas de la matriz de distancias, necesita solo $\Theta(kn)$ espacio. La estrategia de búsqueda es igual solo que ahora está limitado a k pivotes.
- PAESA [Figueroa and Fredikson, 1998]
  - Divide la base de datos en P bloques de tamaño n/P objetos. Ahora tiene P AESA matrices, requiere $\Theta(n^2/P)$ espacio(y cálculos de distancia)
- BAESA- Binary/Bounded AESA. [Figueroa and Fredikson, 1998]
  - Para cada objeto $o_i$ en la base de datos se calculan y almacenan solo b distancias o radios, es decir, $R_{0\ldots n-1,0\ldots b-1}$.

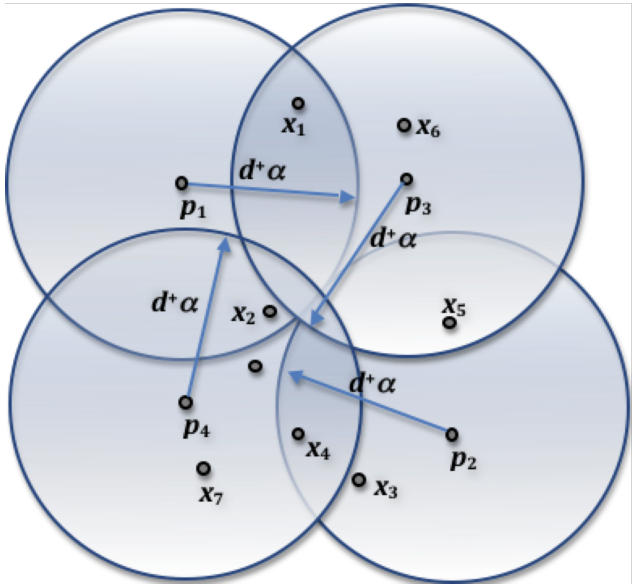
# Selección de Pivotes

- Existen distintas maneras de seleccionar pivotes desde *X*:
  - Aleatoriamente: se eligen al azar objetos de la base de datos para que actúen como pivotes.
  - Con alguna estrategia de selección:
    - SSS: Sparse Spatial Selection
    - INC: Incremental
    - Conjunto que maximice la media de las distancias.
  - Buenos pivotes para una consulta *q* son elementos cercanos o lejanos a *q.*



# Sparse Spatial Selection (SSS)

- Método dinámico de selección de pivotes.
- La cantidad de pivotes se adapta al espacio.
- Es necesario conocer o estimar la distancia máxima $d^+$ y utilizar un valor α como parámetro ($0 \leq \alpha \leq 1$).
- Buenos valores para α son 0,4 o 0,5.
- Hay estrategias para revisar si a un pivote valdría la pena mantenerlo o si vale la pena descartarlo.



# Algoritmos basados en particiones compactas

BST

Bisector Tree

SAT

Spatial Aproximation Tree

LC

List of Cluster

# Principios de particionamiento

## Particionamiento con un radio

Dividir la base de datos $\mathbb{U}$ en dos subconjuntos $\mathbb{U}_1$ y $\mathbb{U}_2$ usando un corte esférico respecto a un punto $p \in \mathbb{U}$. Donde $p$ es un *pivote*. Sea $d_m$ la mediana de $\{d(u,p), \forall u \in \mathbb{U}\}$.

- $\mathbb{U}_1 \leftarrow \{u_i | d(u_i,p) \leq d_m\}$
- $\mathbb{U}_2 \leftarrow \{u_j | d(u_j,p) \geq d_m\}$

## Particionamiento por Hiperplanos generalizados

Dividir la base de datos $\mathbb{U}$ en dos subconjuntos $\mathbb{U}_1$ y $\mathbb{U}_2$ usando un corte por hiperplanos respecto a dos puntos $p_1, p_2 \in \mathbb{U}$. Donde $p_1$ y $p_2$ son *pivotes*.

- $\mathbb{U}_1 \leftarrow \{u_i | d(u_i,p_1) \leq d(p_2,u_i)\}$
- $\mathbb{U}_2 \leftarrow \{u_j | d(u_j,p_1) \geq d(p_2,u_j)\}$

## Excluyendo la partición central

Dividir la base de datos $\mathbb{U}$ en tres subconjuntos $\mathbb{U}_1$, $\mathbb{U}_2$ y $\mathbb{U}_3$ usando un corte esférico respecto a un punto $p \in \mathbb{U}$. Donde $p$ es un *pivote*. Sea $d_m$ la mediana de $\{d(u,p), \forall u \in \mathbb{U}\}$. Además hay una zona de exclusión $2\rho$.

- $\mathbb{U}_1 \leftarrow \{u_i | d(u_i,p) \leq d_m - \rho\}$
- $\mathbb{U}_2 \leftarrow \{u_j | d(u_j,p) \geq d_m - \rho\}$
- $\mathbb{U}_3 \leftarrow$ en otro caso.

