

Integración de bases de conocimiento

Razonamiento ontológico bajo incertidumbre

Profesor invitado: Gerardo I. Simari (UNS Bahía Blanca)

Departamento de Computación, Universidad de Buenos Aires

Segundo cuatrimestre de 2020

Agenda

- Breve introducción a la incertidumbre
- Parte 1: Introducción a modelos probabilísticos gráficos
 - Redes Bayesianas
 - Redes de Markov
 - Lógica de Markov
 - Cadenas de Markov
- Parte 2: Ontologías probabilísticas
 - Repaso de Datalog+/-
 - Datalog+/- probabilístico:
 - Modelo
 - Evaluación empírica

Introducción

- La *incertidumbre* aparece en todas partes en la Web:
 - Incertidumbre natural / *inherente* al dominio (por ejemplo, en pronósticos del tiempo)
 - Incertidumbre proveniente del *procesamiento automático* de datos (como ya han visto en el curso)
 - Incertidumbre proveniente de la presencia de *inconsistencia* e *incompletitud*
- Actualmente, los buscadores y otras tecnologías de la Web no manejan la incertidumbre de manera *principada*.

Introducción

- Meta: acortar esta brecha mediante el desarrollo de herramientas que luego pueden ser aplicadas en la Web; por ejemplo, en *búsqueda semántica*.
- Una forma de lograr esto es mediante la integración de lenguajes de *ontología* con tecnología de *bases de datos* y *modelos probabilísticos*.
- En esta clase veremos:
 - Algunos *modelos* de datos probabilísticos que son útiles para el modelado de contenidos de la Web.
 - *Algoritmos* de respuesta a consultas.
 - *Fragmentos* escalables (pero expresivos) de los modelos.

Ejemplo

Consideremos el problema de la *extracción de entidades* en el siguiente texto, tomado de una página Web de noticias:

Fifty Shades novels drop in sales EL James has vacated the top of the UK book charts after 22 weeks, according to trade magazine The Bookseller.

According to the Bookseller, £29.3m was spent at UK booksellers between 15 and 22 September - a rise of £700,000 on the previous week.

	number
	book
	dl
	author
	country
	magazine
	money
	shop
	date

Modelos probabilísticos: *Introducción*

- Los modelos probabilísticos gráficos (*PGMs*) son estructuras basada en grafos que se utilizan para representar conocimiento acerca de un *dominio incierto*.
- Representación:
 - Nodos: *variables* aleatorias
 - Arcos: *dependencias* probabilísticas entre variables; la falta de arco entre variables señala independencia condicional.

Modelos probabilísticos: *Introducción*

Veremos cuatro tipos diferentes de PGMs:

- Redes Bayesianas (*BNs*)
- Redes de Markov / *Markov Random Fields* (*MRFs*)
- Redes Lógicas de Markov (*MLNs*)
- Cadenas de Markov (*MCs*)

Modelos probabilísticos:

Redes Bayesianas

Redes Bayesianas (BNs)

Una Red Bayesiana (BN) es un grafo dirigido acíclico donde:

- cada nodo representa una *variable aleatoria discreta*;
- si existe un *arco* entre el nodo X y el nodo Y , decimos que X es un *padre* de Y , y representa la *dependencia directa* entre X e Y ;
- a cada nodo se le asigna una *distribución probabilística condicional* $P(X_i | \text{Padres}(X_i))$ que cuantifica el efecto de los nodos “padre” sobre la variable X_i ;
- cada variable es *independiente* de sus no descendientes en el grafo, dado el estado de sus padres;
- la ausencia de arco entre dos nodos representa la independencia condicional entre las variables correspondientes.

Ejemplo

Supongamos que tenemos una *alarma anti-robo* instalada en casa. Es bastante confiable en la detección de robos, pero a veces responde también a *terremotos* menores.

Tenemos dos vecinos, John y Mary, que se han comprometido en llamarnos al trabajo si escuchan la alarma. John siempre llama cuando escucha la alarma, pero a veces *confunde* el sonido del teléfono con el de la alarma. Mary, por otro lado, disfruta de escuchar música a alto volumen y a veces *no escucha* la alarma.

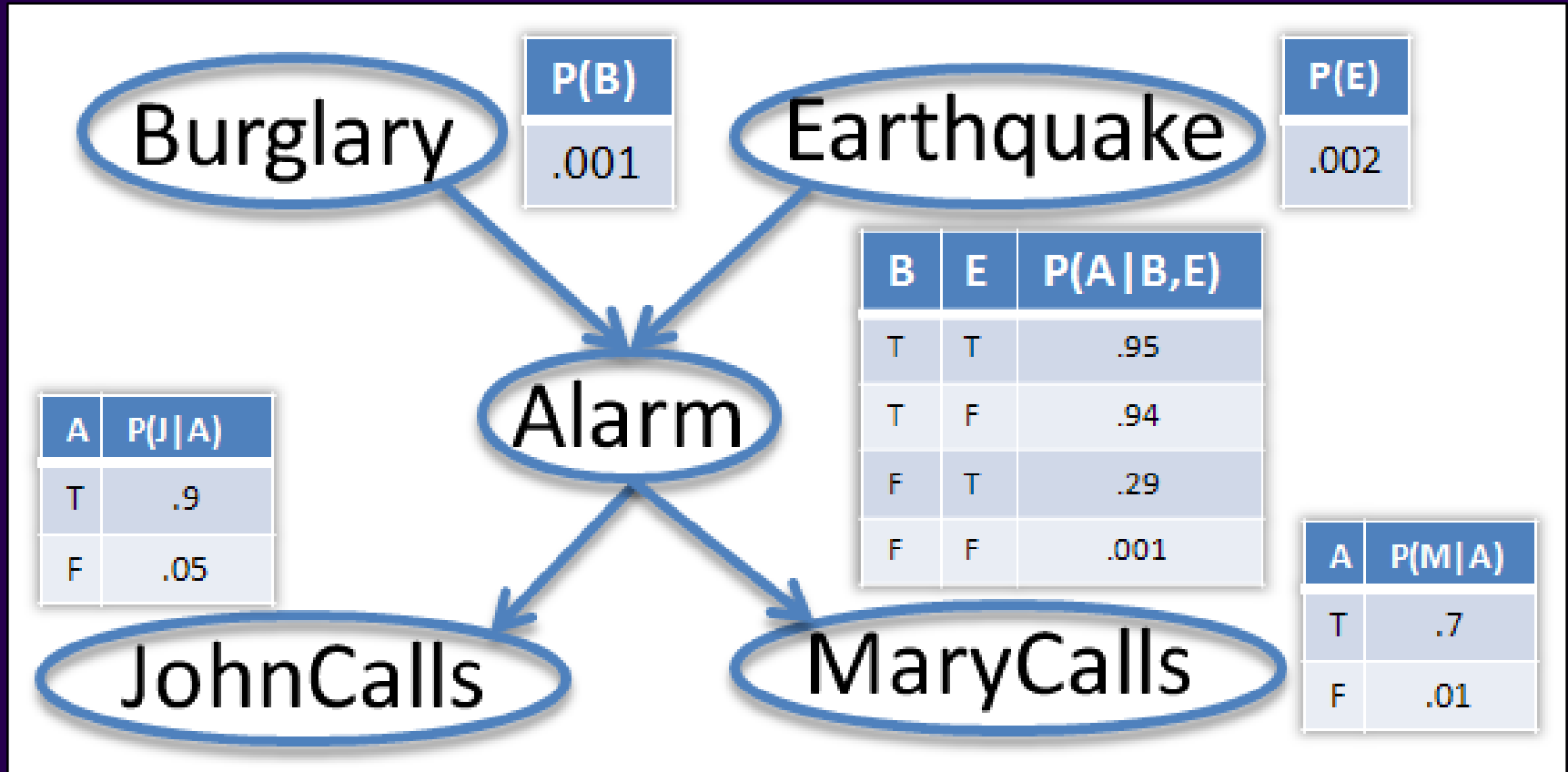
Dada la evidencia de quién ha llamado o no llamado, nos gustaría estimar la *probabilidad de que haya habido un robo*:

Estamos en el trabajo, llama John para avisar que escucha la alarma, pero Mary no llama; ¿hay robo?

Variables: *Burglary, Earthquake, Alarm, JohnCalls, MaryCalls*

Ejemplo

Una BN para este ejemplo podría ser la siguiente:



Redes Bayesianas (BNs)

- Una BN *describe completamente* una distribución.
- Es decir, se sabe la probabilidad de *cualquier conjunción* de asignaciones de valores a cada variable:

$$P(X_1 = x_1 \wedge X_2 = x_2 \wedge \dots \wedge X_n = x_n) = P(x_1, x_2, \dots, x_n)$$

- Cada entrada en la distribución completa puede ser calculada a partir de la información presente en la red:

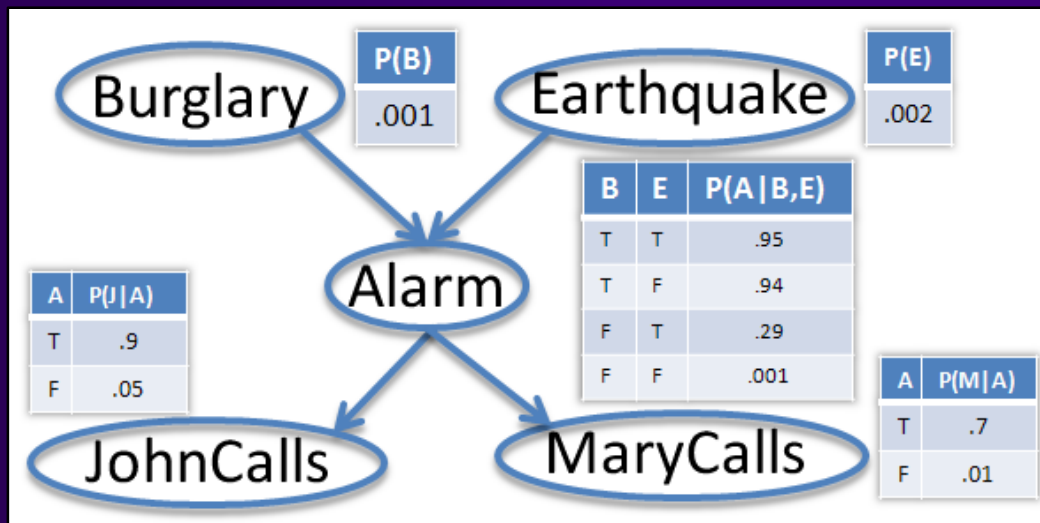
$$P(x_1, x_2, \dots, x_n) = \prod_{i=1}^n P(x_i \mid \text{padres}(X_i))$$

donde $\text{padres}(X_i)$ denota los valores de las variables en $\text{Padres}(X_i)$.

