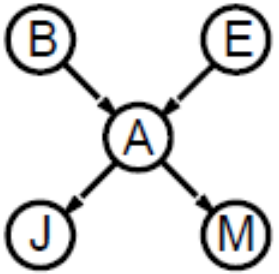
Progetto su Incertezza

IALab A.A. 2019/2020

Reti Bayesiane

Pruning: Nodi Irrilevanti

Consider the query $P(JohnCalls|Burglary = true)$



$$P(J|b) = \alpha P(b) \sum_e P(e) \sum_a P(a|b,e) P(J|a) \sum_m P(m|a)$$

Sum over m is identically 1; M is **irrelevant** to the query

Thm 1: Y is irrelevant unless $Y \in Ancestors(\{X\} \cup \mathbf{E})$

Here, $X = JohnCalls$, $\mathbf{E} = \{Burglary\}$, and
$Ancestors(\{X\} \cup \mathbf{E}) = \{Alarm, Earthquake\}$
so $MaryCalls$ is irrelevant

(Compare this to backward chaining from the query in Horn clause KBs)

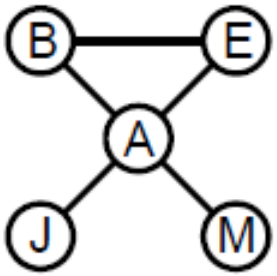
Pruning: Nodi Irrilevanti

Defn: moral graph of Bayes net: marry all parents and drop arrows

Defn: $\mathbf{A}$ is m-separated from $\mathbf{B}$ by $\mathbf{C}$ iff separated by $\mathbf{C}$ in the moral graph

Thm 2: Y is irrelevant if m-separated from X by $\mathbf{E}$

For $P(JohnCalls|Alarm = true)$, both $Burglary$ and $Earthquake$ are irrelevant



Per dettagli si **m-separation** vedete **sezione 2.3** di questo documento:

https://ourspace.uregina.ca/bitstream/handle/10294/7635/dosSantos_Andre_200334126_MSC_CS_Fall2016.pdf

Pruning: Archi Irrilevanti

Given a Bayesian network **N** and a query (**Q, e**), one can also eliminate some of the network edges and reduce some of its CPTs without affecting its ability to compute the joint marginal Pr(**Q, e**) correctly. In particular, for each edge $U \rightarrow X$ that originates from a node U in **E**, we can:

1. Remove the edge $U \rightarrow X$ from the network.
2. Replace the CPT $\Theta_{X|\mathbf{U}}$ for node X by a smaller CPT, which is obtained from $\Theta_{X|\mathbf{U}}$ by assuming the value u of parent U given in evidence **e**. This new CPT corresponds to $\sum_U \Theta^u_{X|\mathbf{U}}$.

The result of this operation is denoted by pruneEdges(**N**, **e**) and we have the following result.

Pruning: Archi Irrilevanti

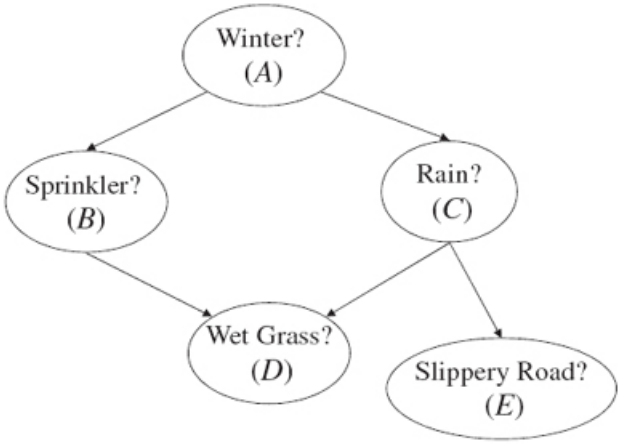
Theorem 6.5. *Let* $\mathbf{N}$ *be a Bayesian network and let* $\mathbf{e}$ *be an instantiation. If* $\mathbf{N}'$ = pruneEdges($\mathbf{N}$, $\mathbf{e}$), *then* Pr($\mathbf{Q}$, $\mathbf{e}$) = Pr′($\mathbf{Q}$, $\mathbf{e}$) *where* Pr *and* Pr′ *are the probability distributions induced by networks* $\mathbf{N}$ *and* $\mathbf{N}'$, *respectively.*

?