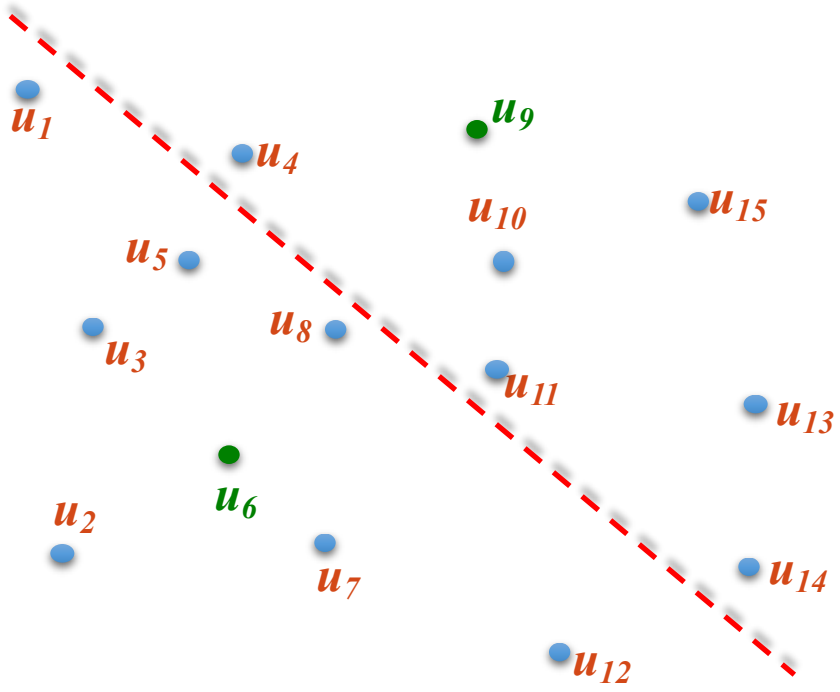
# BST. Bisector Tree

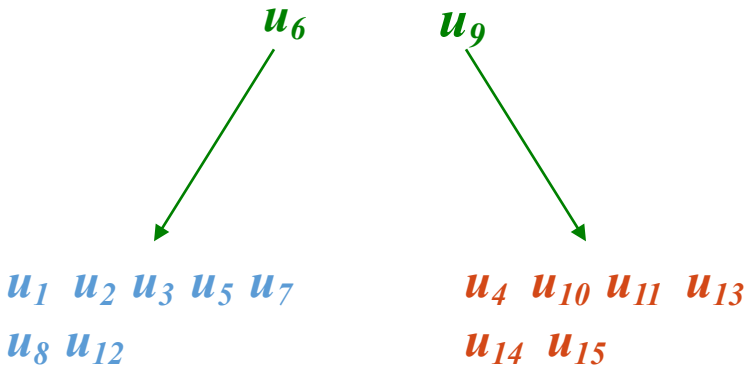
[Kalantari and McDonald, 1983] Kalantari, I. and McDonald, G. (1983). **A data structure and an algorithm for the nearest point problem**. *IEEE Transactions on Software Engineering (TSE 1983),* 9(5):631-634. IEEE Computer Society.

# Bisector Tree (BST)

- Cada nodo tiene dos centros: $c_1$ y $c_2$
- Los elementos más cercanos a $c_1$ que a $c_2$ van al subárbol izquierdo.
- Los elementos más cercanos a $c_2$ que a $c_1$ van al subárbol derecho.
- Cada centro almacena su radio de cobertura.