Ejemplo

Podemos calcular la probabilidad de que la alarma *suene*, pero que no haya habido *ni robo ni terremoto*, y que tanto John como Mary *llamen*; usando variables abreviadas, tenemos:

$$\begin{aligned} &P(a \wedge \neg b \wedge \neg e \wedge j \wedge m) \\ &= P(j \mid a)P(m \mid a)P(a \mid \neg b \wedge \neg e)P(\neg b)P(\neg e) \\ &= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 = 0.000628 \end{aligned}$$



BNs: *Problemas/Consultas*

Los problemas más comunes a resolver dada una BN son:

- PE (Probabilidad de “evidencia” / *Inferencia*): computar la probabilidad de que un subconjunto de variables tengan valores dados (*#P-completo*).
- MAP (Probabilidad *marginal* posterior): dada evidencia e y variable X_i , computar $Pr(X_i = x_i \mid e)$ (*PP-completo en su versión de decisión*).
- MPE (explicación más probable): dada evidencia, encontrar la asignación para el resto de las variables que tenga la mayor probabilidad (*NP-completo en su versión de decisión*).

Si bien todos estos problemas son *intratables* en su caso general, existen casos especiales con algoritmos polinomiales (exactos o de aproximación).

Modelos probabilísticos:

Redes de Markov

Redes de Markov (MRFs)

Una Red de Markov (o *Markov Random Field*, MRF) es un grafo *no dirigido* donde:

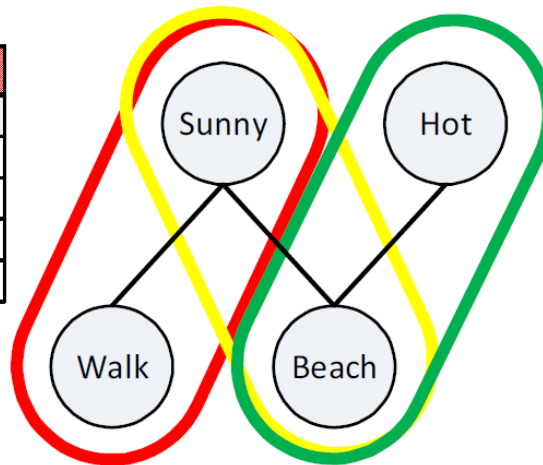
- cada *nodo* representa una variable aleatoria discreta;
- los *arcos* corresponden a una noción de *interacción* probabilística directa; ésta se parametriza con *funciones potenciales* (hay una función potencial por cada *clique* maximal);
- *potenciales*: funciones reales no negativas de los valores de las variables en cada clique (el *estado* del clique);
- un nodo es *condicionalmente independiente* del resto de los nodos en el grafo dados los valores de sus vecinos inmediatos (la *Markov blanket* del nodo).

Ejemplo

Variables:

- Sunny (el día está soleado)
- Hot (el día está caluroso)
- Beach (vamos a la playa)
- Walk (vamos a caminar)

Clique 1		
S	W	$\emptyset(H,B)$
t	f	2
t	t	1.7
f	t	0.3
f	f	3.1



Clique 2		
H	B	$\emptyset(H,B)$
t	f	1.6
t	t	3
f	t	0.4
f	f	2.5

Clique 3		
B	S	$\emptyset(H,B)$
t	f	2.8
t	t	1.7
f	t	0.2
f	f	2.8

Redes de Markov (MRFs)

La *distribución conjunta* de las variables $X = \{X_1, X_2, \dots, X_n\}$ se puede definir:

$$P(X = x) = \frac{1}{Z} \prod_i \phi_i(x_{\{i\}})$$

donde ϕ_i es la función potencial y $x_{\{i\}}$ es el estado del i -ésimo clique maximal.

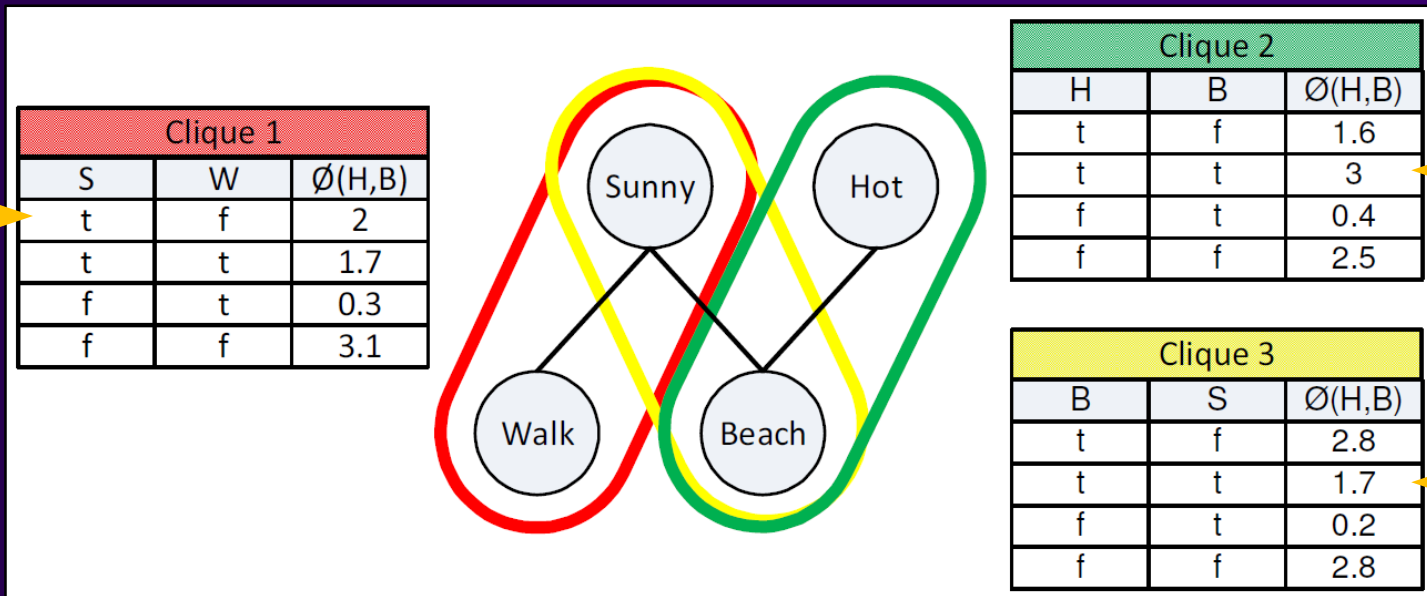
Z es una *constante normalizadora* para hacer que la suma de las probabilidades sea 1:

$$Z = \sum_{x \in X} \prod_i \phi_i(x_{\{i\}})$$

Ejemplo

Podemos calcular la probabilidad de que esté *soleado* y *caluroso*, y que vayamos a la *playa* pero *no a caminar*:

$$P(s \wedge h \wedge b \wedge \neg w) = \frac{1}{Z} (2 \times 3 \times 1.7) = \frac{10.2}{Z} = \frac{10.2}{82.82} \approx 0.1231$$



Redes de Markov (MRFs)

- Problema: expresar un valor para cada estado de cada clique es *exponencial* en el tamaño de los cliques.
- Podemos obtener una representación más *compacta* mediante funciones llamadas *features*.
- Por ejemplo, el modelo *log-linear* define:

$$P(X = x) = \frac{1}{Z} e^{\sum_i w_i f_i(x)}$$

donde los i varían sobre el conjunto de cliques:

$$Z = \sum_{x \in X} e^{\sum_i w_i f_i(x)}$$

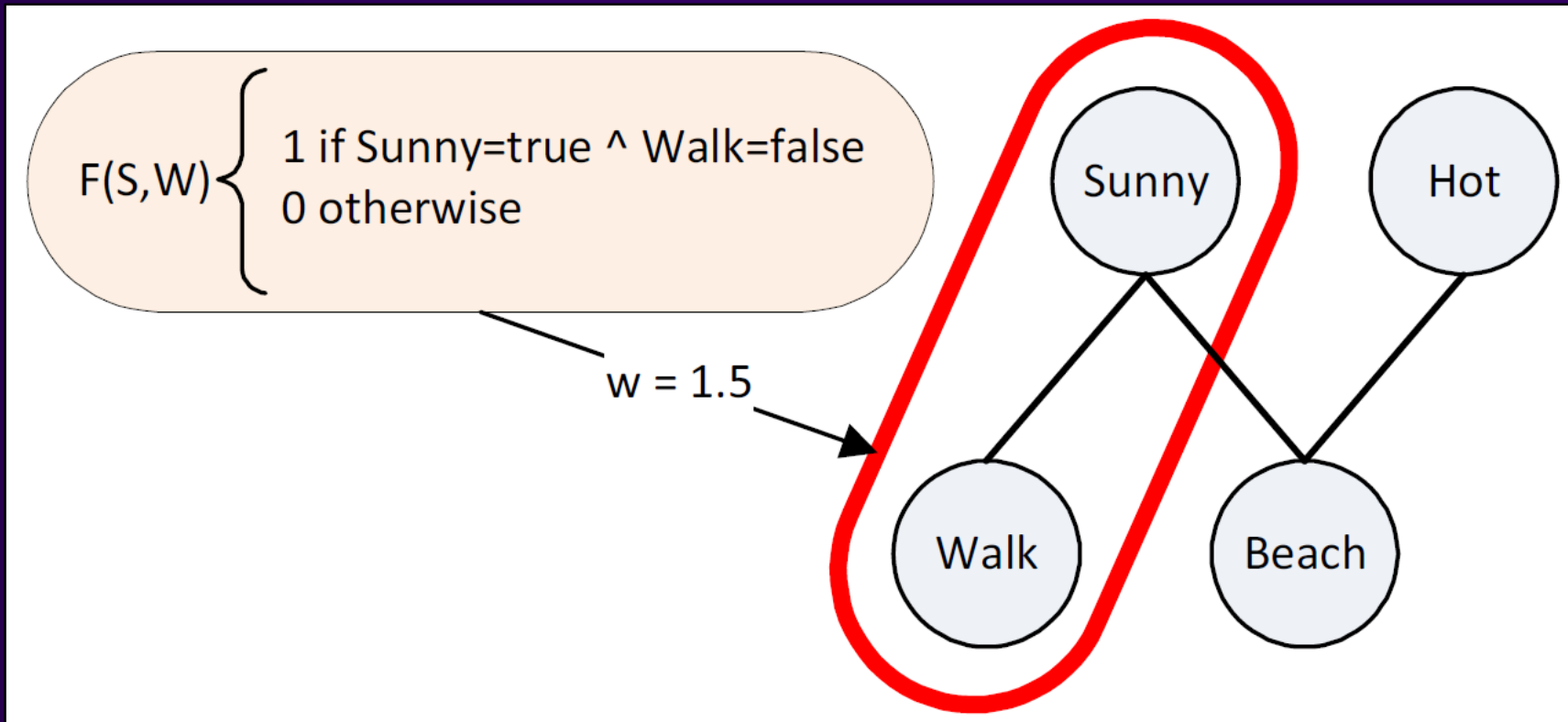
Redes de Markov (MRFs)

- Ahora, los *features* $f_i(x)$ (también funciones reales del estado) reemplazan a las funciones *potenciales*.
- Cada $f_i(x)$ tiene asociado un *peso* w_i
- Aquí consideramos features *binarios*: $f_i(x) \in \{0,1\}$.
- La traducción más directa de la forma anterior a ésta es:

un feature correspondiente a cada estado posible $x_{\{i\}}$
de cada clique, con peso $\ln \phi_i(x_{\{i\}})$.

