

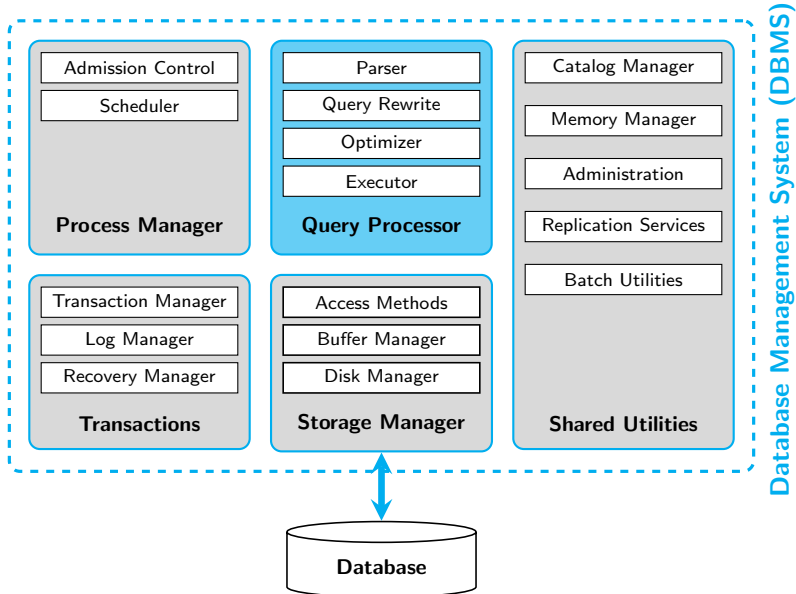
Consultas acíclicas

Clase 18

IIC 3413

Prof. Cristian Riveros

Evaluación eficiente de consultas conjuntivas



¿cuáles son las consultas conjuntivas **difíciles**?

Ejemplo: consultas con ciclos

$$ans(x, y, z) := R(x, y), S(y, z), T(z, x)$$

con las siguientes relaciones:

<i>R</i>	<i>x</i>	<i>y</i>	<i>S</i>	<i>y</i>	<i>z</i>	<i>T</i>	<i>z</i>	<i>x</i>
	0	a		0	a		0	a
	0	b		0	b		0	b
	1	a		1	a		1	a
	1	b		1	b		1	b
	a	0		a	0		a	0
	a	1		a	1		a	1
	b	0		b	0		b	0
	b	1		b	1		b	1

- ¿cuál es el tamaño de sus relaciones intermedias?
- ¿cuál es el tamaño del resultado total?

¿cómo definimos cuando una consulta **no tiene ciclos** (acíclica)?

Outline

Consultas acíclicas

Árbol de join

Algoritmo de Yannakakis

Outline

Consultas acíclicas

Árbol de join

Algoritmo de Yannakakis

Algunas simplificaciones

Para simplificar el análisis supondremos una consulta conjuntiva:

$$Q: \text{ans}(\bar{y}) := R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)$$

- cada \bar{x}_i no contiene constantes ni variables repetidas,
- \bar{y} contiene todas las variables en Q , y
- cada $R_i \neq R_j$ para $i \neq j$ (no hay self-joins).

Ejemplo

$$\text{ans}(x, y, z) := R(x, y), S(y, z), T(z, x)$$

¿por qué podemos hacer esta simplificación?

Hipergrafo de una consulta conjuntiva

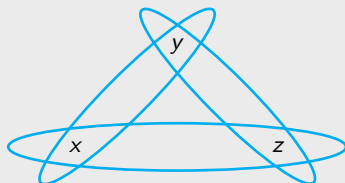
Definición

Para una CQ $Q = R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)$ definimos su **hipergrafo** $\mathcal{H}_Q = (V, E)$:

- $V = \{x \mid \exists i. x \text{ es una variable en } R_i\}$.
- $E = \{\{x_1, \dots, x_k\} \subseteq 2^V \mid \exists i. x_1, \dots, x_k \text{ son todas las variables de } R_i\}$.

Ejemplo

El **hipergrafo** para $R(x, y), S(y, z), T(z, x)$:



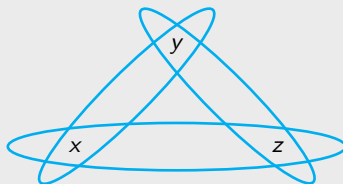
¿cómo definimos aciclicidad para un hipergrafo?