# BST: Construcción

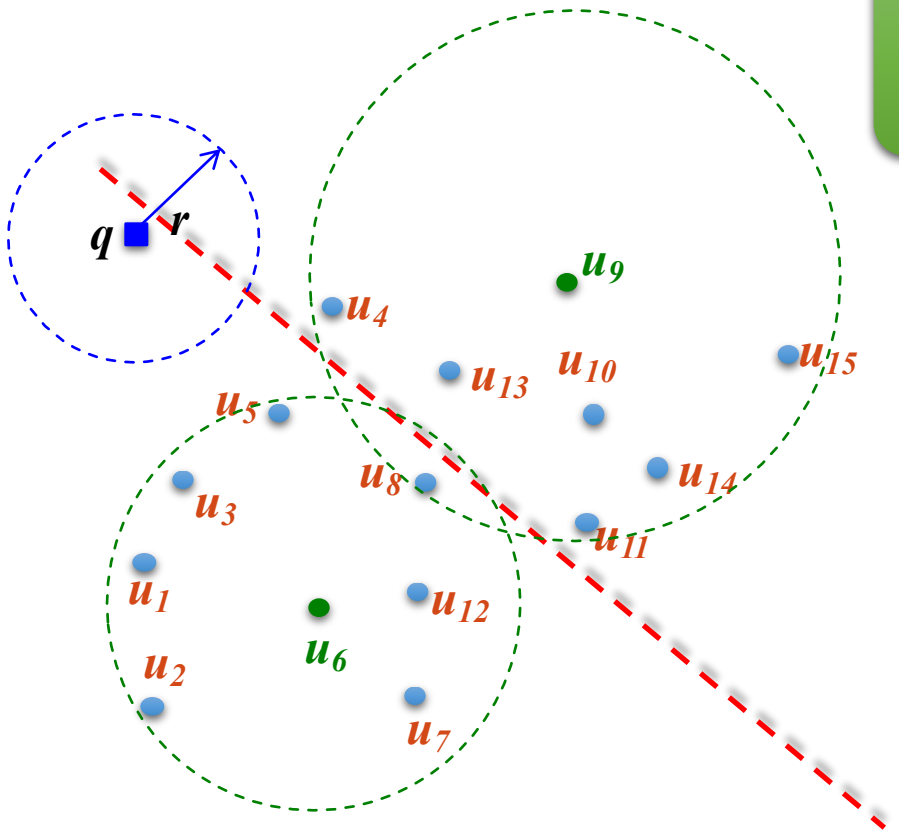



$$rc\ (u_6) = máx\ \{d(u_6, u_i) / u_i \in U_{u_6}\}$$

$$rc\ (u_9) = máx\ \{d(u_9, u_i) / u_i \in U_{u_9}\}$$

# BST: Búsqueda



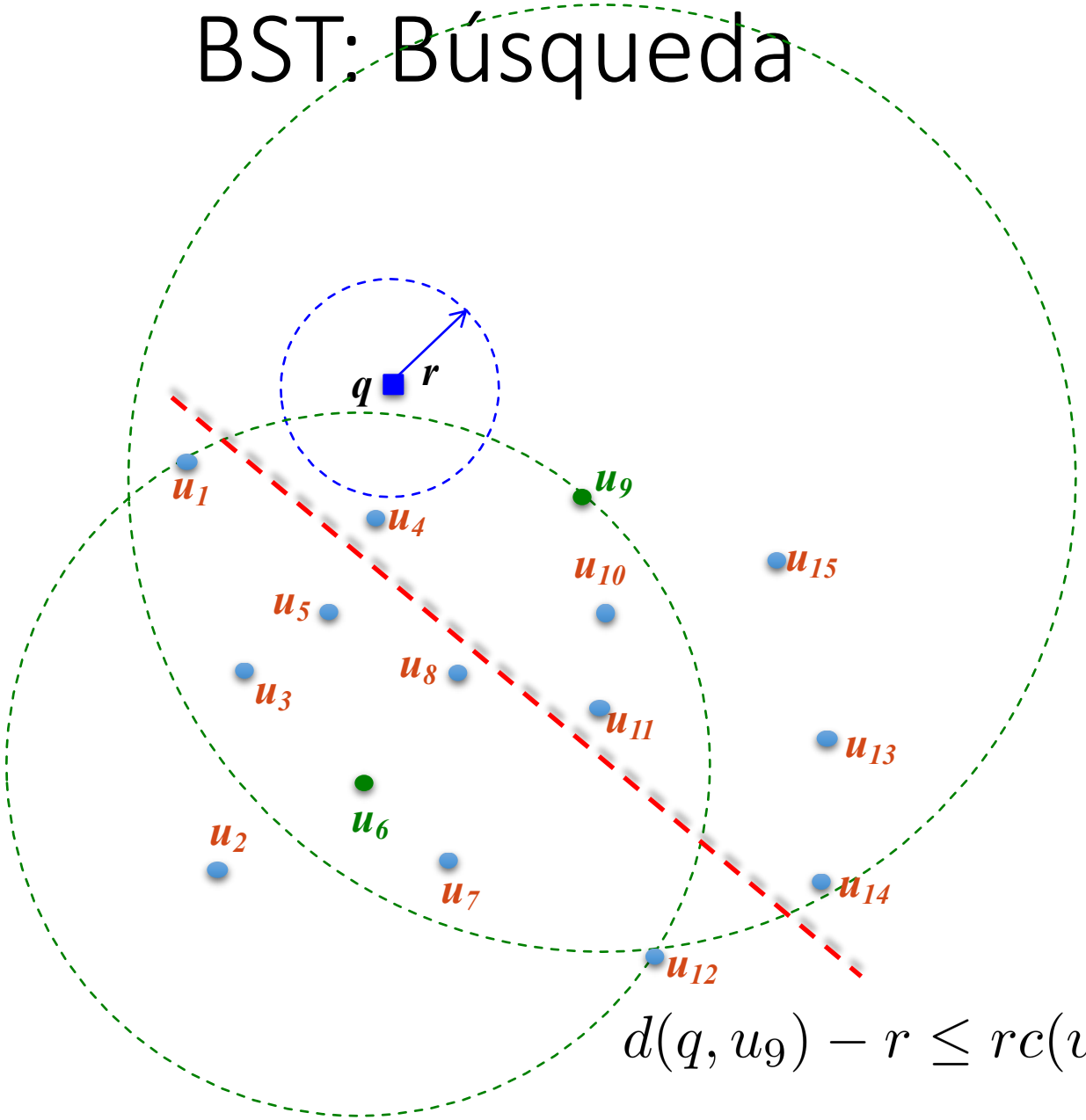

BST usa el criterio de radio de cobertura para podar las búsquedas.

¿En dónde debe entrar a buscar en este caso?

A pesar de intersectar la consulta a ambos hiperplanos, no entraría a buscar en ninguna de las ramas del árbol

$$d(q, u_9) - r > rc(u_9) \wedge d(q, u_6) - r > rc(u_6)$$

# BST: Búsqueda




¿En dónde debe entrar a buscar en este caso?

A pesar de que la consulta cae completamente en el hiperplano de $u_9$, debe buscar en ambas ramas