Ejemplo

Volviendo a nuestro ejemplo, podemos definir un feature simple para el clique $\{Sunny, Walk\}$ de la siguiente manera:



Modelos probabilísticos:
Redes Lógicas de Markov
(o Lógica de Markov)

Redes Lógicas de Markov (MLNs)

Una MLN es un conjunto finito de *pares* (F_i, w_i) , donde:

- F_i es una *fórmula* en lógica de primer orden (FOL)
- w_i es un número real (el *peso* de la fórmula)

Junto con un conjunto finito de *constantes* $C = \{c_1, c_2, \dots, c_n\}$, define un *MRF* $M_{L,C}$ de la siguiente manera:

- $M_{L,C}$ contiene un nodo binario para cada posible *instancia básica* de un *átomo* en L . El valor del nodo es 1 si el átomo es verdadero, y 0 si es falso.
- $M_{L,C}$ contiene un *feature* por cada *instancia básica* de *fórmulas* F_i en L . El valor del feature es 1 si es verdadera y 0 si no, y el peso es el valor w_i asociado con F_i en L .

Redes Lógicas de Markov (MLNs)

Observaciones:

- Las MLNs pueden ser vistas como “plantillas” que *generan* redes de Markov al *instanciarse* con constantes.
- Los átomos básicos generan *nodos* en la red.
- Hay un *arco* entre dos nodos si y sólo si los átomos básicos correspondientes *aparecen juntos* en al menos una instancia básica de una fórmula en L .
- Las *fórmulas* generan *cliques* en la red.

Ejemplo

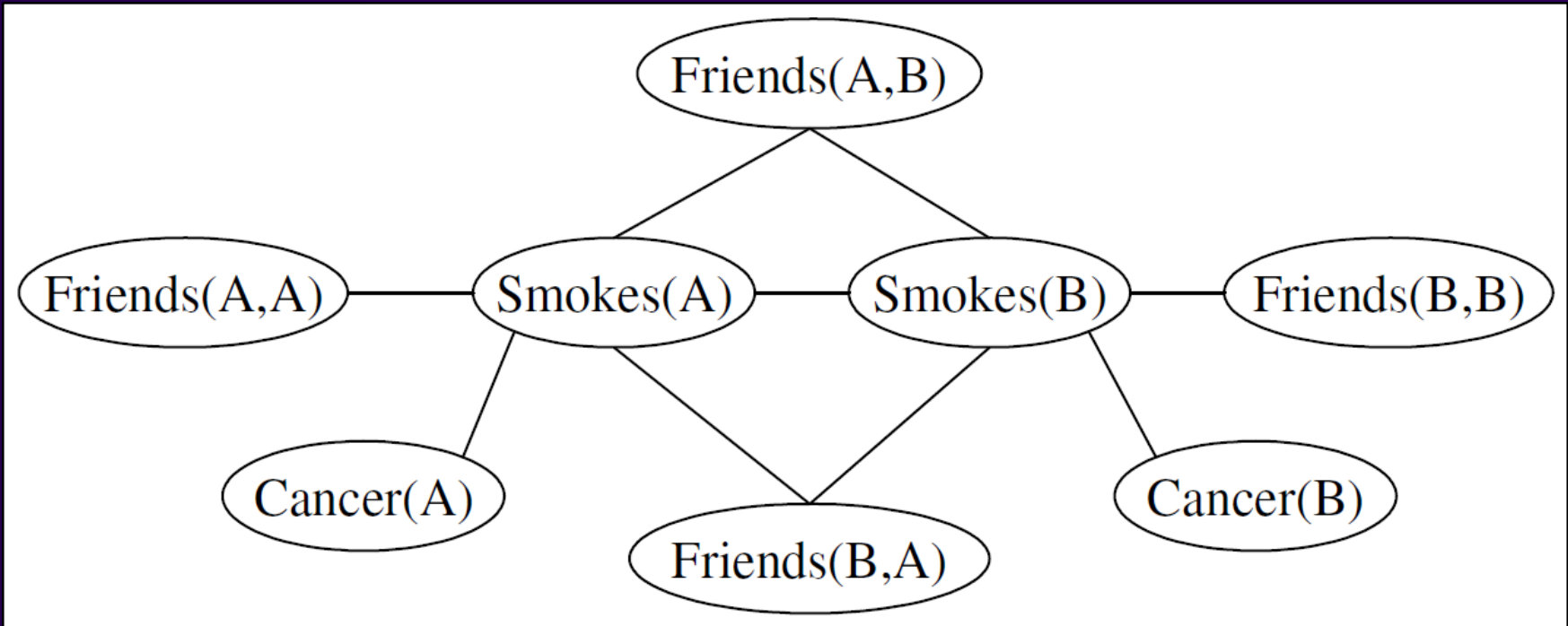
- Consideremos la MLN definida por los pares:
 - $(\forall x Sm(x) \Rightarrow Ca(x), 1.5) \rightsquigarrow$ Fumar causa cáncer
 - $(\forall x \forall y Fr(x, y) \Rightarrow (Sm(x) \Leftrightarrow Sm(y)), 1.1) \rightsquigarrow$ si dos personas son amigas, o bien las dos fuman o ninguna de las dos fuma.

Tenemos las constantes: $\{Anna, Bob\}$.

- $M_{L,C}$ ahora puede ser usada para inferir la **probabilidad** de que *Anna* y *Bob* son amigos dados sus hábitos de fumar, la probabilidad de que *Bob* tenga cáncer dada su amistad con *Anna*, y si ella tiene cáncer, etc.

Ejemplo

El siguiente grafo corresponde a la *MRF inducida*:



Fórmulas: $\forall x Sm(x) \Rightarrow Ca(x), \quad \forall x \forall y Fr(x, y) \Rightarrow (Sm(x) \Leftrightarrow Sm(y))$

Redes Lógicas de Markov (MLNs)

La *distribución* de probabilidad representada por la MLN es la siguiente:

$$P(X = x) = \frac{1}{Z} e^{\sum_i w_i n_i(x)}$$

donde $n_i(x)$ es la *cantidad* de instancias básicas de F_i que son satisfechas por x , y Z es la constante de normalización.

Un ejemplo simple completo

- Definamos una MLN con los siguientes *pares*:

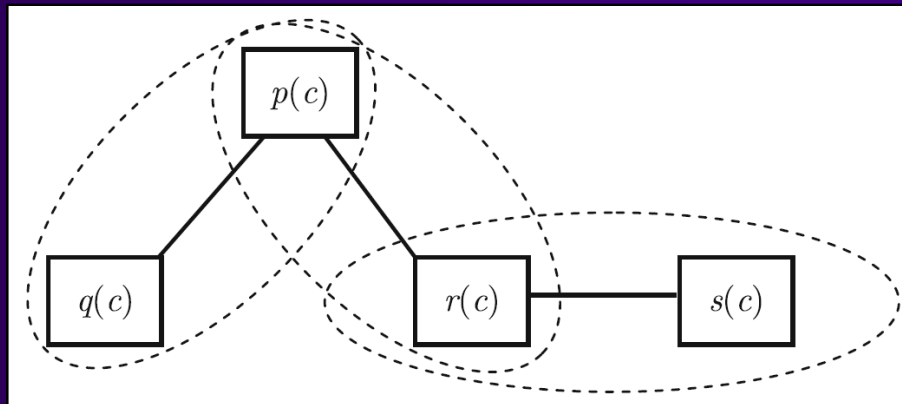
$$\psi_1: (p(X) \Rightarrow q(X), 1.2)$$

$$\psi_2: (p(X) \Rightarrow r(X), 2)$$

$$\psi_3: (s(X) \Rightarrow r(X), 3)$$

y el conjunto de *constantes* $\{c\}$.

- Conjunto de *átomos básicos* $\{p(c), q(c), r(c), s(c)\}$, y el grafo:



Un ejemplo simple completo

Tenemos entonces $2^4 = 16$ posibles *asignaciones* de valores para las variables de la MRF. Las probabilidades son:

λ_i	$p(c)$	$q(c)$	$r(c)$	$s(c)$	Satisfies	Potential	Probability
1	false	false	false	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
2	false	false	false	true	ψ_1, ψ_2	$1.2 + 2 = 3.2$	$e^{3.2}/Z \approx 0.006$
3	false	false	true	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
4	false	false	true	true	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
5	false	true	false	false	ψ_1, ψ_2	$1.2 + 2 = 3.2$	$e^{3.2}/Z \approx 0.006$
6	false	true	false	true	ψ_1, ψ_2	$1.2 + 2 = 3.2$	$e^{3.2}/Z \approx 0.006$
7	false	true	true	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
8	false	true	true	true	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
9	true	false	false	false		0	$e^0/Z \approx 0$
10	true	false	false	true		0	$e^0/Z \approx 0$
11	true	false	true	false	ψ_2, ψ_3	$2 + 3 = 5$	$e^5/Z \approx 0.038$
12	true	false	true	true	ψ_2, ψ_3	$2 + 3 = 5$	$e^5/Z \approx 0.038$
13	true	true	false	false	ψ_1, ψ_3	$1.2 + 3 = 4.2$	$e^{4.2}/Z \approx 0.017$
14	true	true	false	true	ψ_1	1.2	$e^{1.2}/Z \approx 0$
15	true	true	true	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
16	true	true	true	true	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$

Un ejemplo simple completo

- El factor normalizador Z se calcula como sigue:

$$Z = 7e^{6.2} + 3e^{3.2} + 2e^0 + 2e^5 + e^{4.2} + e^{1.2} \approx 3,891.673$$

- Si queremos calcular la probabilidad de la *fórmula* $p(c) \wedge q(c)$, debemos *sumar* las probabilidades de todos los *mundos* que la satisfacen, es decir 13, 14, 15 y 16:

$$\frac{e^{4.2} + e^{1.2} + e^{6.2} + e^{6.2}}{Z} \approx \frac{1,055.5}{3,891.673} \approx 0.271$$

λ_i	$p(c)$	$q(c)$	$r(c)$	$s(c)$	Satisfies	Potential	Probability
1	false	false	false	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
2	false	false	false	true	ψ_1, ψ_2	$1.2 + 2 = 3.2$	$e^{3.2}/Z \approx 0.006$
3	false	false	true	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
4	false	false	true	true	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
5	false	true	false	false	ψ_1, ψ_2	$1.2 + 2 = 3.2$	$e^{3.2}/Z \approx 0.006$
6	false	true	false	true	ψ_1, ψ_2	$1.2 + 2 = 3.2$	$e^{3.2}/Z \approx 0.006$
7	false	true	true	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
8	false	true	true	true	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
9	true	false	false	false		0	$e^0/Z \approx 0$
10	true	false	false	true		0	$e^0/Z \approx 0$
11	true	false	true	false	ψ_2, ψ_3	$2 + 3 = 5$	$e^5/Z \approx 0.038$
12	true	false	true	true	ψ_2, ψ_3	$2 + 3 = 5$	$e^5/Z \approx 0.038$
13	true	true	false	false	ψ_1, ψ_3	$1.2 + 3 = 4.2$	$e^{4.2}/Z \approx 0.017$
14	true	true	false	true	ψ_1	1.2	$e^{1.2}/Z \approx 0$
15	true	true	true	false	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$
16	true	true	true	true	ψ_1, ψ_2, ψ_3	$1.2 + 2 + 3 = 6.2$	$e^{6.2}/Z \approx 0.127$

