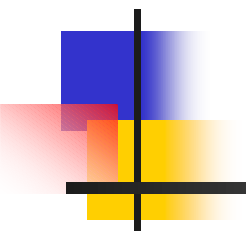
Modelos de Recuperación de la Información

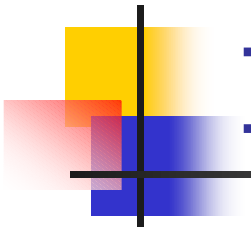


ARI – Curso 2002/2003

Introducción

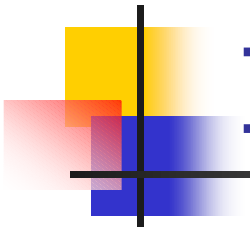
- 3 Modelos clásicos
  - Booleano
  - Vectorial
  - Probabilístico

- Normalmente nos basamos en términos para indexar y también para recuperar



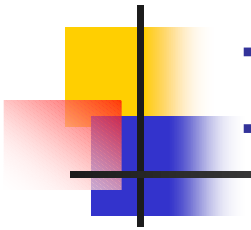
Introducción

- Los términos son palabras claves que representan al documento
  - Manualmente (mejores, pero alguien tiene que elegirlos)
  - Automáticamente
- Los términos no tienen porque aparecer en el documento



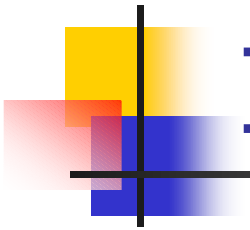
Introducción

- Problema:
  - El enfoque es una simplificación
  - Sólo tenemos aspecto léxico
  - No tenemos
    - Sintaxis
    - Semántica
    - Pragmática
- Los 3 modelos clásicos usan esta simplificación
- Los documentos se representan por un conjunto de términos de indexación.



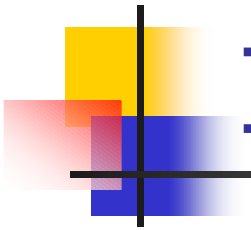
Introducción

- Modelo booleano
  - Los documentos son un conjunto de términos
  - Las preguntas son expresiones booleanas
- Modelo vectorial
- Modelo probabilístico



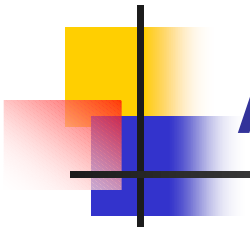
Introducción

- El modelo de representación de los documentos (D)
- Método de representación de las preguntas (P)
- Una función S: $D \times P \rightarrow \Re$
  - Para cada par (documento,pregunta) asigna un valor real de similitud



Introducción

- Estas 3 características determinan el núcleo de un SARI, ignorando el modo de uso
- D y P son conjuntos de términos que analizaremos en los temas 3 y 4
- En este tema nos centraremos en la función



Archivos

- Tenemos un archivo con todos los documentos
- Una solución es recorrer el archivo buscando palabras → Ineficiente.
- Otra forma es tener un registro para cada documento con un 0 ó 1 por cada término