$$d(q, u_9) - r \leq rc(u_9) \wedge d(q, u_6) - r \leq rc(u_6)$$

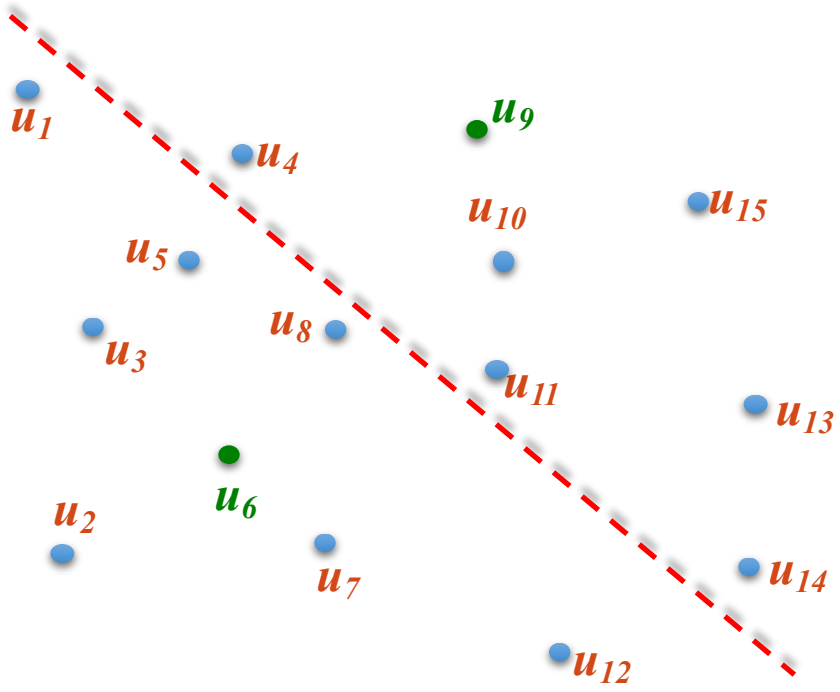
# GHT. Generalized Hyperplane Tree

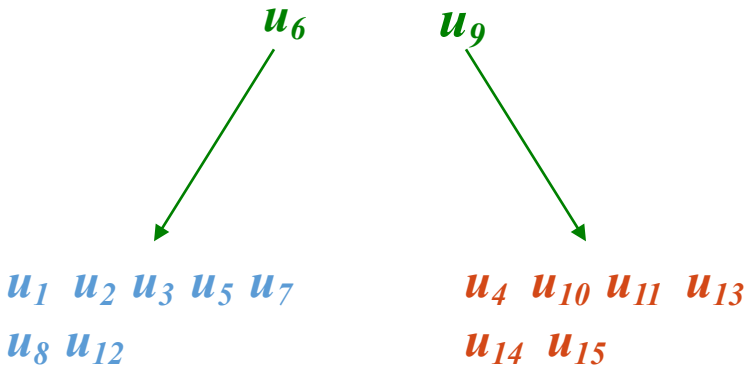
[Uhlmann, 1991] Uhlmann, J. K. (1991). Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters,* 40(4): 175-179. Elsevier.

# Generalized-Hyperplane Tree (GHT)

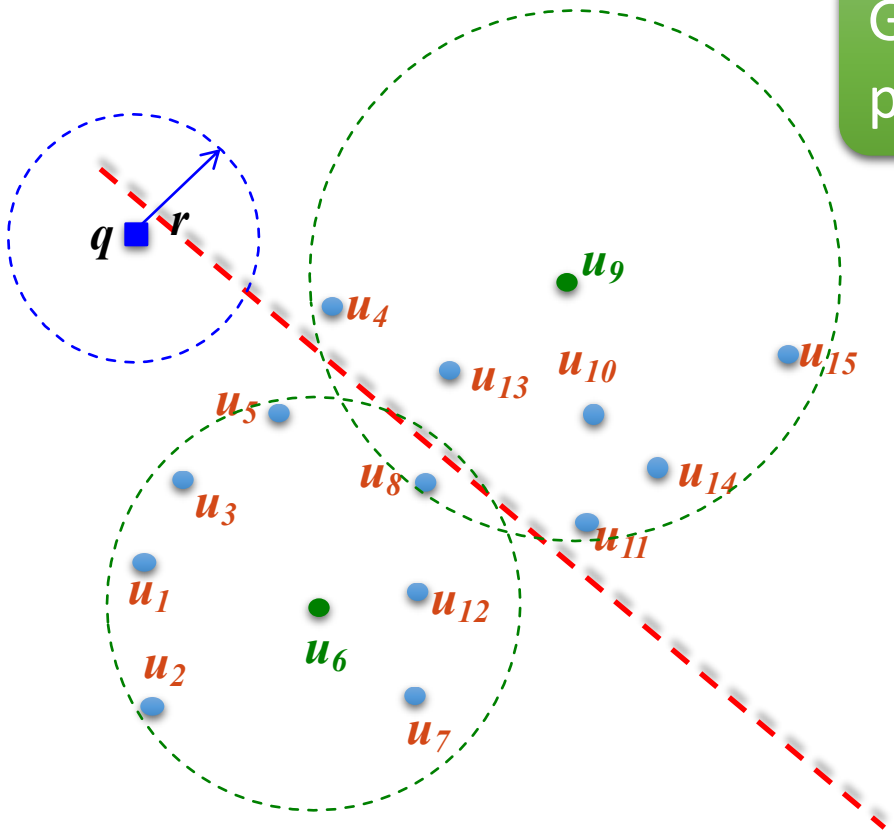
- Cada nodo tiene dos centros: $c_1$ y $c_2$
- Los elementos más cercanos a $c_1$ que a $c_2$ van al subárbol izquierdo.
- Los elementos más cercanos a $c_2$ que a $c_1$ van al subárbol derecho.
- La construcción es idéntica al BST, pero cada centro ya no almacena su radio de cobertura.