Modelos probabilísticos:

Cadenas de Markov

Cadenas de Markov (MCs)

Una Cadena de Markov (MC) es un proceso estocástico

$\{X_n\}_{n \in \mathbb{N} \cup \{0\}}$:

- Conjunto de variables aleatorias que representan la *evolución* de un sistema de valores aleatorios en el tiempo.
- Verifica la propiedad de Markov:

dado $n \in \mathbb{N} \cup \{0\}$ y los estados $x_0, x_1, \dots, x_n, x_{n+1}$, tenemos:

$$P(X_{n+1} = x_{n+1} \mid X_n = x_n, \dots, X_0 = x_0) = P(X_{n+1} = x_{n+1} \mid X_n = x_n)$$

Cadenas de Markov (MCs)

- La propiedad de Markov afirma que la distribución de probabilidad condicional de estados futuros depende *sólo del estado actual*.
- Las MC se pueden representar como *secuencia de grafos* donde los arcos del grafo n se etiquetan con la probabilidad de ir de un estado en el momento n a otros estados en el momento $n + 1$:

$$P(X_{n+1} = x \mid X_n = x_n).$$

Cadenas de Markov (MCs)

- Se puede representar la misma información con la *matriz de transición* del tiempo n al $n + 1$:

$$\begin{pmatrix} p_{11} & p_{12} & \dots & p_{1j} & \dots \\ p_{21} & p_{22} & \dots & p_{2j} & \dots \\ \vdots & \vdots & \ddots & \vdots & \ddots \\ p_{i1} & p_{i2} & \dots & p_{ij} & \dots \\ \vdots & \vdots & \ddots & \vdots & \ddots \end{pmatrix}$$

donde $p_{ij} = P(X_{n+1} = x_j \mid X_n = x_i)$.

- Dado que las probabilidades de transicionar del estado i al resto de los estados debe sumar 1, tenemos $\sum_j p_{ij} = 1$.

Ejemplo

Hay *modelos sociológicos* que clasifican a las personas según sus ingresos en: clase baja, clase media y clase alta.

Algunos estudios sugieren que el *determinante más fuerte* de la clase de ingresos de una persona es la clase de sus padres.

Tenemos entonces:

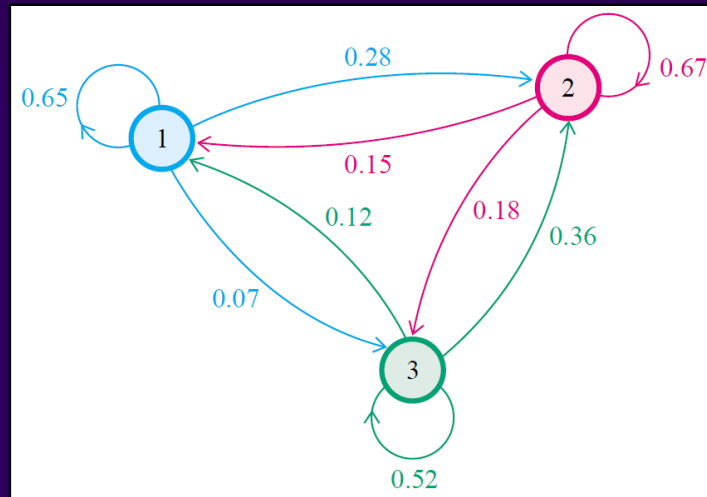
- una persona en la clase baja está en el *estado 1*
- una persona en la clase media está en el *estado 2*
- una persona en la clase alta está en el *estado 3*

Próxima generación

		<i>Próxima generación</i>		
<i>Generación actual</i>	Estado	1	2	3
	1	0.65	0.28	0.07
	2	0.18	0.67	0.18
	3	0.12	0.36	0.52

Ejemplo

- La información se puede expresar con un *diagrama de transición*:



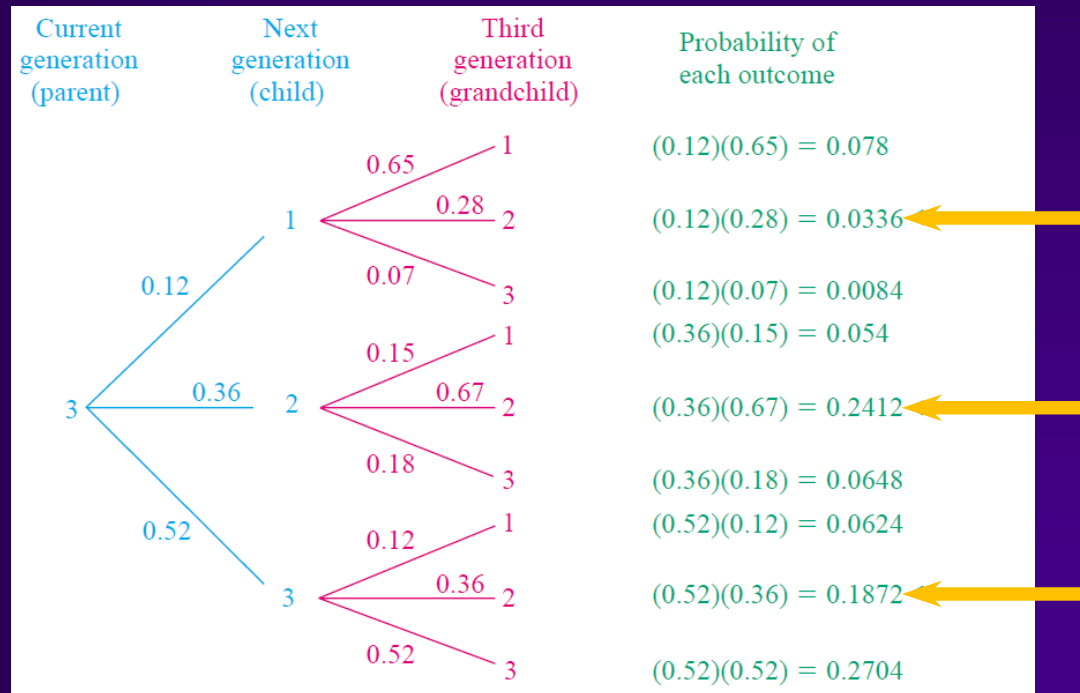
- O con una *matriz de transición*:

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0.65 & 0.28 & 0.07 \\ 2 & 0.15 & 0.67 & 0.18 \\ 3 & 0.12 & 0.36 & 0.52 \end{array} = P$$

Ejemplo

Ahora queremos saber las probabilidades de **cambios** en clase de ingreso en dos generaciones.

Por ejemplo, si un padre está en el estado 3, ¿cuál es la probabilidad de que un nieto dado esté en el estado 2?



Probabilidad: $0.0336 + 0.2412 + 0.1872 = 0.462$

Ejemplo

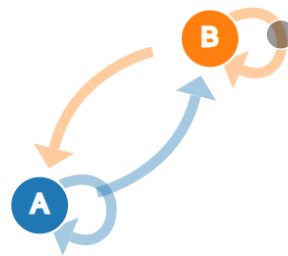
También podríamos tomar el elemento $(P^2)_{32}$:

$$\begin{pmatrix} 0.65 & 0.28 & 0.07 \\ 0.15 & 0.67 & 0.18 \\ 0.12 & 0.36 & 0.52 \end{pmatrix} \cdot \begin{pmatrix} 0.65 & 0.28 & 0.07 \\ 0.15 & 0.67 & 0.18 \\ 0.12 & 0.36 & 0.52 \end{pmatrix} =$$
$$= \begin{pmatrix} * & * & * \\ * & * & * \\ * & 0.12 \times 0.28 + 0.36 \times 0.67 + 0.52 \times 0.36 = 0.462 & * \end{pmatrix}$$

Visualización

Markov Chains explained visually:

<https://setosa.io/ev/markov-chains/>







With two states (A and B) in our state space, there are 4 possible transitions (not 2, because a state can transition back into itself). If we're at 'A' we could transition to 'B' or stay at 'A'. If we're at 'B' we could transition to 'A' or stay at 'B'. In this two state diagram, the probability of transitioning from any state to any other state is 0.5.

Of course, real modelers don't always draw out Markov chain diagrams. Instead they use a "transition matrix" to tally the transition probabilities. Every state in the state space is included once as a row and again as a column, and each cell in the matrix tells you the probability of transitioning from its row's state to its column's state. So, in the matrix, the cells do the same job that the arrows do in the diagram.

 speed



	A	B
A	P(A A): 0.50 	P(B A): 0.50 
B	P(A B): 0.50 	P(B B): 0.50 

MCs: *Distribución Estacionaria*

- Decimos que una MC es *regular* si alguna potencia de su matriz de transición contiene todas entradas *positivas*.
- Si una MC con matriz P es regular, entonces existe un *vector* único V , tal que para cualquier vector de probabilidades v y valores grande de n tenemos:

$$v \cdot P^n \approx V.$$

- El vector V se llama la *distribución estacionaria* de la MC.
- Para hallar V podemos resolver la ecuación $V \cdot P = V$, usando el hecho de las entradas deben sumar 1.
- Las potencias P^n se van *acercando* a la matriz cuyas filas corresponden a la distribución estacionaria V .

Ejemplo

Queremos hallar $V = (v_1, v_2, v_3)$ tal que $V.P = V$, es decir,

$$(v_1 \quad v_2 \quad v_3) \cdot \begin{pmatrix} 0.65 & 0.28 & 0.07 \\ 0.15 & 0.67 & 0.18 \\ 0.12 & 0.36 & 0.52 \end{pmatrix} = (v_1 \quad v_2 \quad v_3)$$

Resolviendo el sistema de ecuaciones:

$$\begin{cases} -0.35v_1 + 0.15v_2 + 0.12v_3 = 0 \\ 0.28v_1 - 0.33v_2 + 0.36v_3 = 0 \\ 0.07v_1 + 0.18v_2 - 0.48v_3 = 0 \\ v_1 + v_2 + v_3 = 1 \end{cases}$$

obtenemos $V = \left(\frac{104}{363}, \frac{532}{1089}, \frac{245}{1089}\right) \approx (0.2865, 0.4885, 0.2250)$

Ejemplo

Si computamos varias potencias de la matriz, observamos que se *aproximan* a la matriz cuyas filas corresponden a la distribución *estacionaria*:

$$P^4 = \begin{pmatrix} 0.34 & 0.47 & 0.20 \\ 0.27 & 0.50 & 0.23 \\ 0.26 & 0.29 & 0.25 \end{pmatrix}$$

$$P^{10} = \begin{pmatrix} 0.29 & 0.49 & 0.22 \\ 0.29 & 0.49 & 0.23 \\ 0.29 & 0.49 & 0.23 \end{pmatrix}$$

$$P^{16} = \begin{pmatrix} 0.29 & 0.49 & 0.22 \\ 0.29 & 0.49 & 0.22 \\ 0.29 & 0.49 & 0.22 \end{pmatrix}$$

¿Para qué podemos usar las MCs?