Archivos

	T1	T2	T3	T4	T5
D1	0	0	0	1	0
D2	0	1	0	1	0
D3	1	0	1	1	0
D4	0	1	1	0	1

Archivo Directo

- Para saber si un documento tiene un término miramos en la tabla

Archivos

	D1	D2	D3	D4	D5
T1	0	0	0	1	0
T2	0	1	0	1	0
T3	1	0	1	1	0
T4	0	1	1	0	1

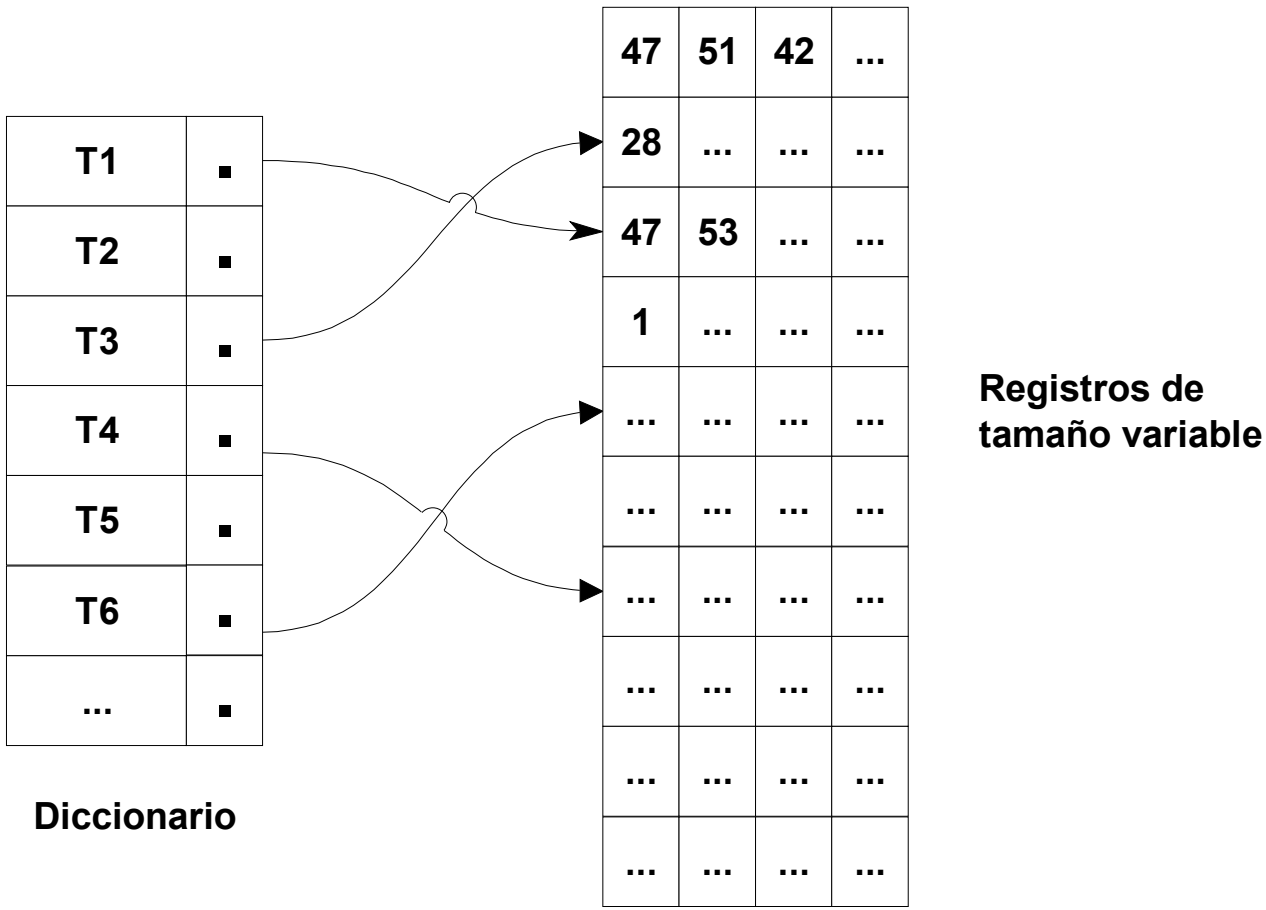
Archivo Invertido

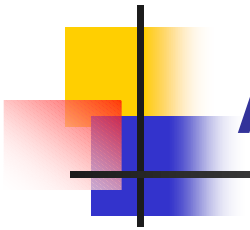
- Es más eficiente pues al buscar un término sólo miramos un registro

Archivos: Archivo Invertido

- El tamaño de los registros aumenta con el número de documentos
- La matriz está llena de 0’s
- Solución: Partir el archivo en 2:
  - Diccionario
  - Archivo de almacenamiento de listas (Postings file)

Archivos: Archivo Invertido



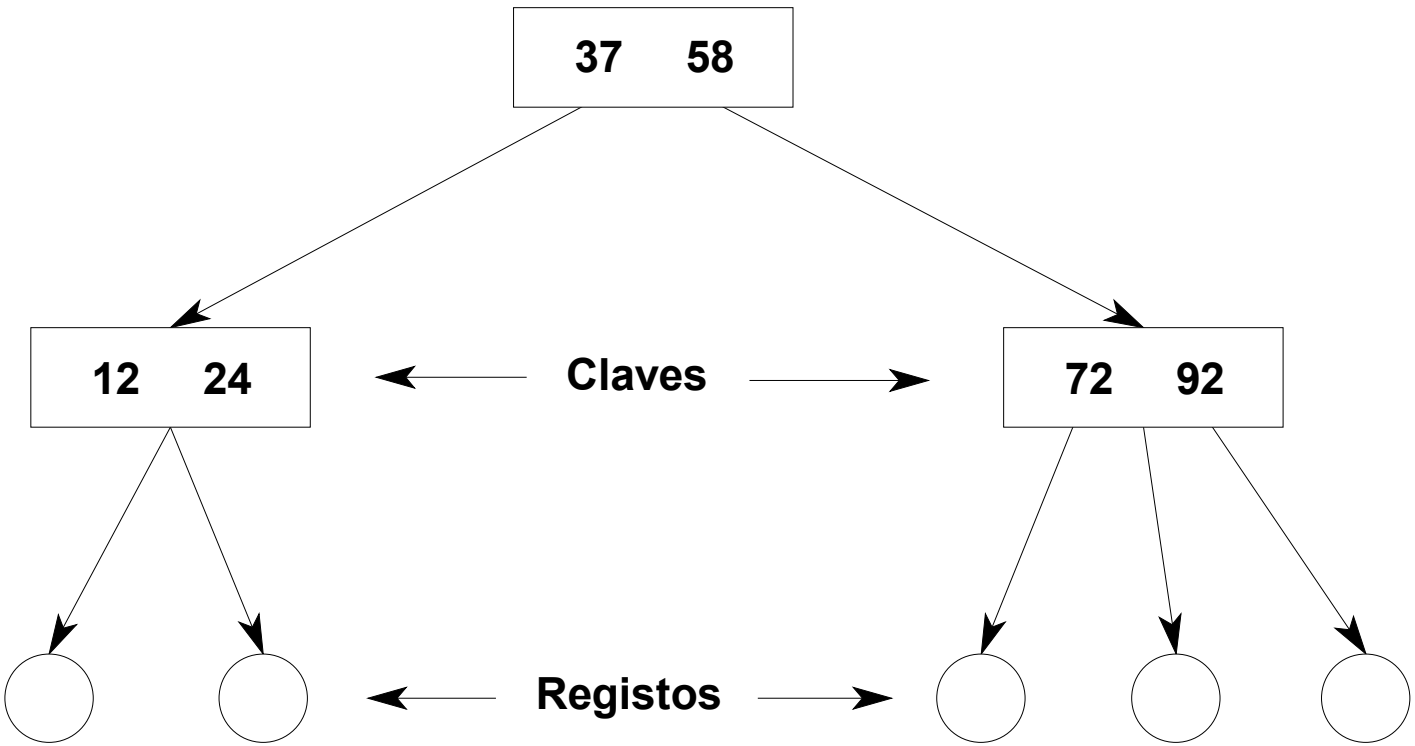


Archivos: Diccionario

- Implementación:
  - Orden de aparición
    - A medida que vamos añadiendo documentos vamos metiendo nuevos nuevos términos
    - Ineficiente
  - Tabla ordenada
    - El problema es que tiene que ser dinámica
  - Árboles B
    - Eficientes para implementación en disco