Pruning: Archi Irrilevanti

Questa rete con

C=false:



A	Θ_A
true	.6
false	.4

A	B	$\Theta_{B\|A}$
true	true	.2
true	false	.8
false	true	.75
false	false	.25

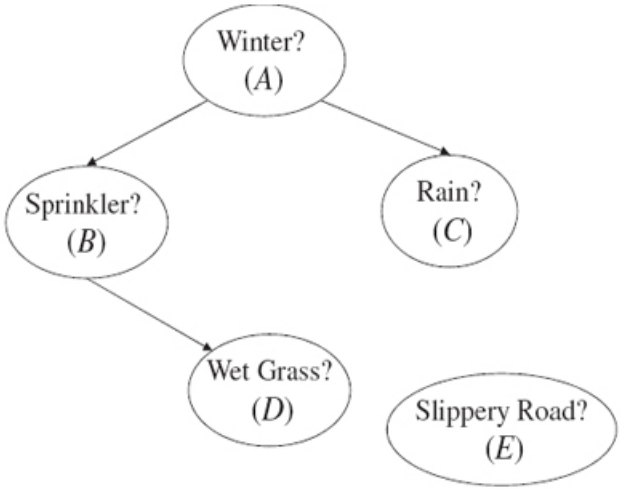
A	C	$\Theta_{C\|A}$
true	true	.8
true	false	.2
false	true	.1
false	false	.9

B	C	D	$\Theta_{D\|BC}$
true	true	true	.95
true	true	false	.05
true	false	true	.9
true	false	false	.1
false	true	true	.8
false	true	false	.2
false	false	true	0
false	false	false	1

C	E	$\Theta_{E\|C}$
true	true	.7
true	false	.3
false	true	0
false	false	1

Pruning: Archi Irrilevanti

Diventa:



A	Θ_A
true	.6
false	.4

A	B	$\Theta_{B\|A}$
true	true	.2
true	false	.8
false	true	.75
false	false	.25

A	C	$\Theta_{C\|A}$
true	true	.8
true	false	.2
false	true	.1
false	false	.9

B	D	$\sum_C \Theta_{D\|BC}^{C=\text{false}}$
true	true	.9
true	false	.1
false	true	0
false	false	1

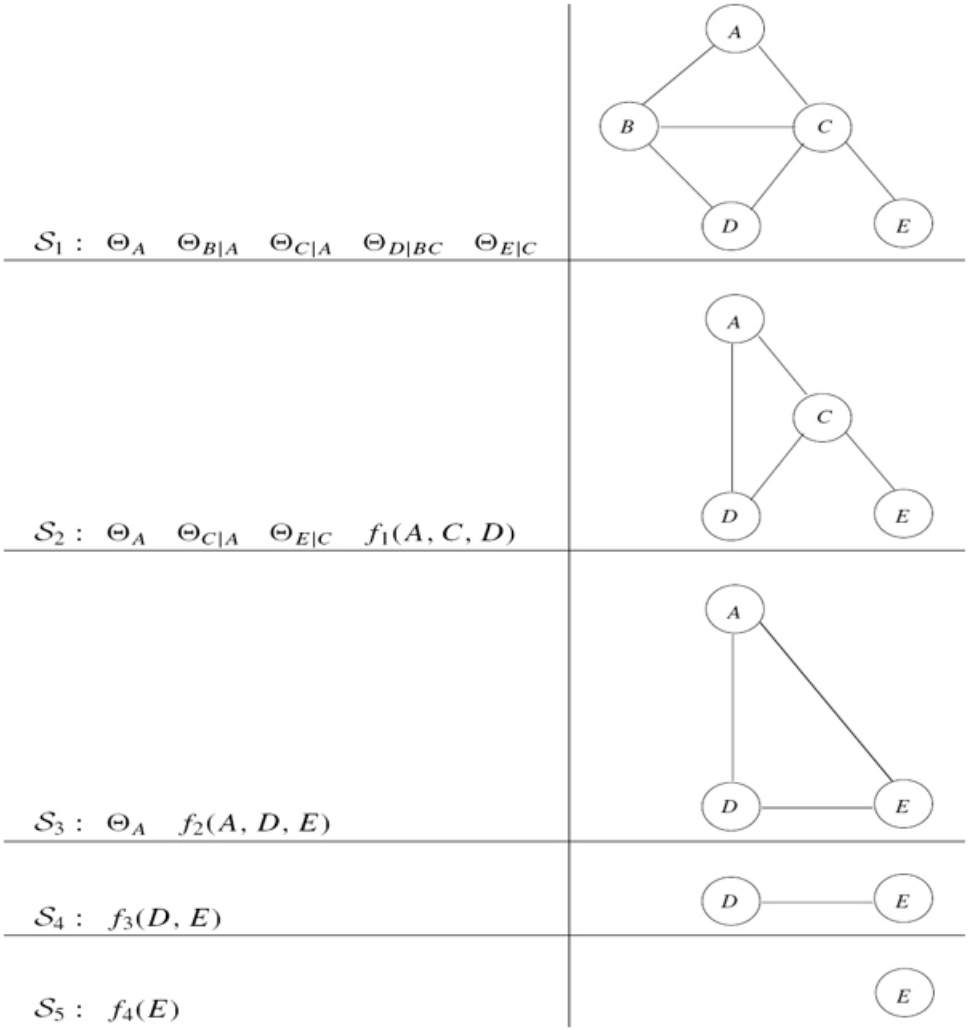
E	$\sum_C \Theta_{E\|C}^{C=\text{false}}$
true	0
false	1