- Las Cadenas de Markov son la base de muchos algoritmos basados en tomar muestras a partir de una distribución (*sampling*).
- Dan lugar a los métodos *Markov Chain Monte Carlo*, nombrados en honor al Casino de Monte-Carlo en Mónaco:



Markov chain Monte Carlo (MCMC)

- Queremos *aproximar* una distribución $P(X|x)$
- La idea del método es simular una MC $\{X_i\}_{i \in \mathbb{N} \cup \{0\}}$ con distribución *estacionaria* $P(X|x)$:
 - 1) Comenzamos con un estado *aleatorio* X_0 .
 - 2) Se genera el *próximo* estado iterativamente tomando *muestras* del *valor* de una de las variables X_i *condicionada* por los valores actuales de las variables en la *Markov blanket* de X_i .
 - 3) La distribución estacionaria se aproxima ejecutando el paso 2 suficientes veces (esta cantidad de veces se denomina *mixing time*).
- Nótese que cada muestra del valor de X_i depende sólo de su *predecesor* X_{i-1} .
- La probabilidad de una *consulta* también se puede computar mediante este tipo de *random walk*.

Ejemplo

Consulta: $P(\text{Beach} \mid \text{Sunny} = \text{true}, \text{Hot} = \text{true})$

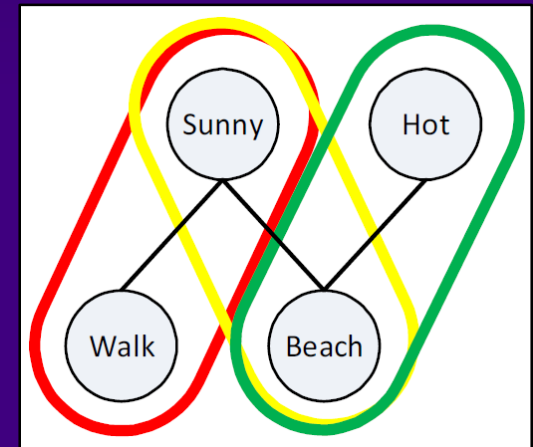
- Como primer estado podemos tomar: $\{s, h, b, \neg w\}$
- Para el próximo tomamos una muestra de *Walk* dada su *Markov blanket*, es decir, $P(\text{Walk} \mid \text{Sunny} = \text{true})$
- Si la muestra es $\text{Walk} = \text{true}$, el próximo estado es: $\{s, h, b, w\}$
- Si luego de 10 pasos tenemos 9 donde $\text{Beach} = \text{true}$ y 2 donde $\text{Beach} = \text{false}$, tenemos una **aproximación** de

$$P(B = \text{true} \mid S = \text{true}, H = \text{true}) = 0.9$$

El valor **exacto** se calcula:

$$\frac{8.67+10.2}{8.67+10.2+0.544+0.64} = 0.94$$

(usando los valores de la transparencia 19)



Algunas conclusiones

- Vimos cuatro ejemplos de modelos que se pueden usar para representar la *incertidumbre* del dominio de aplicación:
 - Las BNs son útiles cuando las dependencias son *acíclicas* y tenemos la *información* de las dependencias condicionales (*estructura* del grafo + *probabilidades*).
 - Las MRFs son más *flexibles* dado que permiten ciclos y probabilidades derivadas de pesos; la desventaja es que la *relación* entre pesos y probabilidades no siempre es clara.

Algunas conclusiones

- Vimos cuatro ejemplos de modelos que se pueden usar para representar la *incertidumbre* del dominio de aplicación:
 - Las MLNs son esencialmente *moldes de primer orden* para MRFs.
 - Las MCs son útiles en métodos *MCMC* para aproximar las distribuciones especificadas por otros modelos (como BNs o MRFs), y también otros sistemas dinámicos.
- La *complejidad* computacional de todos estos modelos es alta, pero existen métodos de aproximación y/o restricción a fragmentos para atacarla.

Parte 2:

Ontologías Datalog+/- probabilísticas

Datalog+/- Probabilístico

- **Objetivo:** combinar Datalog+/- “clásico” con modelos probabilísticos (en esta clase, MLNs).
- La idea básica consiste en **anotar** fórmulas con conjuntos de eventos probabilísticos:
 - Las anotaciones significan que la fórmula en cuestión sólo aplica cuando el evento asociado sucede.
 - La distribución de probabilidad asociada a los eventos se describe mediante una MLN.
- En esta clase nos vamos a enfocar en las llamadas “**ranking queries**”, pero también existen resultados para consultas **conjuntivas** y **threshold** (umbral).

Repaso de Datalog+/-

- Repaso de notación::
 - Universo infinito de *constantes* Δ
 - Conjunto infinito de *nulls* etiquetados Δ_N
 - Conjunto infinito de *variables* \mathcal{V}
 - Esquema relacional \mathcal{R} , el cual consiste de un conjunto finito de *nombres de relación* (o símbolos predicativos).
- Nota: Diferentes constantes representan diferentes valores, pero diferentes nulls pueden representar el mismo valor.
- Usamos \mathbf{X} para denotar la secuencia X_1, \dots, X_n , con $n \geq 0$.
- Un *término* t es una constante, null o variable.
- Una fórmula atómica (o *átomo*) tiene la forma $P(t_1, \dots, t_n)$, donde P es un predicado n -ario y los t_i 's son términos.

Repaso de Datalog+/-

- Una (instancia) de base de datos D sobre \mathcal{R} es un conjunto de átomos con predicados de \mathcal{R} y argumentos de Δ .

$$D = \{ emp(bob), manager(bob), directs(bob, hr), emp(ann), supervises(bob, ann), manager(ann), works_in(ann, hr), works_in(bob, hr), works_in(bob, finance) \}$$

- Una consulta conjuntiva (**CQ**) sobre \mathcal{R} tiene la forma:
 $Q(\mathbf{X}) = \exists \mathbf{Y} \ \Phi(\mathbf{X}, \mathbf{Y})$, donde Φ es una conjunción de átomos.

$$Q(X) = manager(X) \wedge directs(X, hr) \quad X = \dots$$

- Una CQ Booleana (**BCQ**) sobre \mathcal{R} tiene la forma:
 $Q() = \exists \mathbf{X}, \mathbf{Y} \ \Phi(\mathbf{X}, \mathbf{Y})$, donde Φ es una conj. de átomos.

$$Q() = \exists X \ manager(X) \wedge directs(X, hr) \quad Yes / No$$

Repaso de Datalog+/-

- Las respuestas a las consultas se definen vía **homomorfismos**, que son mapeos $\mu: \Delta \cup \Delta_N \cup \mathcal{V} \rightarrow \Delta \cup \Delta_N \cup \mathcal{V}$ tal que.:
 - $c \in \Delta$ implica $\mu(c) = c$
 - $c \in \Delta_N$ implica $\mu(c) \in \Delta \cup \Delta_N$
 - μ se extiende a átomos, conjuntos de átomos y conjunciones.
- La **respuesta** $Q(D)$ es el conjunto de tuplas t sobre Δ tal que $\exists \mu: \mathbf{X} \cup \mathbf{Y} \rightarrow \Delta \cup \Delta_N, \mu(\Phi(\mathbf{X}, \mathbf{Y})) \subseteq D$, y $\mu(\mathbf{X}) = t$.

Para $Q(X) = \text{manager}(X) \wedge \text{directs}(X, hr)$, el conjunto de respuestas sobre D es $Q(D) = \{bob\}$.

La respuesta a $Q() = \exists X \text{manager}(X) \wedge \text{directs}(X, hr)$ es *Yes*.

Repaso de Datalog+/-

- Las *Tuple-generating Dependencies* (TGDs) son restricciones de la forma $\forall \mathbf{X} \forall \mathbf{Y} \Phi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z} \Psi(\mathbf{X}, \mathbf{Z})$ donde Φ y Ψ son conjunciones atómicas sobre \mathcal{R} llamadas el cuerpo y cabeza de la TGD, respectivamente.
- Ejemplos de TGDs:

$$\text{manager}(M) \rightarrow \text{emp}(M)$$

$$\text{manager}(M) \rightarrow \exists P \text{ directs}(M, P)$$

$$\text{emp}(E) \wedge \text{directs}(E, P) \rightarrow$$

$$\exists E' \text{ emp}(E') \wedge \text{supervises}(E, E') \wedge \text{works_in}(E', P)$$

Repaso de Datalog+/-

- Dada una BD D y un conjunto Σ de TGDs, el conjunto de **modelos** $mods(D, \Sigma)$ es el conjunto de todos los B tal que:
 - $D \subseteq B$
 - toda $\sigma \in \Sigma$ está satisfecha en B .
- El conjunto de **respuestas** para una CQ Q a D y Σ , $ans(Q, D, \Sigma)$, es el conjunto de todas las tuplas a tales que $a \in Q(B)$ para todo $B \in mods(D, \Sigma)$.
- Las respuestas se pueden computar por medio del **chase**, un procedimiento para reparar una BD con respecto a un conjunto de dependencias.

Repaso del *Chase*

- Regla (informal) del chase para TGDs:
 - una TGD σ es **aplicable** en una BD si $body(\sigma)$ mapea a átomos en D ;
 - si no está ya en D , la aplicación de σ sobre D **agrega un átomo con nulls “frescos”** correspondientes a cada variables existencialmente cuantificada en $head(\sigma)$.
- El chase (posiblemente infinito) es un **modelo universal**: existe un homomorfismo de $chase(D, \Sigma)$ a cada $B \in mods(D, \Sigma)$.
- Por lo tanto, tenemos que $D \cup \Sigma \models Q$ iff $chase(D, \Sigma) \models Q$.
- Si Σ respeta ciertas restricciones, las CQs se pueden evaluar sobre un fragmento de **profundidad constante** $k \cdot |Q|$, el cual es PTIME en la complejidad data.

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \, emp(M) \wedge managerOf(M, E)$

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \, emp(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \text{ emp}(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

$emp(z_1)$

$managerOf(z_1, bob)$

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \text{ emp}(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

$emp(z_1)$

$managerOf(z_1, bob)$

$emp(z_2)$

$managerOf(z_2, z_1)$

...

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \text{ emp}(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

$emp(z_1)$

$managerOf(z_1, bob)$

$emp(z_2)$

$managerOf(z_2, z_1)$

...

Consulta:

$Q() \equiv \exists X \text{ managerOf}(X, bob)$

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \text{ emp}(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

$emp(z_1)$

$managerOf(z_1, bob)$

$emp(z_2)$

$managerOf(z_2, z_1)$

...

Consulta:

$Q() \equiv \exists X \text{ managerOf}(X, bob)$

Respuesta: *Yes*

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \text{ emp}(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

$emp(z_1)$

$managerOf(z_1, bob)$

$emp(z_2)$

$managerOf(z_2, z_1)$

...

Consulta:

$Q(X) \equiv managerOf(X, bob)$

Repaso del *Chase*

Consideremos la ontología:

$emp(bob)$

$emp(E) \rightarrow \exists M \text{ emp}(M) \wedge managerOf(M, E)$

Chase:

$emp(bob)$

$emp(z_1)$

$managerOf(z_1, bob)$

$emp(z_2)$

$managerOf(z_2, z_1)$

...