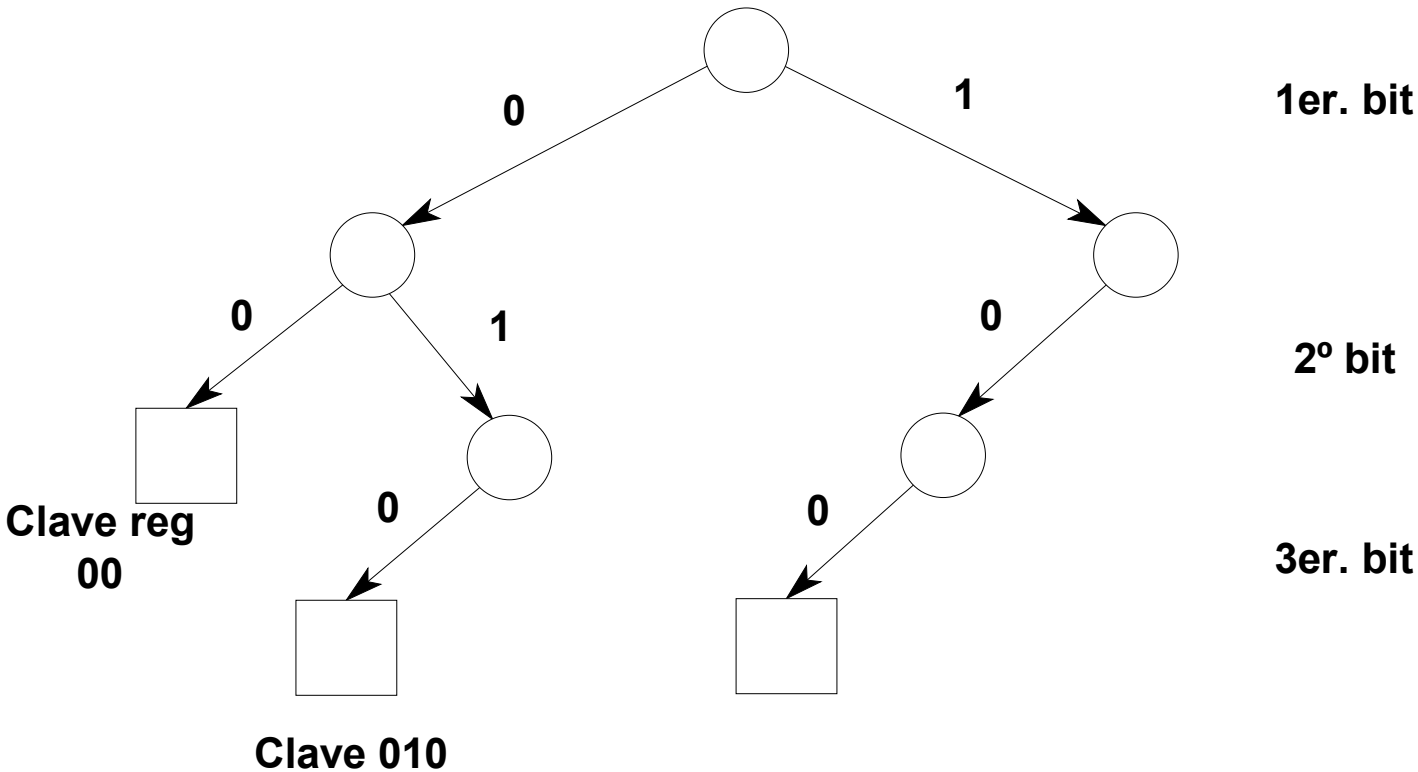
Archivos: Diccionario

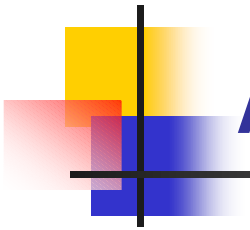
- Árbol B



Archivos: Diccionario

- Árboles digitales: TRIE





Archivos: Diccionario

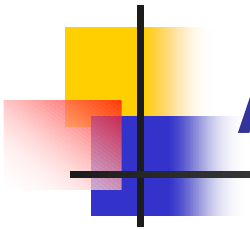
- Árboles digitales:
  - Las claves se construyen con 0's y 1's
  - Las claves están repartidas por árbol
  - Es más eficiente que los árboles B
  - Hay muchos tipos el de la figura anterior se llama TRIE
    - Traducción CUPE (reTRIEval → reCUPEración)
  - Son muy útiles cuando se hacen búsquedas por prefijos

Archivos: Diccionario

- Árboles digitales:
  - Árbol Patricia
    - Optimización del TRIE
    - Los nodos que sólo tienen un descendiente se eliminan y se indica el nivel que se salta

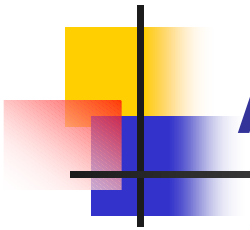
Archivos: Diccionario

- Tablas de dispersión (Tablas Hash)
  - Es la más eficiente
    - mejor tiempo de acceso
  - Problemas:
    - Si es muy dinámica y está muy llena:
      - Colisiones
      - Método de resolución de colisiones
  - Es la forma más utilizada



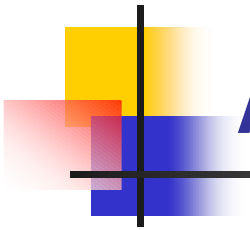
Archivos: Diccionario

- Tablas de dispersión (Tablas Hash)
  - La clave se utiliza como argumento de la función que nos dice donde está
  - La función la mayoría de las veces acierta
  - El índice de aciertos decrece a medida que la tabla se va llenando



Archivos: Diccionario

- Tablas de dispersión (Tablas Hash)
  - Cuando hay colisiones
    - Utilizar la clave donde debería estar como clave para la nueva localización
    - Tener un área de desbordamiento
  - Las búsquedas son muy eficientes
    - Es lo que más hacemos