# GHT: Construcción



# GHT: Búsqueda



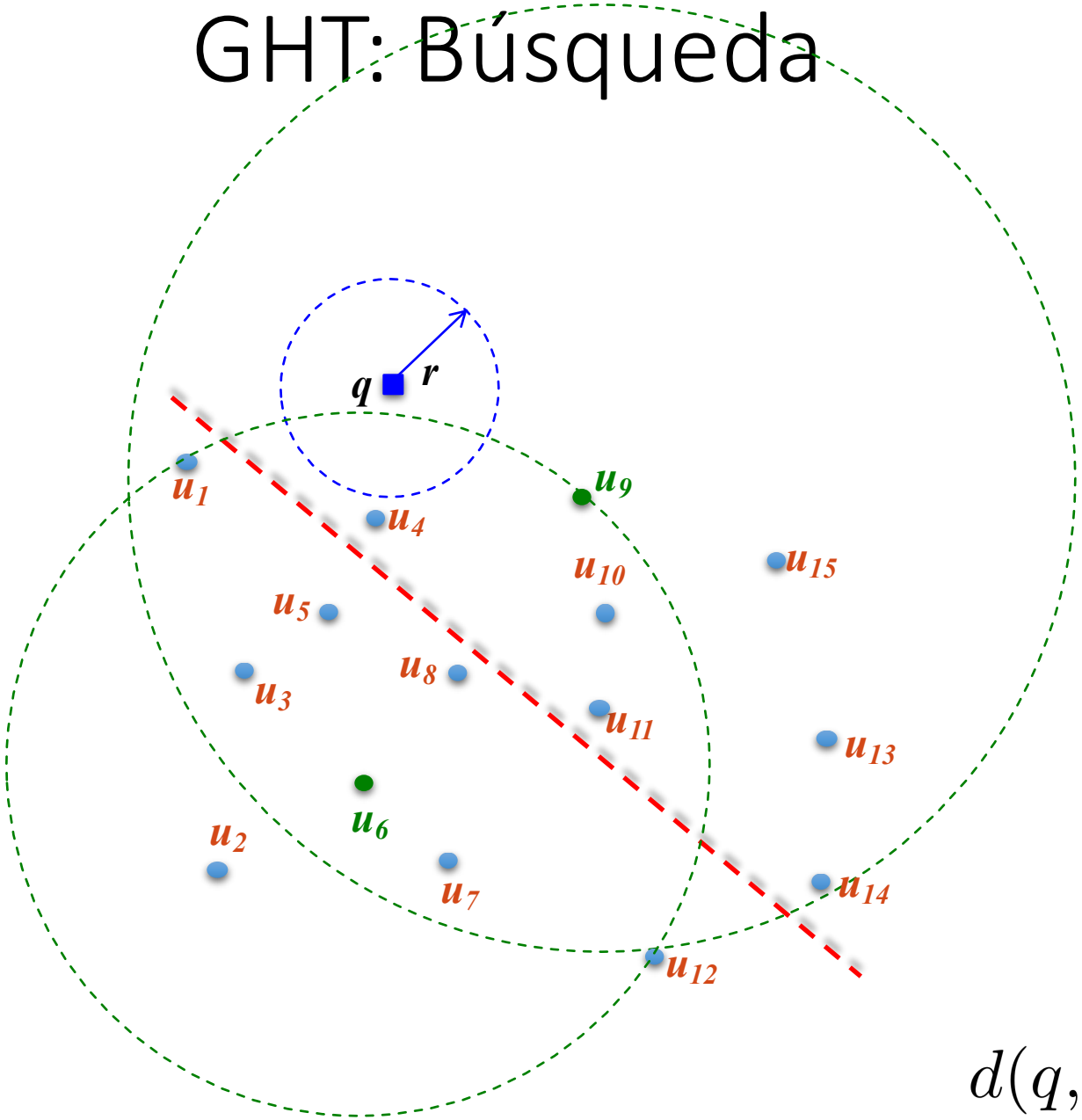

GHT usa el criterio del hiperplano para podar las búsquedas.

¿En dónde debe entrar a buscar en este caso?

Como la consulta intersecta a ambos hiperplanos, entraría a buscar en ambas ramas del árbol

$$d(q, u_6) + r > d(q, u_9) - r \wedge d(q, u_9) + r > d(q, u_6) - r$$

# GHT: Búsqueda



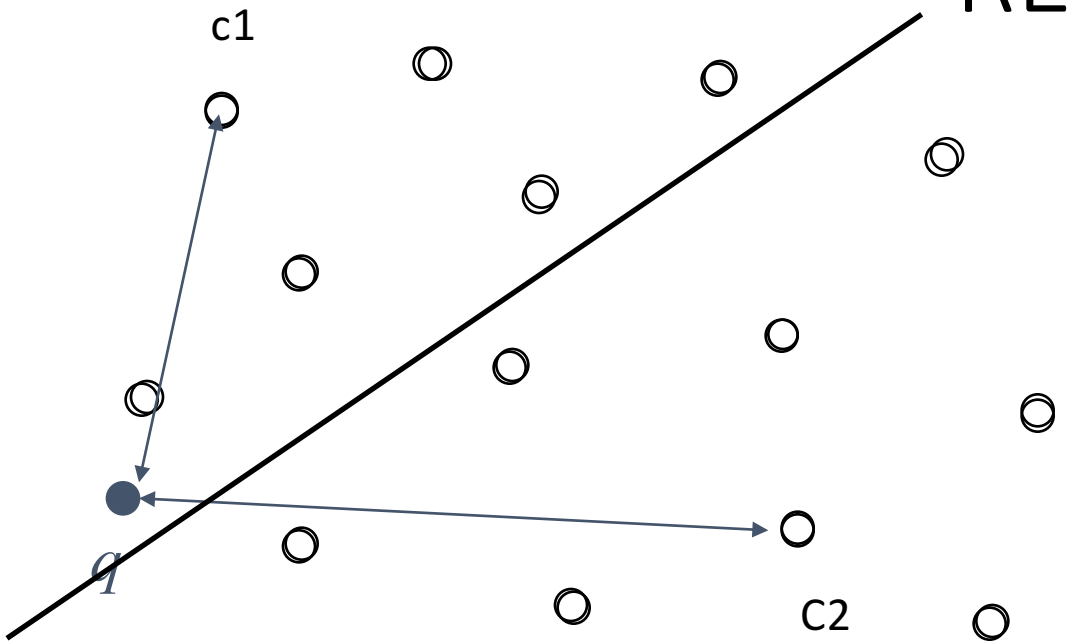

¿En dónde debe entrar a buscar en este caso?

Como la consulta cae completamente en el hiperplano de $u_9$, debe buscar sólo en esa rama

$$d(q, u_9) + r < d(q, u_6) - r$$

# RESUMEN: BST

## Bisector Tree



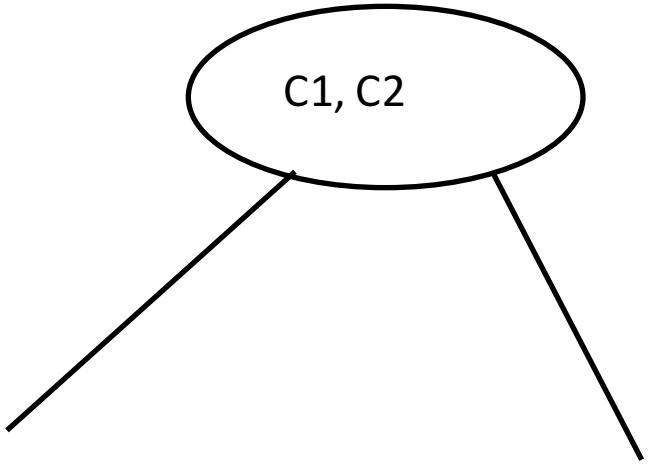

Por cercanía

Espacio: $O(n)$

## GHT

Por hiperplanos

Generalized hyperplane Tree




GNAT GHT de aridad m

Espacio: $O(nm^2)$

# LC. List of Clusters

E. Chávez and G. Navarro. **An effective clustering algorithm to index high dimensional metric spaces.** In IEEE CS Press, editor, *7th International Symposium on String Processing and Information Retrieval (SPIRE 2000)*, pages 75–86, 2000.