Consulta:

$Q(X) \equiv managerOf(X, bob)$

Respuesta: $\{\}$

Repaso: NCs y EGDs

- Las *Negative Constraints* (NCs) son fórmulas de la forma $\forall \mathbf{X} \Phi(\mathbf{X}) \rightarrow \perp$, donde $\Phi(\mathbf{X})$ es una conjunción de átomos.
- Son fáciles de verificar, dado que se puede hacer la CQ $\Phi(\mathbf{X})$ y verificar que el conjunto de respuestas sea vacío.
- Las *Equality Generating Dependencies* (EGDs) son de la forma $\forall \mathbf{X} \Phi(\mathbf{X}) \rightarrow X_i = X_j$, donde Φ es una conjunción de átomos y X_i, X_j son variables en \mathbf{X} .
- El chase se puede extender fácilmente para TGDs y EGDs.
- En general suponemos que las son separables, lo cual intuitivamente significa que las EGDs y TGDs son independientes entre sí.

Ontologías Datalog+/- Probabilísticas

- En esta parte vamos a usar las Redes Lógicas de Markov como *modelo probabilístico* de base para extender Datalog+/-.
- Si bien algunos aspectos algorítmicos (buscando tratabilidad) dependen de este modelo, la propuesta se puede formular utilizando *cualquier modelo (representación)* de una distribución de probabilidad sobre eventos.

Ejemplo

Modelemos el ejemplo del comienzo con una **MLN**:

$\phi_1: ann(S_1, I_1, num) \wedge ann(S_2, I_2, X) \wedge overlap(I_1, I_2) : 3$

$\phi_2: ann(S_1, I_1, shop) \wedge ann(S_2, I_2, mag) \wedge overlap(I_1, I_2) : 1$

$\phi_3: ann(S_1, I_1, dl) \wedge ann(S_2, I_2, pers) \wedge overlap(I_1, I_2) : 0.25$

The diagram illustrates two sentences with semantic role labels (S₁, I₁, S₂, I₂) and a legend. The legend defines the roles: number (light blue), book (blue), dl (green), author (red), country (olive green), magazine (yellow), money (pink), shop (cyan), and date (orange).

Sentence 1: Fifty Shades novels drop in sales EL James has vacated the top of the UK book charts after 22 weeks, according to trade magazine The Bookseller.

Sentence 2: According to the Bookseller, £29.3m was spent at UK booksellers between 15 and 22 September - a rise of £700,000 on the previous week.

Legend:

- number
- book
- dl
- author
- country
- magazine
- money
- shop
- date

Ejemplo

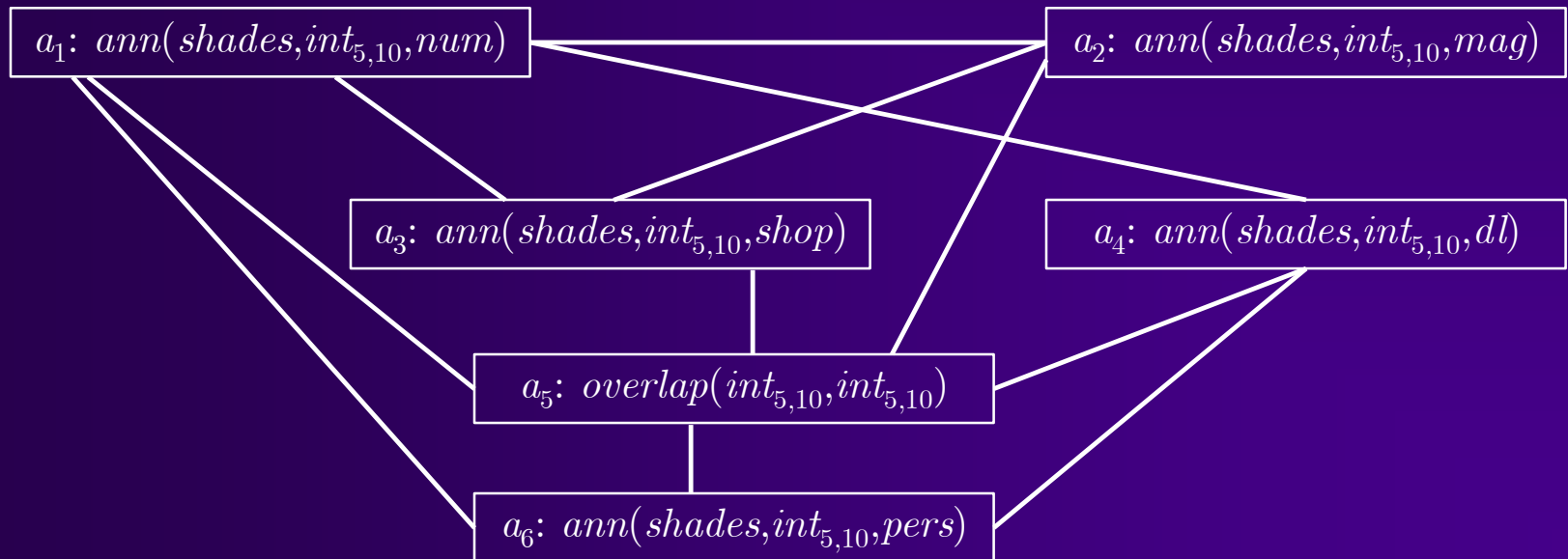
Modelemos el ejemplo del comienzo con una **MLN**:

$$\phi_1: \text{ann}(S_1, I_1, \text{num}) \wedge \text{ann}(S_2, I_2, X) \wedge \text{overlap}(I_1, I_2) : 3$$

$$\phi_2: \text{ann}(S_1, I_1, \text{shop}) \wedge \text{ann}(S_2, I_2, \text{mag}) \wedge \text{overlap}(I_1, I_2) : 1$$

$$\phi_3: \text{ann}(S_1, I_1, \text{dl}) \wedge \text{ann}(S_2, I_2, \text{pers}) \wedge \text{overlap}(I_1, I_2) : 0.25$$

Representación en grafo:



Ejemplo

Las probabilidades asociadas a cada mundo son:

λ_i	a_1	a_2	a_3	a_4	a_5	a_6	SAT	Probability
1	False	False	False	False	False	False	—	e^0 / Z
2	False	False	False	True	True	True	ϕ_3	$e^{0.25} / Z$
3	True	False	False	True	True	True	ϕ_1, ϕ_3	$e^{3+0.25} / Z$
4	True	False	True	True	True	True	ϕ_1, ϕ_3	$e^{3+0.25} / Z$
5	False	True	False	False	True	False	—	e^0 / Z
6	False	True	True	False	True	True	ϕ_2	e^1 / Z
7	False	True	True	True	True	True	ϕ_2, ϕ_3	$e^{1+0.25} / Z$
8	True	True	True	True	True	True	ϕ_1, ϕ_2, ϕ_3	$e^{3+1+0.25} / Z$

... (en total son $2^6 = 64$ combinaciones de valores posibles para las 6 variables aleatorias Booleanas).

Ontologías Datalog+/- Probabilísticas

- Una ontología Datalog+/- probabilística consiste de una ontología Datalog+/- clásica O junto con una MLN M .
Notación: $KB = (O, M)$
- Las fórmulas de O se **anotan** con un conjunto de pares $\langle X_i = x_i \rangle$, con $x_i \in \{true, false\}$ (o bien 0 y 1, resp.).
- Las variables que **no aparecen** en la anotación no están restringidas (y por lo tanto pueden tomar **cualquier valor**).
- **Mundo posible**: conjunto de pares $\langle X_i = x_i \rangle$ donde cada $X_i \in X$ tiene un par correspondiente.
- Intuición: dado un mundo posible, se **induce** un subconjunto de las fórmulas de O (en Datalog+/- clásico).

Ontologías Datalog+/- Probabilísticas

- Una ontología Datalog+/- probabilística consiste de una ontología Datalog+/- clásica O junto con una MLN M .
Notación: $KB = (O, M)$
- Las fórmulas de O se **anotan** con un conjunto de pares $\langle X_i = x_i \rangle$, con $x_i \in \{true, false\}$ (o bien 0 y 1, resp.).
- En ontologías con “acoplamiento alto” (**tightly coupled**) permitimos **variables** en las **anotaciones**, las cuales pueden aparecer del lado de la ontología:

Ejemplo: $number(X): \{ann(X, I, num) = true\}$

- Esto aumenta la expresividad, pero causa que la cantidad de mundos **dependa** del tamaño de la BD.

Volviendo al ejemplo

Adaptamos las fórmulas de los ejemplos anteriores para armar una ontología Datalog+/- probabilística:

$$book(X) \rightarrow editorialProd(X) : \{\}$$

$$magazine(X) \rightarrow editorialProd(X) : \{\}$$

$$author(X) \rightarrow person(X,P) : \{\}$$

$$descLogic(X) \wedge author(X) \rightarrow \perp : \{ann(X,I_1,dl) = 1 \wedge ann(X,I_2,pers) = 1 \\ overlap(I_1,I_2) = 0\}$$

$$shop(X) \wedge editorialProd(X) \rightarrow \perp : \{ann(X,I_1,shop) = 1 \wedge ann(X,I_2,mag) = 1 \\ overlap(I_1,I_2) = 0\}$$

$$number(X) \wedge date(X) \rightarrow \perp : \{ann(X,I_1,num) = 1 \wedge ann(X,I_1,date) = 1 \\ overlap(I_1,I_2) = 0\}$$

Nota: Las fórmulas con anotación vacía **valen siempre**.

Ranking Queries (RQs)

- **RQ:** ¿cuáles son los átomos básicos que se infieren de una KB, en orden decreciente de probabilidad?
- **Semántica:** la probabilidad de que un **átomo básico** a se infiera corresponde a la **suma** de las probabilidades de los mundos posibles en los que la KB inducida **infiere** la CQ a .
- Recordemos que los mundos son eventos **disjuntos**.
- Lamentablemente, computar las probabilidades de los átomos es un problema **intratable**:
Teorema 1: Dada una KB, computar $Pr(a)$ es **#P-hard** en la complejidad de datos.
- Veremos algunas formas de atacar esta intratabilidad.

MLNs conjuntivas

- Consideremos una clase especial de MLNs:

Una *MLN conjuntiva* (cMLN) es una MLN en la cual todas las componentes (F, w) son tales que F es una conjunción de átomos.

- Esta restricción permite definir *clases de equivalencia* sobre los mundos posibles de M :
 - Informalmente, dos mundos son *equivalentes* si y sólo si satisfacen las *mismas fórmulas* en M .
 - Si bien aun hay una cantidad exponencial de clases, hay algunas propiedades que podemos aprovechar.
- Proposición 1: Dada una cMLN M , decidir si una clase de equivalencia C *es vacía* es un problema en PTIME.

MLNs conjuntivas: Propiedades

- Proposición 2: Dada una cMLN M y clase C , todos los elementos de C se pueden obtener en tiempo *lineal* en el tamaño de la *salida*.
- Proposición 3: Dada una cMLN M y mundos λ_1 y λ_2 , tenemos que si $\lambda_1 \sim_M \lambda_2$ entonces $Pr(\lambda_1) = Pr(\lambda_2)$.
- Proposición 4: Dada una cMLN M y mundos λ_1 y λ_2 , decidir si $Pr(\lambda_1) \leq Pr(\lambda_2)$ es un problema en PTIME.
- Lamentablemente, computar probabilidades *exactas* en cMLNs sigue siendo *intratable*:

Teorema 2: Sea a átomo; decidir si $Pr(a) \geq k$ es PP-hard en la complejidad de datos.