Ejemplos

- $R(x, y), S(y, z), T(z, w)$



- $R(x, y), S(y, z), T(z, x)$

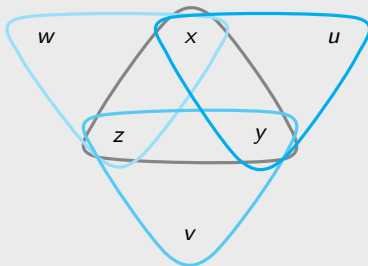


¿cómo definimos aciclicidad para un hipergrafo?

Ejemplos

¿qué sucede con esta consulta?

$$R(x, y, z), S(x, y, u), T(y, z, v), U(z, x, w)$$



Una **oreja** de un hipergrafo



Definición

Para un hipergrafo $\mathcal{H} = (V, E)$, una hiper-arista $O \in E$ es una **oreja** si existe una hiper-arista $C \in E$ (la “cara”) tal que para todo $x \in O$, una de dos:

- x esta solo en O (i.e. $x \in O$ y $x \notin H$ para todo $H \in (E - \{O\})$), o
- x aparece en C .

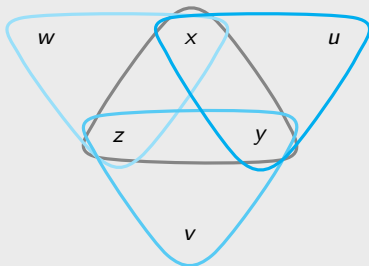
Una oreja de un hipergrafo

Definición

Para un hipergrafo $\mathcal{H} = (V, E)$, una hiper-arista $O \in E$ es una **oreja** si existe una hiper-arista $C \in E$ (la “cara”) tal que para todo $x \in O$, una de dos:

- x esta solo en O (i.e. $x \in O$ y $x \notin H$ para todo $H \in (E - \{O\})$), o
- x aparece en C .

Ejemplo



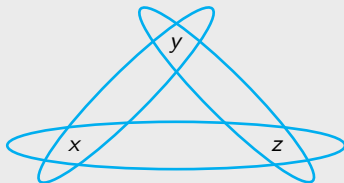
Una oreja de un hipergrafo

Definición

Para un hipergrafo $\mathcal{H} = (V, E)$, una hiper-arista $O \in E$ es una oreja si existe una hiper-arista $C \in E$ (la “cara”) tal que para todo $x \in O$, una de dos:

- x esta solo en O (i.e. $x \in O$ y $x \notin H$ para todo $H \in (E - \{O\})$), o
- x aparece en C .

Ejemplo



Eliminación de orejas

Definición

Para un hipergrafo $\mathcal{H} = (V, E)$, la **eliminación de una oreja** O produce el hipergrafo $\mathcal{H}' = (V', E')$ tal que:

$$\begin{aligned} E' &= E - \{O\} \\ V' &= \bigcup_{X \in E'} X \end{aligned}$$

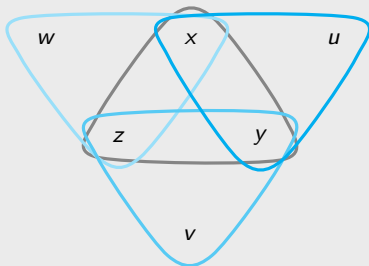
¿qué ocurre con las otras orejas al eliminar una oreja?

Hipergrafos acíclicos y orejas

Definición

Un hipergrafo es **acíclico** si es posible reducirlo a **una sola hiper-arista** por medio de eliminación de orejas.

¿és este hipergrafo acíclico?

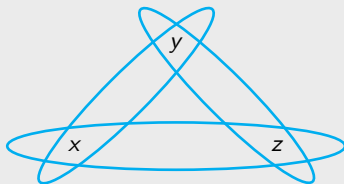


Hipergrafos acíclicos y orejas

Definición

Un hipergrafo es **acíclico** si es posible reducirlo a **una sola hiper-arista** por medio de eliminación de orejas.

¿és este hipergrafo acíclico?



Hipergrafos acíclicos y orejas

Definición

Un hipergrafo es **acíclico** si es posible reducirlo a **una sola hiper-arista** por medio de eliminación de orejas.

Teorema

Un grafo es acíclico si, y solo si,
para todo orden de eliminación de orejas el resultado final es una sola arista.

Esto nos permite verificar eficientemente
si la consulta es acíclica

...¿cómo?

Outline

Consultas acíclicas

Árbol de join

Algoritmo de Yannakakis

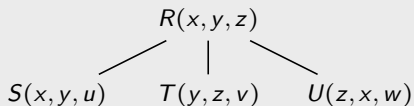
Árboles de joins o join trees

Definición

Un **join tree** para $Q = R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)$ es un árbol $T = (N, A)$ tal que:

- $N = \{R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)\}$ y
- Para toda variable x , el subgrafo de T de todos los átomos de Q que contienen a x , es **conexo**.