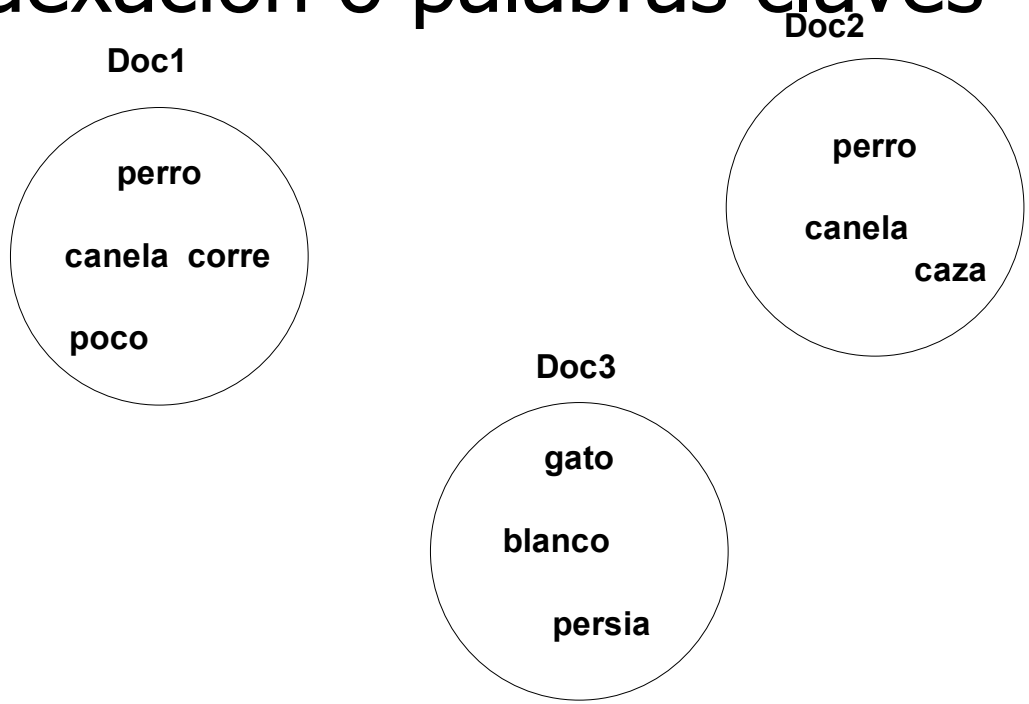


Archivos: Diccionario

- Tablas de dispersión (Tablas Hash)
  - Las inserciones dependen de lo llena que esté.
  - Un orden de llenado del 60% - 70% está bien
  - Problemas:
    - No hay orden lexicográfico
    - Claves desordenadas
    - Nos sabemos cual es el siguiente a un término

Modelo Booleano

- El método de representación de los documentos es un conjunto de términos de indexación o palabras claves



Modelo Booleano

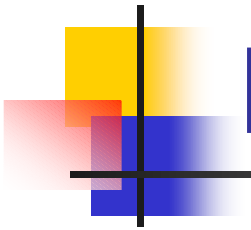
- Diccionario: Conjunto de todos los términos

  T = {t1, t2, t3, ...}

- Documento: Conjunto de términos del diccionario donde tiene valor

  Di = {t1, t2, t3, ...}

  ti = Verdad si es una palabra clave del doc

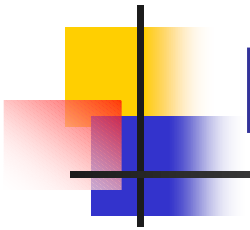


Modelo Booleano

- Las preguntas son expresiones booleanas cuyos componentes son términos de nuestro diccionario
- Operadores
  - O ($\cup$)
  - Y ($\cap$)
  - No (-)
    - (No suele implementarse, se suele implementar y_no)
- Ejemplo: (Perro o gato) y blanco

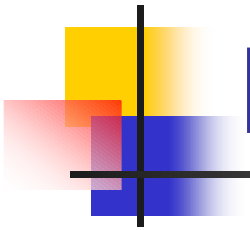
Modelo Booleano

- Función de similitud o semejanza
  - Sem(di,p) es verdad si p(di) = verdad
  - Sem(di,p) es falso si p(di) = falso
  - Ej:
    - sem(d1,p) = (perro o gato) y blanco = falso
    - Sem(d3,p) = (perro o gato) y blanco = verdad



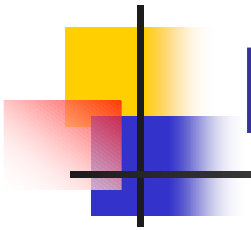
Modelo Booleano

- Ventajas:
  - Más sencillo imposible
- Desventajas:
  - La función semejanza sólo tiene 2 valores
  - El lenguaje de consulta no es sencillo



Modelo Booleano

- Algoritmo
  - Nos permite calcular el valor de la función de semejanza
  - 1ª aproximación: aplicar la función a todos los docs, pero esto no es eficiente
  - Necesitamos una función que nos devuelva los id de los docs que tienen un término
    - fácil mirando el archivo invertido
  - Luego mezclamos las listas



Modelo Booleano

- Algoritmo:
  - Entrada: 2 listas ordenadas ascendentemente
  - Salida: 1 lista ordenada con la mezcla de las 2 listas de entrada
    - El orden puede ser el número de identificación de documento

Modelo Booleano

- Algoritmo

```
MIENTRAS verdad
    SI ambas listas están vacias ENTONCES FIN
    SI_NO SI una lista de entrada está vacia
        ENTONCES
                transferir resto de elementos de la lista no vacia a salida
        FIN
        SI_NO tomar elemento de cabeza de L1(R1) y L2(R2)
            SI R1 < R2
                ENTONCES transferir R1 a salida y eliminarlo de L1
            SI_NO transferir R2 a salida y eliminarlo de L2
FIN_MIENTRAS
```

Modelo Booleano

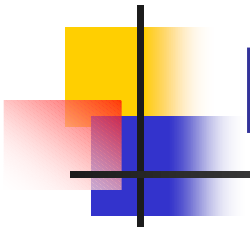
- Ti y Tj
  - Mezclamos las 2 listas y es verdad para los términos que estén duplicado en la mezcla
  - Ejemplo
    - d1=(t1,t3,t4) d2=(t1,t2,t4)
    - t1={d1,d2} t2={d2} t3={d1} t4={d1,d2}
    - t1 y t4: Mezcla={d1,d1,d2,d2}→{d1,d2}
    - t1 y t3: Mezcla={d1,d1,d2}→{d1}

Modelo Booleano

- Ti o Tj
  - Mezclamos las 2 listas y es verdad para los términos que estén 1 ó 2 veces
- No Ti
  - Los que no estén en la lista → Ineficiente
- Ti y no Tj
  - Hacemos Ti y Tj, Mezclamos(Ti, Ti y Tj) y quitamos los que aparecen más de una vez