MLNs conjuntivas: Ejemplo

Consideremos la siguiente KB:

$$p(X) \rightarrow q(X) \quad : \{ \langle m(X) = 1 \rangle, \langle n(X) = 0 \rangle \}$$

$$p(a) \quad : \{ \langle m(a) = 1 \rangle \}$$

$$p(b) \quad : \{ \langle n(a) = 1 \rangle \}$$

$$p(c) \quad : \{ \langle m(c) = 1 \rangle, \langle n(c) = 0 \rangle \}$$

con cMLN $M = \{ (m(X), 1.5), (n(X), 0.8) \}$ y constantes $\{a, b, c\}$:

$$f_1: (m(a), 1.5), \quad f_2: (n(a), 0.8)$$

$$f_3: (m(b), 1.5), \quad f_4: (n(b), 0.8)$$

$$f_5: (m(c), 1.5), \quad f_6: (n(c), 0.8)$$

En este caso, tenemos $2^6 = 64$ clases (en este caso, un mundo por clase).

Algoritmo *anytimeRank*

Entradas: $KB = (O, M)$, predicado $stopCond$ (M no tiene variables)

1. Inicializar $score$, un mapeo de átomos a \mathbb{R} ; $i := 1$;
2. While $i \leq 2^{|M|}$ and $!stopCond$ do
 - a) $C := compMostProbEqClass(M, i)$; // Por construcción
 - b) $S := \emptyset$; $i := i + 1$;
 - c) While $(|S| \leq |C|)$ and $!stopCond$ do // Prop. 1
 - $\lambda := compWorld(C, S)$; // $\lambda \in C$ y $\lambda \notin S$ // Prop. 2
 - $S := S \cup \{\lambda\}$;
 - for all atoms $a \in atomicCons(O_\lambda)$ do
 - $score := score + \exp(\sum_{F_j \in M, C \models F_j} w_j)$ // Prop. 3
 - $out := out \cup atomicCons(O_\lambda)$
3. Retornar out ordenado en forma decreciente de $score$. // Prop. 4

Ejemplo (continuado)

$Class$	f_1	f_3	f_5	f_2	f_4	f_6	$\Sigma_j w_j$
C_1	1	1	1	1	1	1	6.9
C_2	1	1	1	1	1	0	6.1
C_3	1	1	1	1	0	1	6.1
C_4	1	1	1	0	1	1	6.1
C_5	1	1	1	1	0	0	5.3

Esta tabla muestra 5 de las 64 clases en el ejemplo, ordenadas de acuerdo al **score** asignado a sus mundos (*e* elevado al valor en la última columna).

Traza de *anytimeRank*

$O_{c_1} \models \{p(a), p(b)\}$

sumarle $e^{6.9}$ a $score(p(a))$ y $score(p(b))$

Traza de *anytimeRank*

$O_{C_1} \models \{p(a), p(b)\}$

sumarle $e^{6.9}$ **a** $score(p(a))$ **y** $score(p(b))$

$O_{C_2} \models \{p(a), p(b), p(c), q(c)\}$

sumarle $e^{6.1}$ **a** $score(p(a)), score(p(b)), score(p(c)),$ **y** $score(q(c))$

Traza de *anytimeRank*

$O_{C_1} \models \{p(a), p(b)\}$

sumarle $e^{6.9}$ **a** $score(p(a))$ **y** $score(p(b))$

$O_{C_2} \models \{p(a), p(b), p(c), q(c)\}$

sumarle $e^{6.1}$ **a** $score(p(a)), score(p(b)), score(p(c)),$ **y** $score(q(c))$

$O_{C_3} \models \{p(a)\}$

sumarle $e^{6.1}$ **a** $score(p(a))$

Traza de *anytimeRank*

$O_{C_1} \models \{p(a), p(b)\}$

sumarle $e^{6.9}$ **a** $score(p(a))$ **y** $score(p(b))$

$O_{C_2} \models \{p(a), p(b), p(c), q(c)\}$

sumarle $e^{6.1}$ **a** $score(p(a)), score(p(b)), score(p(c)),$ **y** $score(q(c))$

$O_{C_3} \models \{p(a)\}$

sumarle $e^{6.1}$ **a** $score(p(a))$

$O_{C_4} \models \{p(a), p(c), q(c)\}$

sumarle $e^{6.1}$ **a** $score(p(a)), score(p(c)),$ **y** $score(q(c))$

Traza de *anytimeRank*

$O_{C_1} \models \{p(a), p(b)\}$

sumarle $e^{6.9}$ **a** $score(p(a))$ **y** $score(p(b))$

$O_{C_2} \models \{p(a), p(b), p(c), q(c)\}$

sumarle $e^{6.1}$ **a** $score(p(a))$, $score(p(b))$, $score(p(c))$, **y** $score(q(c))$

$O_{C_3} \models \{p(a)\}$

sumarle $e^{6.1}$ **a** $score(p(a))$

$O_{C_4} \models \{p(a), p(c), q(c)\}$

sumarle $e^{6.1}$ **a** $score(p(a))$, $score(p(c))$, **y** $score(q(c))$

$O_{C_5} \models \{p(a), p(b), q(a)\}$

sumarle $e^{5.3}$ **a** $score(p(a))$, $score(p(b))$, **y** $score(q(a))$

Traza de *anytimeRank*

$$O_{C_1} \models \{p(a), p(b)\}$$

sumarle $e^{6.9}$ **a** $score(p(a))$ **y** $score(p(b))$

$$O_{C_2} \models \{p(a), p(b), p(c), q(c)\}$$

sumarle $e^{6.1}$ **a** $score(p(a))$, $score(p(b))$, $score(p(c))$, **y** $score(q(c))$

$$O_{C_3} \models \{p(a)\}$$

sumarle $e^{6.1}$ **a** $score(p(a))$

$$O_{C_4} \models \{p(a), p(c), q(c)\}$$

sumarle $e^{6.1}$ **a** $score(p(a))$, $score(p(c))$, **y** $score(q(c))$

$$O_{C_5} \models \{p(a), p(b), q(a)\}$$

sumarle $e^{5.3}$ **a** $score(p(a))$, $score(p(b))$, **y** $score(q(a))$

Supongamos que nos detenemos aquí; los puntajes parciales son:

$$score(p(a)) = 2530.18$$

$$score(p(b)) = 1638.47$$

$$score(p(c)) = 891.71$$

$$score(q(c)) = 891.71$$

Acotando el error de *anytimeRank*

- Teorema 3: En una ejecución de *anytimeRank* que ha analizado s mundos y t clases, la masa total de score *sin asignar* está acotada por arriba por:

$$U = (2^n - s) \cdot \exp\left(\sum_{F_j \in M, C_{t+1} \models F_j} w_j\right)$$

- Con este teorema se puede determinar un *orden parcial* demostrablemente *correcto* con respecto a los valores de probabilidad, usando los valores de score *parciales*.
- Por lo tanto, se puede utilizar como parte de la condición de *detención* de *anytimeRank*.

Evaluación empírica

Entorno experimental

- Framework implementado utilizando:
 - El sistema **Nyaya** para respuesta de consultas ontológicas (de Virgilio, Orsi, Tanca, and Torlone); y
 - El **ProbCog** toolbox (Jain, Barthels, and Beetz)
- Se derivaron ontologías usando el benchmark LUBM:
 - sampling aleatorio de una BD de 1.000 átomos;
 - generación aleatoria de una **cMLN** con a lo sumo 5 fórmulas, con a lo sumo 5 átomos y evidencia con a lo sumo 20 átomos;
 - generación aleatoria de anotaciones tal que al menos 70% de la KB está anotada con anotaciones no vacías.

Entorno experimental

Se usaron las siguientes consultas:

$q_1(X, Y, Z) \leftarrow \text{worksFor}(X, Y), \text{affiliatedOrganization}(Y, Z).$

$q_2(X, Y) \leftarrow \text{person}(X, Y), \text{teacherOf}(X, Y), \text{course}(Y).$

$q_3(X, Y, Z) \leftarrow \text{student}(X), \text{advisor}(X, Y), \text{facultyStaff}(Y),$
 $\text{takesCourse}(X, Z), \text{teacherOf}(Y, Z), \text{course}(Z).$

$q_4(X, Y) \leftarrow \text{person}(X, Y), \text{worksFor}(X, Y), \text{organization}(Y).$

$q_5(X, Y) \leftarrow \text{person}(X), \text{worksFor}(X, Y), \text{university}(Y),$
 $\text{hasAlumnus}(Y, X).$

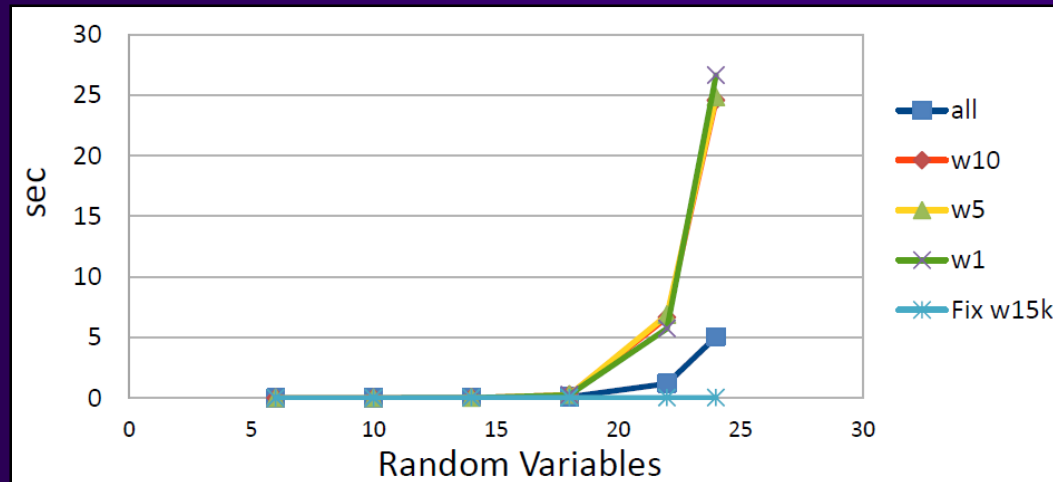
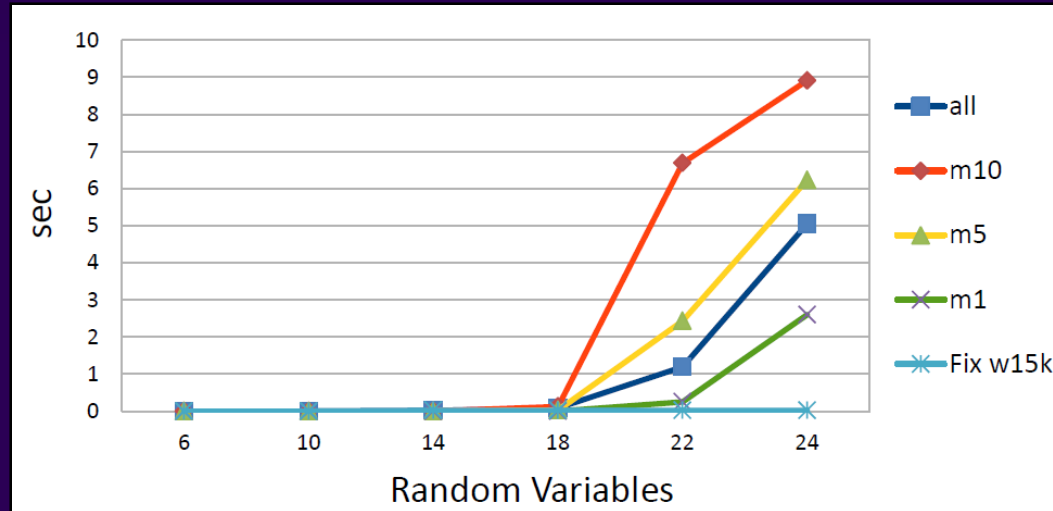
$q_6(X) \leftarrow \text{student}(X).$