Ejemplo: $R(x, y, z), S(x, y, u), T(y, z, v), U(z, x, w)$



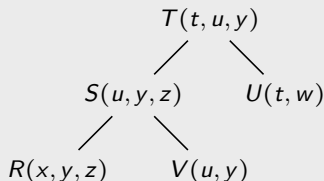
Árboles de joins o join trees

Definición

Un **join tree** para $Q = R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)$ es un árbol $T = (N, A)$ tal que:

- $N = \{R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)\}$ y
- Para toda variable x , el subgrafo de T de todos los átomos de Q que contienen a x , es **conexo**.

Ejemplo: $R(x, y, z), S(u, y, z), T(t, u, y), U(t, w), V(u, y)$



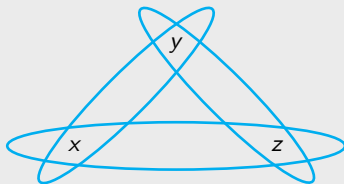
Árboles de joins o join trees

Definición

Un **join tree** para $Q = R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)$ es un árbol $T = (N, A)$ tal que:

- $N = \{R_1(\bar{x}_1), \dots, R_n(\bar{x}_n)\}$ y
- Para toda variable x , el subgrafo de T de todos los átomos de Q que contienen a x , es **conexo**.

Ejemplo: $R(x, y), S(y, z), T(z, x)$



¿es posible encontrar un árbol de join para consultas no acíclicas?

Consultas acíclicas y árbol de join

Teorema

Una consulta Q es acíclica si, y solo si, Q tiene un árbol de join.

Demostración: pizarra.

Outline

Consultas acíclicas

Árbol de join

Algoritmo de Yannakakis

Semi-joins

Definición

Un **semijoin** $R_1(\bar{x}) \ltimes R_2(\bar{y})$ se define como:

$$R_1 \ltimes R_2 := \pi_{\bar{x}} (R_1 \bowtie R_2)$$

Ejemplo

T	t	u	y		S	u	y	z
	1	2	3	\ltimes		2	3	9
	2	4	7			3	5	9
	1	4	6			1	2	3
	1	3	5			1	5	9

Un semi-join deja todas las tuplas en R_1 **que hacen match** con R_2 .

Semi-joins

Definición

Un **semijoin** $R_1(\bar{x}) \ltimes R_2(\bar{y})$ se define como:

$$R_1 \ltimes R_2 := \pi_{\bar{x}}(R_1 \bowtie R_2)$$

Ejemplo

T	t	u	y		S	u	y	z
	1	2	3	\ltimes		2	3	9
	2	4	7			3	5	9
	1	4	6			1	2	3
	1	3	5			1	5	9

Un semi-join deja todas las tuplas en R_1 **que hacen match** con R_2 .

Algoritmo de Yannakakis

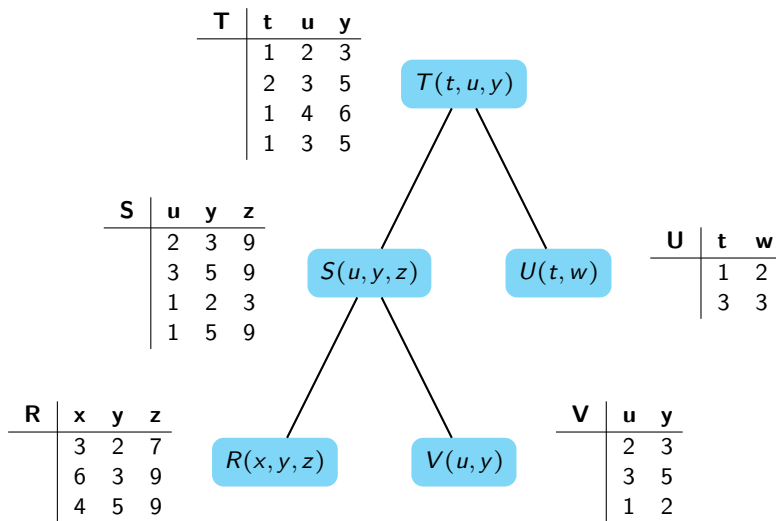
En dos pasos:

1. Desde las hojas hasta la raíz, actualizar cada relación R de la forma:

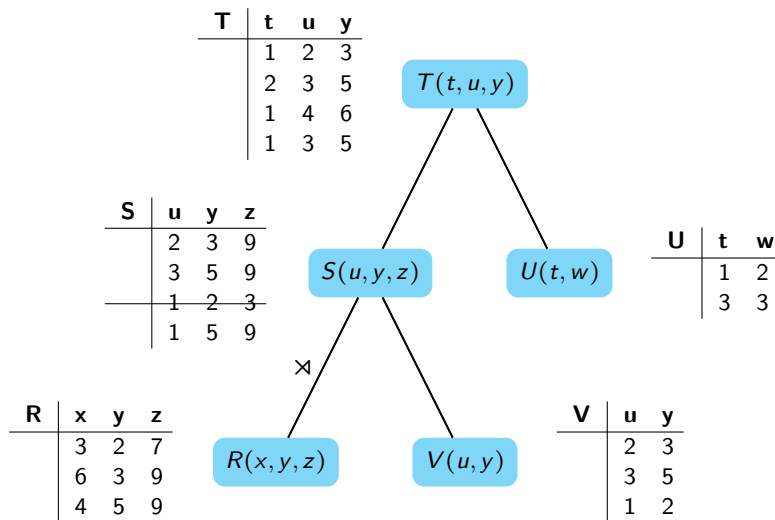
$$R := R \ltimes S$$

para cada hijo S de R en el árbol de join de Q .

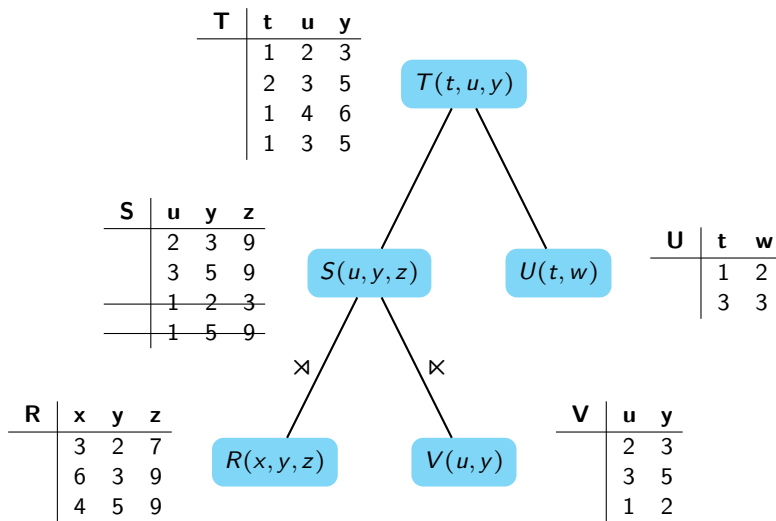
Algoritmo de Yannakakis: Paso 1.



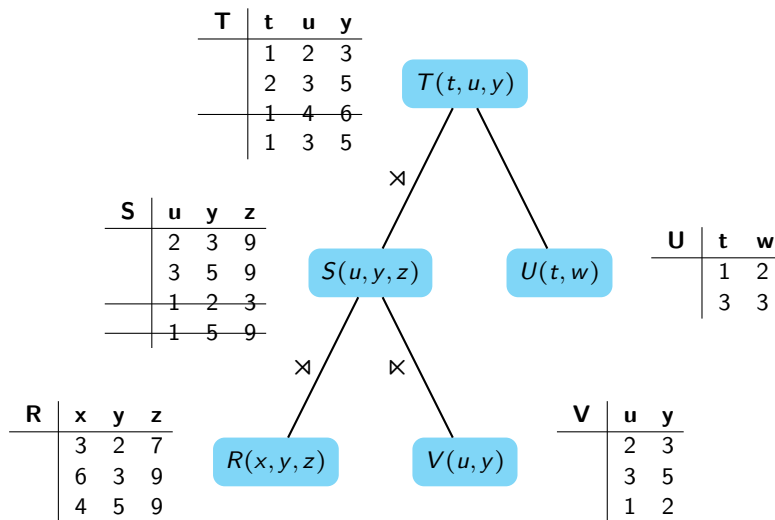
Algoritmo de Yannakakis: Paso 1.