Modelo Booleano

- Ejemplo
  - T1={d1,d3} T2={d1,d2} T3={d2,d3,d4}
  - P = (T1 o T2) y_no T3
    - Mezcla(T1,T2) = {d1,d1,d2,d3}
    - T1 o T2 = {d1,d2,d3} (Aparecen 1 o 2 veces)
    - Mezcla([T1 o T2],T3) = {d1,d2,d2,d3,d3,d4}
    - (T1 o T2) y T3 = {d2,d3} (Aparecen 2 veces)
    - Mezcla([T1 o T2],[(T1 o T2) y T3]) = {d1,d2,d2,d3,d3}
    - (T1 o T2) y_no T3 = {d1} (Quitando los duplicados)



Modelo Vectorial

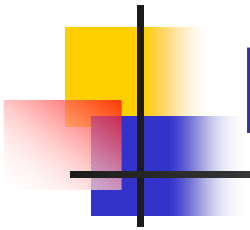
- Es el más usado
- Permite dar graduación a la pertenencia de un documento a una pregunta
- Los docs están representados por un pto en el espacio vectorial que construimos
- La pregunta es otro punto en el mismo espacio vectorial
- Diferencia con el booleano → El método de representación es el mismo para las preguntas y los documentos

Modelo Vectorial

- El espacio vectorial tiene tantas dimensiones como términos de indexación tiene el diccionario
- Cada elemento del vector indica el grado de importancia de los términos en el documento ($\Re+$)
  - di = (wi1,wi2,wi3,...,win)
    - n: nº de términos distintos de la colección
    - wij $\in \Re+$

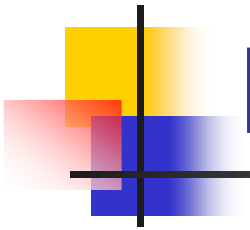
Modelo Vectorial

- Ejemplo
  - Diccionario (t1=perro, t2=gato, t3=azul
    t4=verde, t5=pequeño)
  - d1 = (perro, azul, pequeño) = (1,0,1,0,1)
  - d2 = (gato, verde) = (0,1,0,1,0)
    - Asumiendo peso = 1



Modelo Vectorial

- Preguntas
  - Igual que los documentos
  - P = (wp1, wp2, wp3, ..., wpn)
    - n: nº de términos distintos de la colección
    - wpj $\in \Re+$



Modelo Vectorial

- Función de semejanza
  - Tiene que ordenar los docs dependiendo de la proximidad con la pregunta
  - Una 1ª aproximación es el vector diferencia
    - No se usa
    - Los docs extensos tienen más términos y están más alejados
    - Las preguntas tienen pocos términos
    - Se penalizarían los docs largos

Modelo Vectorial

- Función de semejanza
  - Producto interno (Producto escalar)
    - Favorece los documentos largos pues al tener más términos suman más

$$sem(p, di) = \sum_{j=1}^{n} wpj \times wij$$

Modelo Vectorial

- Función de semejanza
  - Función Coseno
    - Normaliza los vectores respecto a su longitud
    - Si p y di son ortogonales la relevancia es 0
    - Si son paralelos es muy relevante

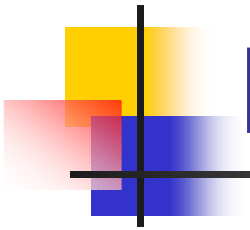
$$sem(p,di) = \frac{\sum_{j=1}^{n} wpj \times wij}{\sqrt{\sum_{j=1}^{n} wpj^2}\sqrt{\sum_{j=1}^{n} wij^2}} = \cos(\alpha)$$

Modelo Vectorial

- Calculo automático de pesos
  - wij: peso del término j en el doc i
  - ftij: frecuencia del término j en el doc i
  - n: nº de términos
  - fdj: nº de docs que tienen el término j
  - d: nº de docs
  - fidj = log(d/fdj) (Frecuencia inversa)
    - Mínimo=0 fdj=d (tj aparece en todos los docs)
    - Máximo=log(d) fdj=1 (tj sólo aparece en 1 doc)

Modelo Vectorial

- Calculo automático de pesos
  - wij = ftij · fidj
  - Frecuencia del término en el doc X frecuencia inversa
  - Lo importante que es el término en el doc X lo importante que es el término en la colección
  - Para las preguntas calculamos wpj igual que wij