# Lista de Clusters (LC)

- La Lista de Clusters divide el espacio en zonas.
- Cada zona tiene un centro $c$, un radio de cobertura $r_c$ y un bucket de objetos internos $I$.
- La bola de centro de $(c,\ r_c)$ corresponde al subconjunto elementos de $U$ (base de datos) tal que están a distancia a lo más $r_c$ de $c$.

# Lista de Clusters

- Se define el bucket de elementos internos como:

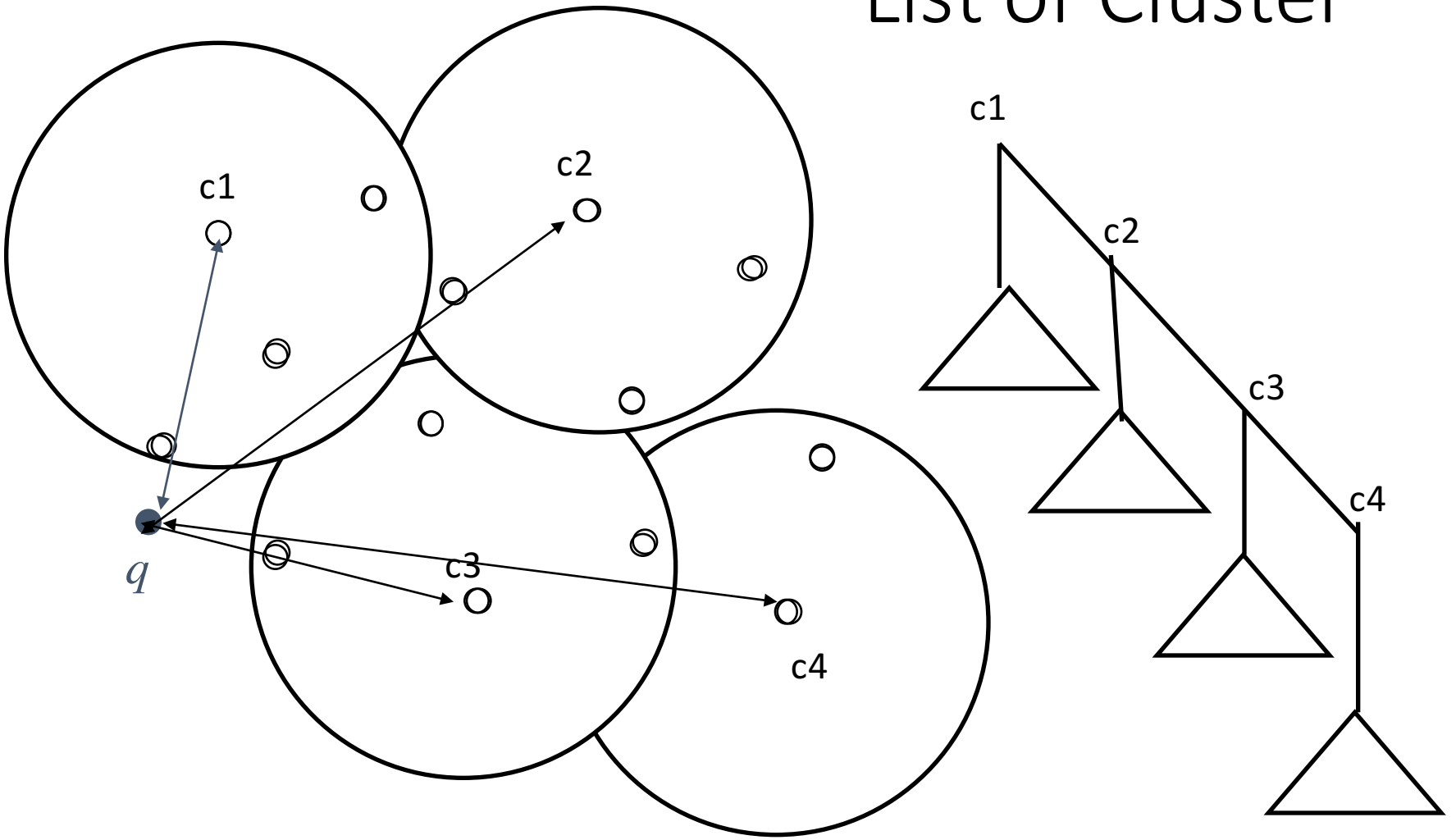
$$I_{X,c,r_c} = \{x \in X - \{c\}, d(c,x) \leq r_c\}$$

- $I_{X,\,c,\,r_c}$ cae dentro de la bola $(c, r_c)$.