$q_7(X) \leftarrow \text{fullProfessor}(X).$

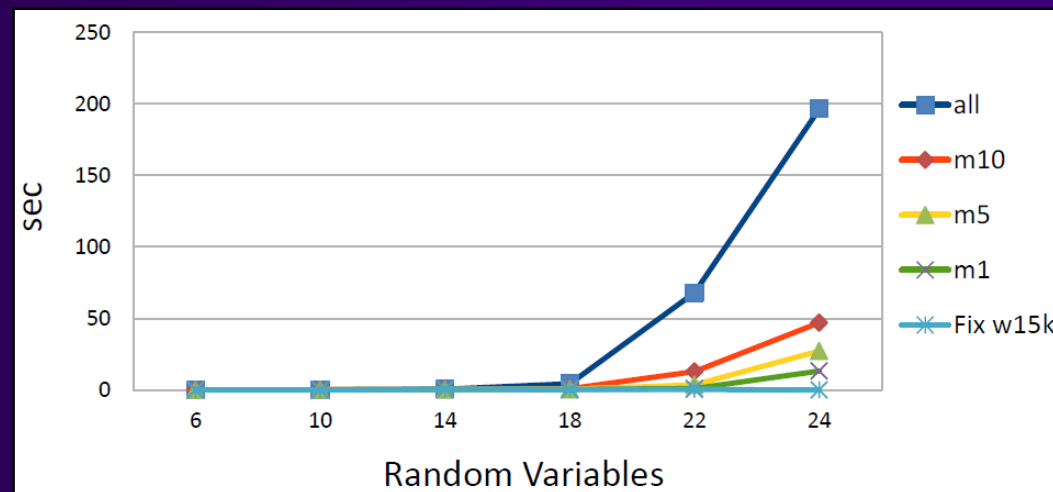
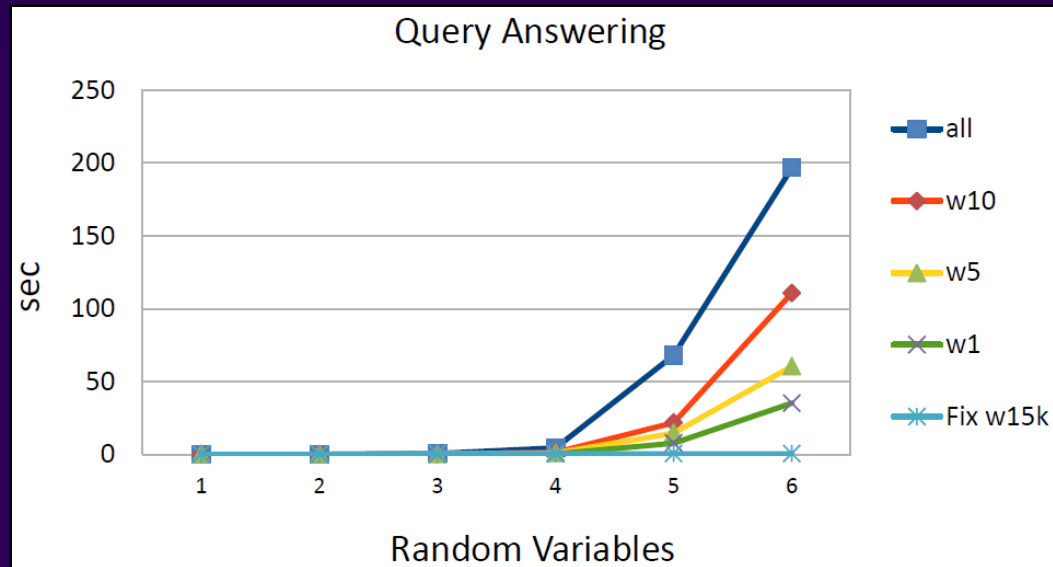
Entorno experimental

- Tarea a evaluar: **ranking** de respuestas a estas consultas.
- Diferentes estrategias de exploración para mundos:
 - Enumeración por fuerza bruta (Naive);
 - Muestreo **Monte Carlo** (MC) del 1%, 5% y 10% del conjunto de mundos posibles, más una muestra fija de 15.000 mundos; y
 - Enumeración **decreciente**, con los mismos tamaños (*Top-down*).
- Experimento 1: Comparación de tiempos de ejecución:
 - Tiempo para hacer el **sampling**: Naive vs. MC;
 - Tiempo para hacer inferencia ontológica usando los mundos muestreados; y
 - Performance de respuesta de consultas: *Top-down* vs. MC.

Resultados de tiempo: *Sampling*



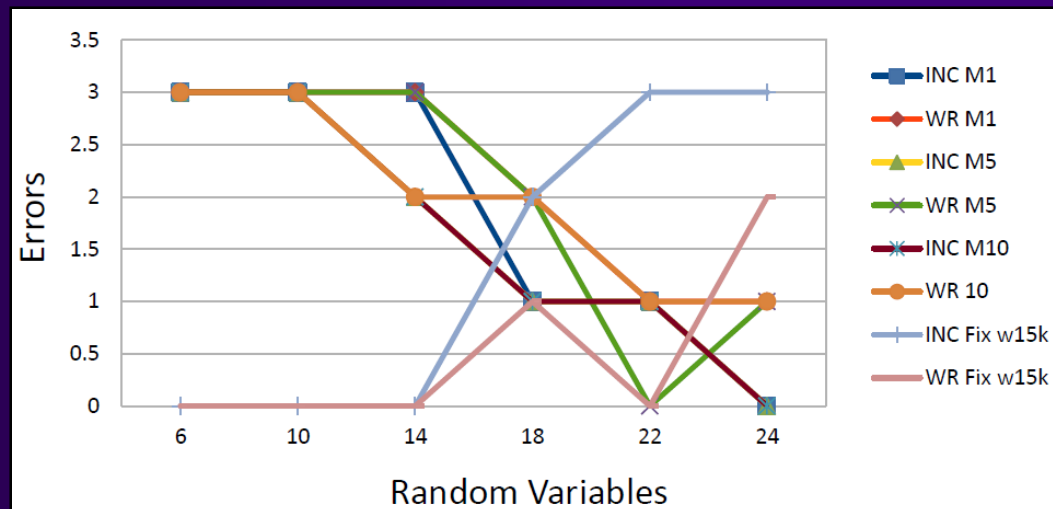
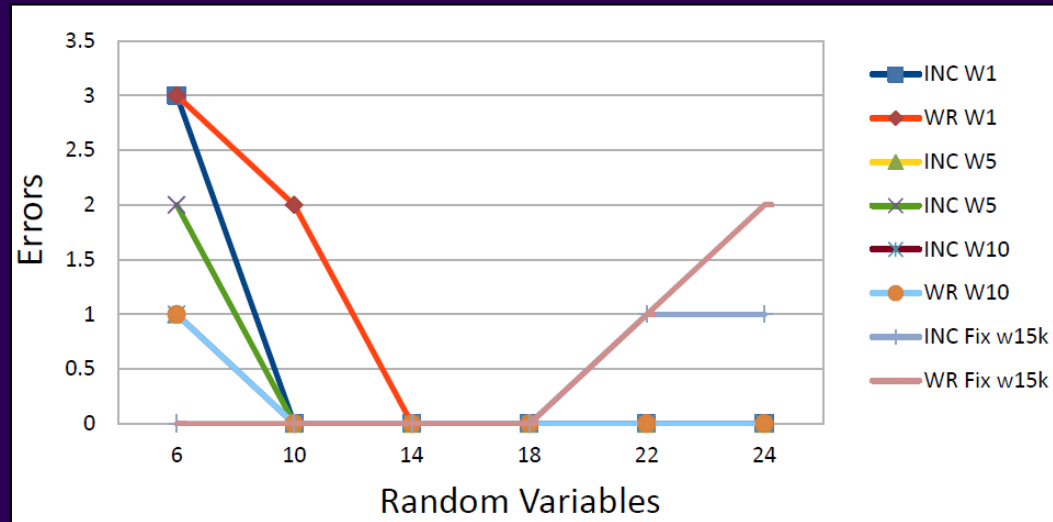
Resultados de tiempo: Consultas



Entorno experimental

- Experimento 2: **Precisión**:
 - Comparación de resultados generados por las aproximaciones vs. el algoritmo **naive**, que es sano y completo.
 - Dos tipos de **errores** posibles:
 - Ranking **incompleto**: faltan tuplas en la respuesta;
 - Ranking **equivocado**: la respuesta es sana y completa en cuanto a contenido, pero hay errores de ordenamiento.
 - El tamaño de la salida de las consultas (rankings) fue de entre 30 y 50 tuplas.

Resultados de precisión: Tasa de error



Otras direcciones

- Existen otros tipos de consultas:
 - Consultas **threshold**: ¿cuál es el conjunto de átomos que se infieren con probabilidad al menos p ?
 - Consultas **conjuntivas**: ¿cuál es la probabilidad con la que una conjunción de átomos se infiere?
- Hay resultados preliminares acerca de la tratabilidad de cada tipo de consulta usando los dos tipos de sampling.
- También se están estudiando otros fragmentos de MLNs.

Referencias

- [Luk12] T. Lukasiewicz, M. V. Martinez, G. Orsi, G. I. Simari: “*Heuristic Ranking in Tightly Coupled Probabilistic Description Logics*”. Proc. of UAI 2012, pp. 554–563.
- [Got13] G. Gottlob, T. Lukasiewicz, M. V. Martinez, G. I. Simari: “*Query Answering Under Uncertainty in Datalog+/- Ontologies*”. AMAI 69(1):37–72, 2013.
- [Luk20] T. Lukasiewicz, M. V. Martinez, G. Orsi, G. I. Simari: “*Exact and Approximate Query Answering in Tightly Coupled Probabilistic Datalog+/-*”. En preparación, 2020.
- [PrOQAW] Project PrOQAW (UK Engineering and Physical Sciences Research Council): <http://www.cs.ox.ac.uk/projects/PrOQAW/>
- [DIADEM] Project DIADEM (European Research Council):
<http://diadem.cs.ox.ac.uk/>
- [Nyaya] Nyaya Ontological Query Answering System:
<http://mais.dia.uniroma3.it/Nyaya/>
- [ProbCog] ProbCog MLN Toolbox:
<http://ias.cs.tum.edu/research/probcog>