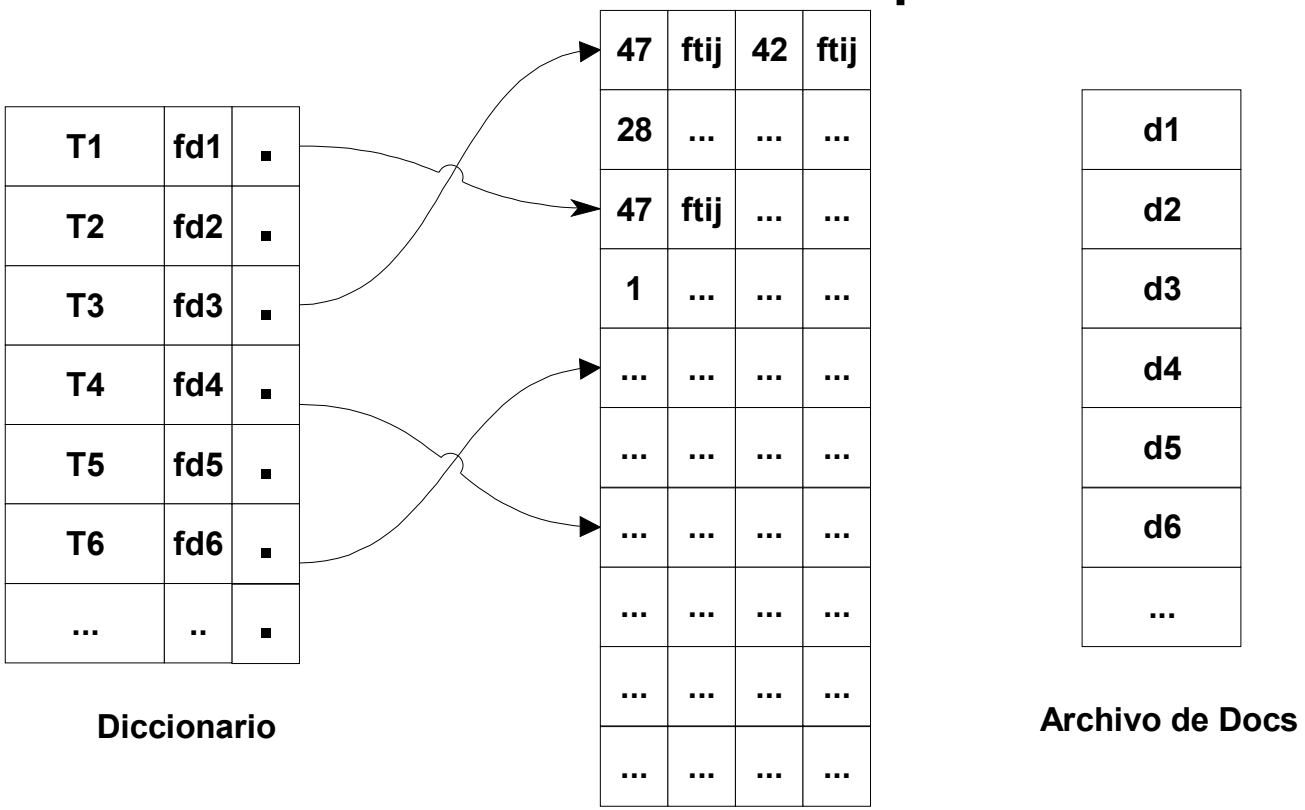


Modelo Vectorial

- Calculo automático de pesos
  - Implementación
    - En los docs se almacena la frec de aparición de cada término
    - La frec inversa se deja como característica de los términos (depende de la colección)
    - Necesitamos d (nº de docs) pero es un entero que depende de la colección
    - Lo más eficaz es almacenar todo en el archivo invertido

Modelo Vectorial

- Calculo automático de pesos



Modelo Vectorial

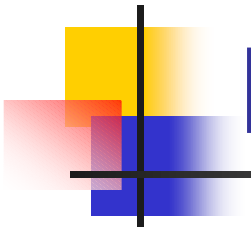
- Cálculo de la semejanza
  - El método típico para el cálculo de la semejanza es extraer los documentos que tienen alguno de los términos de la pregunta
  - Así solo calculamos la semejanza para unos pocos documentos

Modelo Probabilístico

- La base de cálculo es la probabilidad de un documento de ser relevante a una pregunta dada

- La función de semejanza es la probabilidad de que un doc sea relevante

$$Sem(p,di) = P(R|di)$$



Modelo Probabilístico

- Utilizaremos el modelo probabilístico de independencia de términos binarios
  - La probabilidad de los términos es independiente (un término es independiente de los otros)
  - Los pesos asignados a los términos son binarios

Modelo Probabilístico

- Representación igual al modelo booleano
  - di={wi1, wi2, ..., win}
  - wij= {1 si tj es término de di, 0 si no}
- Las preguntas son un subconjunto de términos

Modelo Probabilístico

- Para la función de semejanza es mejor coger como relevante aquellos docs en los que su probabilidad de ser relevante es mayor que la de no serlo

$$P(R \mid di) > P(\overline{R} \mid di)$$

$$sem(p, di) = \frac{P(R \mid di)}{P(\overline{R} \mid di)} \rightarrow \begin{cases} > 1 & relevante \\ < 1 & no \quad relevante \end{cases}$$

Modelo Probabilístico

- Aplicando el Teorema de Bayes

$$sem(p, di) \approx \frac{P(di \mid R)}{P(di \mid \overline{R})}$$

- P(di|R) es la probabilidad de que dado el conjunto de relevantes di esté dentro

Modelo Probabilístico

- Simplificando...

$$sem(p, di) = \sum_{j=1}^{m} w_{pj} w_{ij} \log \frac{p_j (1 - q_j)}{q_j (1 - p_j)}$$

  - pj es la probabilidad de que tj esté en el conjunto de docs relevantes
  - qj es la probabilidad de que tj esté en el conjunto de docs no relevantes

Modelo Probabilístico

- Si conociéramos R:

		Relevancia		
		SI	**NO**	
Término j	**SI**	r	n-r	n
	NO	R-r	N-R-n+r	N-n
		R	N-R	N

- N = Nº docs colección
- R = Nº docs relevantes
- r = Nº docs relevantes que tienen tj
- n = Nº docs colección que tiene tj

Modelo Probabilístico

- Si conociéramos R:
  - pj: probabilidad de que cogiendo un doc relevante tenga tj

$$pj = (r/R)$$

  - qj: probabilidad de que en los docs no relevantes no esté tj

$$qj = (n-r)/(N-R)$$

Modelo Probabilístico

- Si conociéramos R:
  - Substituimos pj y qj

$$sem(p, di) \approx \sum \log \frac{\frac{r}{R-r}}{\frac{n-r}{N-R-n+r}}$$

Modelo Probabilístico

- Como no conocemos R
  - La prob de que un término esté en el conj de docs relevantes es la misma para todos los términos. A priori no hay un término más relevante que otro

$$pj=0'5$$

  - Para qj se usa la frec inversa del término en la colección

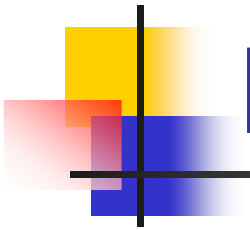
$$qj=fdj/N$$

fdj: Nº de docs con el término tj

N: Nº de docs de la colección

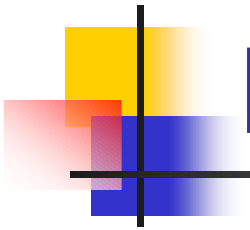
Modelo Probabilístico

- Como no conocemos R:
  - Usamos pj=0’5 y qj=fdj/N
  - Recuperamos los N primeros y consideramos que son relevantes
  - Preguntamos al usuario y recalculamos pj y qj