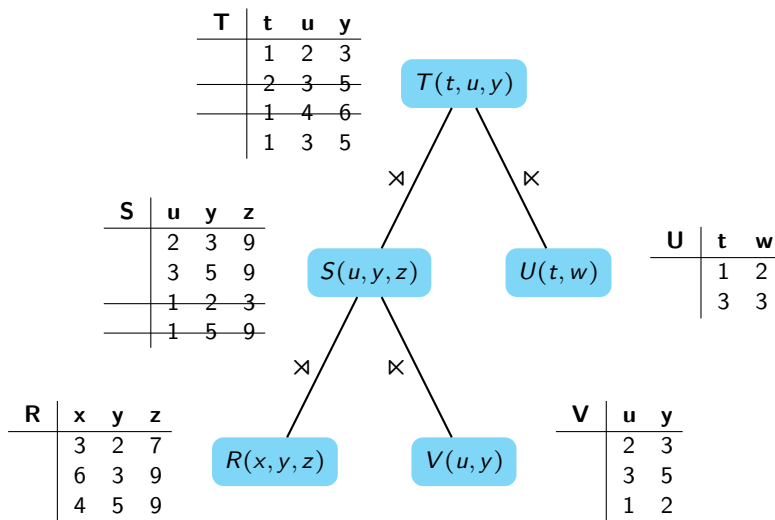
Algoritmo de Yannakakis: Paso 1.



Algoritmo de Yannakakis: Paso 1.



Algoritmo de Yannakakis: Paso 1.



Algoritmo de Yannakakis

En dos pasos:

1. Desde las hojas hasta la raíz, actualizar cada relación R de la forma:

$$R := R \bowtie S$$

para cada hijo S de R en el árbol de join de Q .

2. Si R_1, R_2, \dots, R_n es el **preorden** de los nodos del árbol de join:

foreach $t_1 \in R_1$ **do**

foreach $t_2 \in R_2$ *and* $t_1 \bowtie t_2 = \text{TRUE}$ **do**

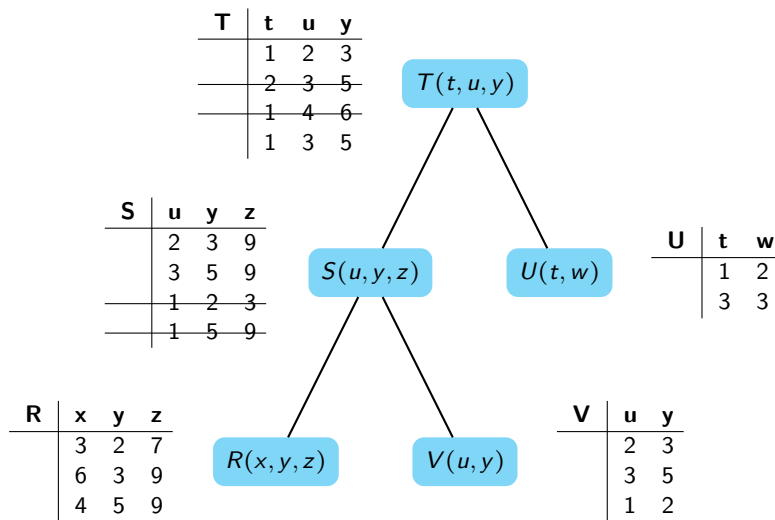
foreach $t_3 \in R_3$ *and* $t_1 \bowtie t_2 \bowtie t_3 = \text{TRUE}$ **do**

\vdots

foreach $t_n \in R_n$ *and* $t_1 \bowtie t_2 \bowtie \dots \bowtie t_n = \text{TRUE}$ **do**

enumerate $t_1 \bowtie t_2 \bowtie \dots \bowtie t_n$

Algoritmo de Yannakakis: Paso 2.



Algoritmo de Yannakakis: correctitud

Teorema

Dado una consulta acíclica Q y una base de datos \mathcal{D} , el algoritmo de Yannakakis enumera todas las tuplas en $Q(\mathcal{D})$ tal que toma:

- tiempo polinomial en Q y \mathcal{D} para entregar la **primera tupla** de $Q(\mathcal{D})$, y
- tiempo polinomial en Q y \mathcal{D} entre cada **siguiente tupla** de $Q(\mathcal{D})$.

Tiempo de algoritmo de Yannakakis

Sea $Q : R_1, \dots, R_n$ una consulta acíclica y
 $|R|$ el tamaño de la relación más grande en la BD \mathcal{D} .

Tiempo Paso 1: $\mathcal{O}(|Q| \cdot |R| \cdot \log(|R|))$

Tiempo Paso 2: $\mathcal{O}(|Q| \cdot |R| \cdot |Q(\mathcal{D})|)$

Tiempo total: $\mathcal{O}(|Q| \cdot |R| \cdot (\log(|R|) + |Q(\mathcal{D})|))$

¿es posible mejorar el tiempo del algoritmo de Yannakakis?