- Los elementos externos son:

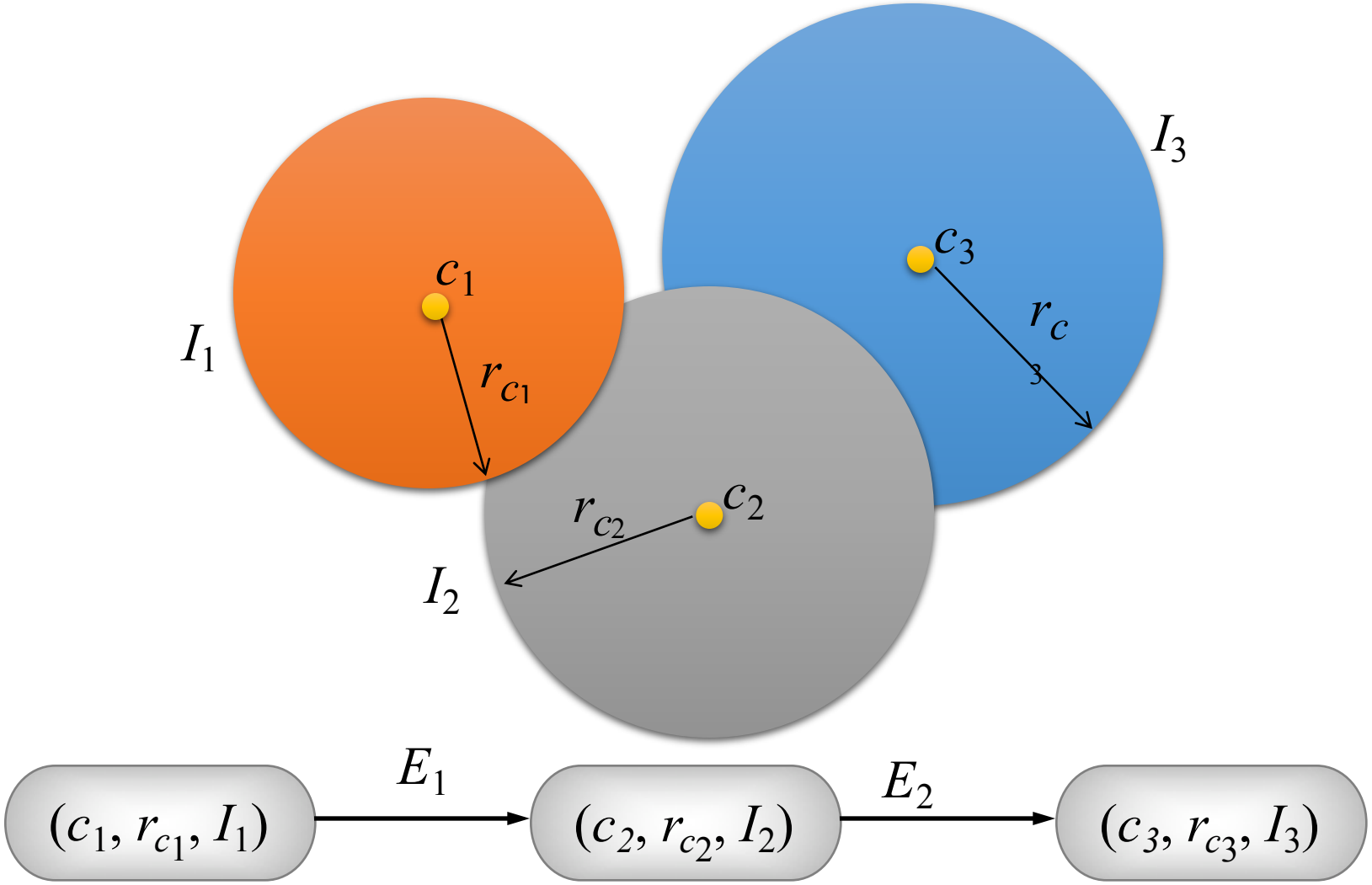
$$E_{X,c,r_c} = \{x \in X, d(c,x) > r_c\}$$

# LC
# List of Cluster




Espacio: $O(n)$

# LC: Construcción



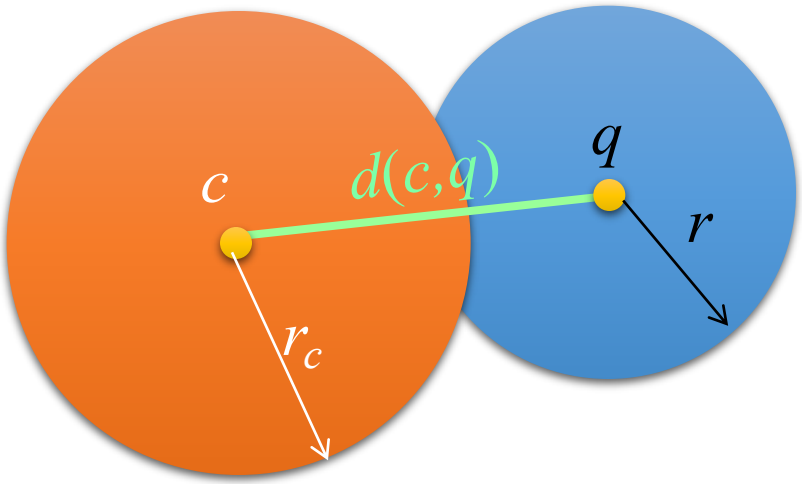
# LC: Centros

- Las diferentes heurísticas de selección de un nuevo centro son:
  - **P1**: aleatoria
  - **P2**: el elemento más cercano al centro anterior en el conjunto restante.
  - **P3**: el elemento más alejado al centro anterior en el conjunto restante.
  - **P4**: el elemento que minimice la suma de las distancias a los centros previos.
  - **P5**: el elemento que maximice la suma de las distancias a los centros previos.