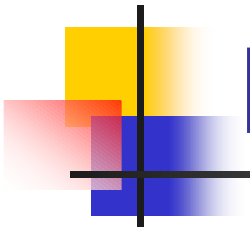
Modelos de navegación

- La necesidad de información no se expresa por una pregunta
- En un sistema real tenemos una mezcla:
  - Hago una pregunta
  - Exploro los resultados
  - Reformulo la pregunta...
- El interfaz de usuario es más importante



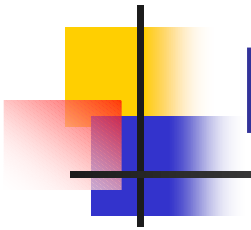
Modelos de navegación

- Navegación directa
  - Los docs están normalmente almacenados en una lista sin un criterio de ordenación útil o agrupados por su proximidad semántica
  - La carga de exploración corre por parte del usuario
  - Para agruparlos se suele utilizar una función de semejanza entre los docs
  - El criterio de semajanza es muy difuso → que estén próximos



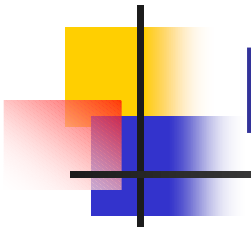
Modelos de navegación

- Navegación guiada por una estructura
  - Los docs están organizados por **UN** criterio
  - Este criterio genera una estructura jerárquica
  - Los temas de la raíz son genéricos y las hojas específicos
  - Se usa en bibliotecas y centros de documentación



Modelos de navegación

- Navegación guiada por hipertexto
  - Los docs están organizados por un sistema multicriterio
  - Obtenemos un grafo
  - Los docs son nodos cualesquiera del grafo
  - Es el sistema de moda



Modelos Avanzados

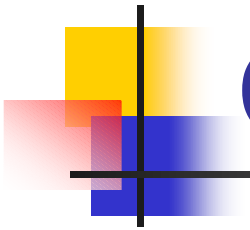
- Son extensiones del modelo booleano
- El modelo booleano no permite dar orden al conjunto de docs relevantes
- El modelo booleano sólo permite decir si son relevantes o no
- En las extensiones se permite dar un orden

Conjuntos Difusos

- Se basa en que la representación de un doc por un conjunto de término no sea categórica
- Se difumina la pertenencia para que no sea binaria
- F(D,t)=0 ó 1 $\Rightarrow$ f(D,t) $\in$ [0,1] (Función de pertenencia)

Conjuntos Difusos

Fórmula lógica	**Fórmula de evaluación**
f(di,tj y tk)	F(di,tj) * F(di,tk) ≡ **min(F(di,tj),F(di,tk)**
f(di,tj o tk)	F(di,tj) + F(di,tk) – (F(di,tj) * F(di,tk)) ≡ **max(F(di,tj),F(di,tk)**
f(di,tj y_no tk)	**F(di,tj) * (1 – F(di,tk))**



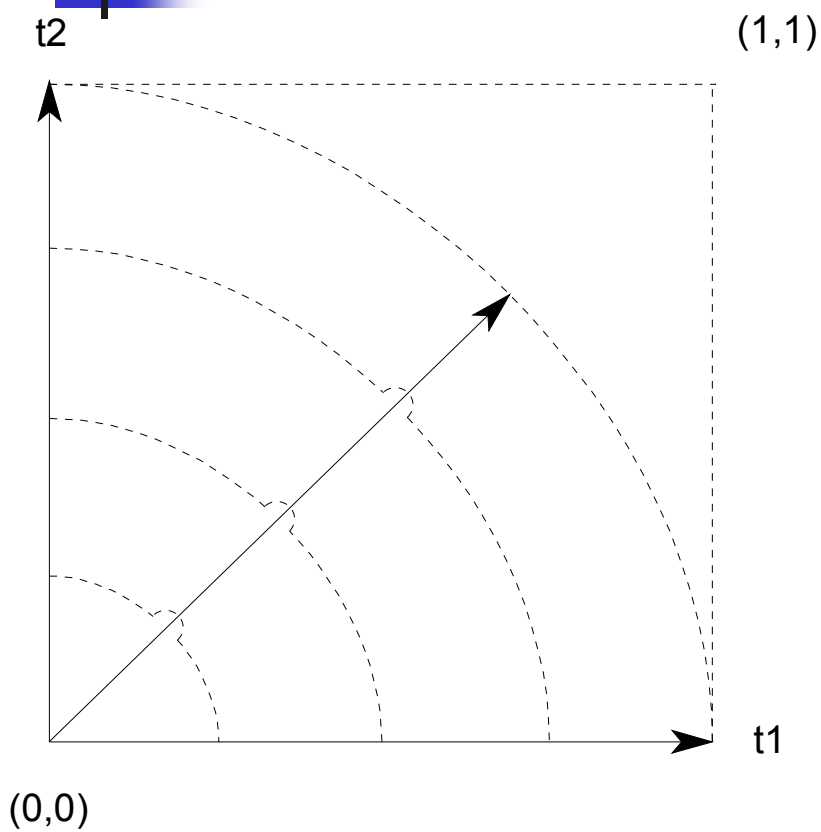
Conjuntos Difusos

- Cálculo de pesos
  - Generalmente se hacer como en el modelo vectorial
    - wij = fdij * fidj
- Cuando las funciones sólo toman 0 ó 1 degenera en el modelo booleano

Conjuntos Difusos

- Desventajas:
  - Es menos flexible de lo que parece
    - No podemos dar pesos a las preguntas
    - La ordenación puede estar desfigurada pues sólo algunos términos influyen en la semejanza
      - Ej: d1={(t3,0'8)} d2={(t1,0'7),(t2,0'7),(t3'08)}
      - P=t1 o t2 o t3
      - A los dos les da 0'8 (Máximo) aun cuando el d2 sería más relevante

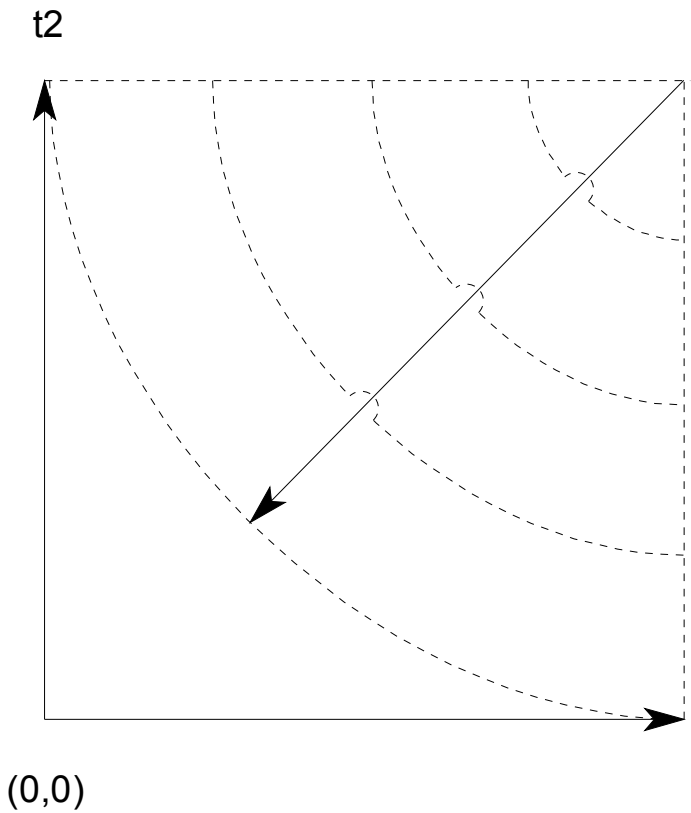
Booleano Extendido



- Introduce el concepto de distancia
- P=(t1 o t2)
- Cuanto más cerca del 0 menos representa al doc

$$sem(d, t_1 \circ t_2) = \sqrt{\frac{t_1^{\,2} + t_2^{\,2}}{2}}$$

Booleano Extendido



- P=(t1 y t2)
- Cuanto más cerca del 0 más representa al doc

$$sem(d, t_1 y t_2) = 1 - \sqrt{\frac{(1-t_1)^2 + (1-t_2)^2}{2}}$$

Booleano Extendido

- Aquí todos los términos contribuyen
- Introducimos los pesos

P=(wpj,wpk) di=(wij,wik)

$$sem(d, t_j \circ t_k) = \sqrt{\frac{w_{pj}^2 w_{ij}^2 + w_{pk}^2 w_{ik}^2}{w_{pj}^2 + w_{pk}^2}}$$

$$sem(d, t_j y t_k) = 1 - \sqrt{\frac{w_{pj}^2 (1 - w_{ij})^2 + w_{pk}^2 (1 - w_{ik})^2}{w_{pj}^2 + w_{pk}^2}}$$

Booleano Extendido

- Podemos dar peso a los operandos (p)

$$sem(d, t_j \circ^p t_k) = \sqrt[p]{\frac{w_{pj}{}^p w_{ij}{}^p + w_{pk}{}^p w_{ik}{}^p}{w_{pj}{}^p + w_{pk}{}^p}}$$

$$sem(d, t_j y^p t_k) = 1 - \sqrt[p]{\frac{w_{pj}{}^p (1 - w_{ij})^p + w_{pk}{}^p (1 - w_{ik})^p}{w_{pj}{}^p + w_{pk}{}^p}}$$

$$sem(di, t_j y\_no^p t_k) = w_{pj} \cdot w_{ij} \cdot w_{pk} \cdot (1 - w_{ik})$$

Booleano Extendido

- Si p=∞

$$sem(d, t_j \circ^{\infty} t_k) = \lim_{p \to \infty} \left(\sqrt[p]{\frac{w_{pj}{}^{p} w_{ij}{}^{p} + w_{pk}{}^{p} w_{ik}{}^{p}}{w_{pj}{}^{p} + w_{pk}{}^{p}}} \right)$$

$$sem(d, t_j \circ^{\infty} t_k) = \frac{\max\left(w_{pj} w_{ij}, w_{pk} w_{ik}\right)}{\max\left(w_{pj}, w_{pk}\right)}$$

$$sem(d, t_j y^{\infty} t_k) = 1 - \frac{\max\left(w_{pj}(1 - w_{ij}) + w_{pk}(1 - w_{ik})\right)}{\max\left(w_{pj} + w_{pk}\right)}$$

Booleano Extendido

- Si $p=\infty$ y $wpj=wpk=1$
  - Sem(di, tj o tk) = max(wij,wik)
  - Sem(di, tj y tk) = 1 – min(wij,wik)
- Es lo mismo que el modelo de conjuntos difusos
- El modelo booleano extendido engloba al modelo de conjuntos difusos

Booleano Extendido

- Si p=1

$$sem(d, t_j \circ^1 t_k) = \frac{w_{pj} w_{ij} + w_{pk} w_{ik}}{w_{pj} + w_{pk}}$$

$$sem(d, t_j y^1 t_k) = 1 - \frac{w_{pj}(1 - w_{ij}) + w_{pk}(1 - w_{ik})}{w_{pj} + w_{pk}}$$

$$sem(d, t_j y^1 t_k) = \frac{w_{pj} w_{ij} + w_{pk} w_{ik}}{w_{pj} + w_{pk}}$$

Booleano Extendido

- Si p=1
  - No hay diferencia entre “y” y “o”
  - La ordenación es igual a la del modelo vectorial
  - El modelo booleano extendido engloba también al modelo vectorial
- Si $1<p<\infty$
  - Situaciones intermedias
  - Con $p \rightarrow \infty$ nos acercamos al booleano estándar
  - Con $p \rightarrow 1$ nos acercamos al modelo vectorial

Booleano Extendido

$$P = \left(\left(A, a \right) y^{2} \left(B, b \right) \right) o^{\infty} \left(C, c \right)$$

- Pregunta: Quiero que el “y” se asemeje al modelo vectorial y el “o” al modelo booleano

Booleano Extendido

- Ventajas
  - Modelo generalista
  - Engloba a muchos otros
- Desventajas
  - Las leyes booleanas (asociativa, distributiva,...) no se cumplen
    - Ej: sem((A y B) y C) ≠ sem(A y (B y C))
  - Coste computacional es muy alto (Normalmente solo se permite p=1, 2 e ∞)
  - Los usuarios no se sienten a gusto con la formulación (Por defecto p=2)