# LC: Búsqueda

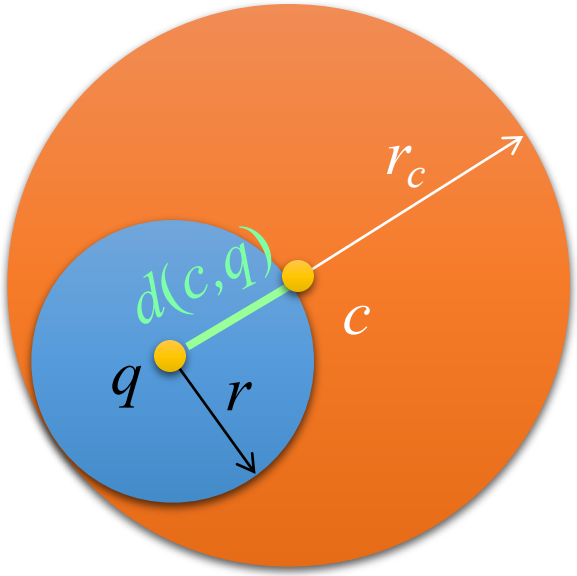
- La búsqueda debe considerar el orden en que se crearon los clusters.
- Por lo tanto, la búsqueda se inicia desde el primer cluster de la lista.
- En cada cluster puede suceder una de las siguientes situaciones:

# LC: Búsqueda



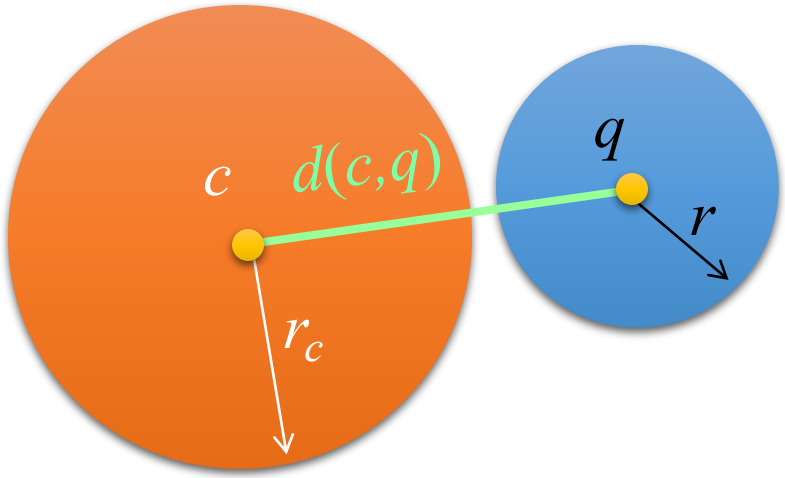

Se busca exhaustivamente en $I$, y se continúa buscando en $E$

# LC: Búsqueda




Se busca exhaustivamente en $I$, pero no se continúa buscando en $E$

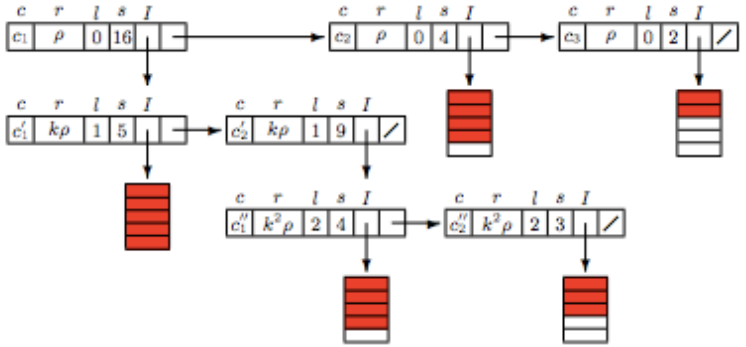
# LC: Búsqueda




No se busca en $I$, pero se continúa buscando en $E$

# Mejoras a la LC

- LC recursiva [Mamede, 2005]



- LC dinámica y para memoria secundaria
  - [Navarro and Reyes, 2014]

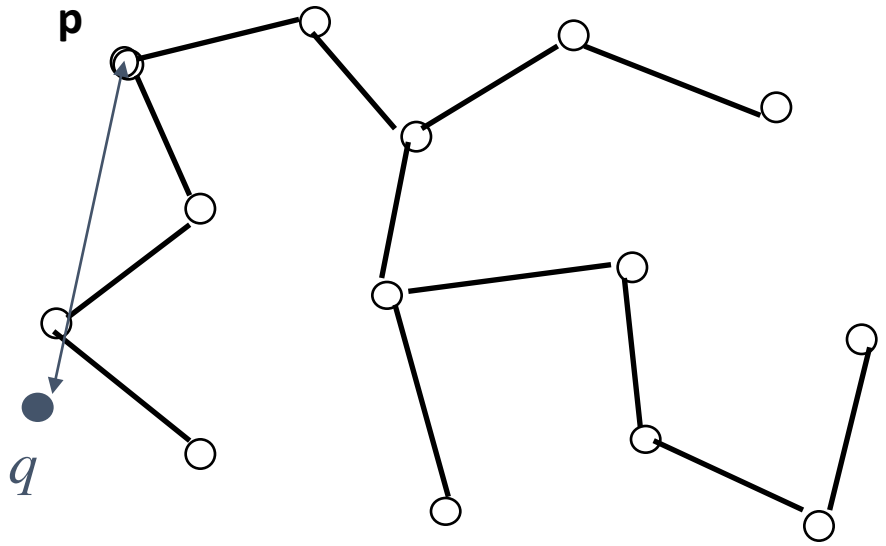
# SAT. Spatial Approximation Tree

G. Navarro. **Searching in metric spaces by spatial approximation**. In *String Processing and Information Retrieval (SPIRE'99)*, pages 141–148. IEEE CS Press, 1999.

# SAT

## Spatial Approximation Tree

Espacio: $O(n)$



# Mejoras al SAT

- SAT Dinámico. [Navarro, Reyes, 2001]
- SAT para memoria secundaria [Reyes]

# Cierre

- Algoritmos basados en pivotes
  - Principio, BKT, FQT, FHQT, FQA,
  - AESA
- Algoritmos basados en particiones compactas
  - Principio, BST, GHT, VPT, VPF, LC, SAT

# 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