Ordinamento

Definition 6.4. Let $f_1,\ldots, f_n$ be a set of factors. The *interaction graph* G of these factors is an undirected graph constructed as follows. The nodes of G are the variables that appear in factors $f_1,\ldots, f_n$. There is an edge between two variables in G iff those variables appear in the same factor.

■

Ordinamento



Ordinamento

Algorithm 5 `MinDegreeOrder`$(\mathcal{N}, \mathbf{X})$

input:

$\mathcal{N}$: Bayesian network

$\mathbf{X}$: variables in network $\mathcal{N}$

output: an ordering π of variables $\mathbf{X}$

main:

1: $G \leftarrow$ interaction graph of the CPTs in network $\mathcal{N}$
2: **for** $i = 1$ to number of variables in $\mathbf{X}$ **do**
3: $\quad \pi(i) \leftarrow$ a variable in $\mathbf{X}$ with smallest number of neighbors in G
4: $\quad$ add an edge between every pair of non-adjacent neighbors of $\pi(i)$ in G
5: $\quad$ delete variable $\pi(i)$ from G and from $\mathbf{X}$
6: **end for**
7: **return** π

Ordinamento

Algorithm 6 `MinFillOrder`$(\mathcal{N}, \mathbf{X})$

input:

$\mathcal{N}$: Bayesian network

$\mathbf{X}$: variables in network $\mathcal{N}$

output: an ordering π of variables $\mathbf{X}$

main:

1: $G \leftarrow$ interaction graph of the CPTs in network $\mathcal{N}$
2: **for** $i = 1$ to number of variables in $\mathbf{X}$ **do**
3: $\pi(i) \leftarrow$ a variable in $\mathbf{X}$ that adds the smallest number of edges on Line 4
4: add an edge between every pair of non-adjacent neighbors of $\pi(i)$
5: delete variable $\pi(i)$ from G and from $\mathbf{X}$
6: **end for**
7: **return** π

Pruning e ordine: esperimenti

Eseguire esperimenti su **molte BN diverse** confrontando il tempo di VE al variare di:

1. dimensioni e "complessità" ("lontananza" da un polytree) della BN
2. numero e posizione variabili evidenza e di query
3. ordine variabili (topologico inverso, min-degree, min-fill)

Non è un esercizio di programmazione!

E' importante fare degli esperimenti ragionati e valutare criticamente i risultati

Reti di Test

- per testare il vostro Progetto BN potete in prima istanza usare le semplici BN viste nella teoria e negli esercizi, come:
  - rete "earthquake"
  - rete Sprinkler
  - ecc.
- per sperimentare con reti più grandi potete far riferimento al Bayesian Network Repository all'URL:

http://www.bnlearn.com/bnrepository/

Reti di Test

- le reti nel BNR sono fornite in vari formati, tra cui:
  - BIF (formato di interchange, purtroppo però una versione vecchia non basata su XML)
  - NET (consigliata per leggerle con SamIam)
- a sua volta SamIam vi permette di salvare le reti caricate in XMLBIF (formato di interchange basato su XML)
- usate poi il parser per XMLBIF che vi ho fornito qui:

  https://gitlab2.educ.di.unito.it/ialabstudenti/bnparser

Reti di Test: generatori

- per test ancora più completi potete usare un generatore di BN come:

http://sites.poli.usp.br/pmr/ltd/Software/BNGenerator/

Reti Bayesiane Dinamiche

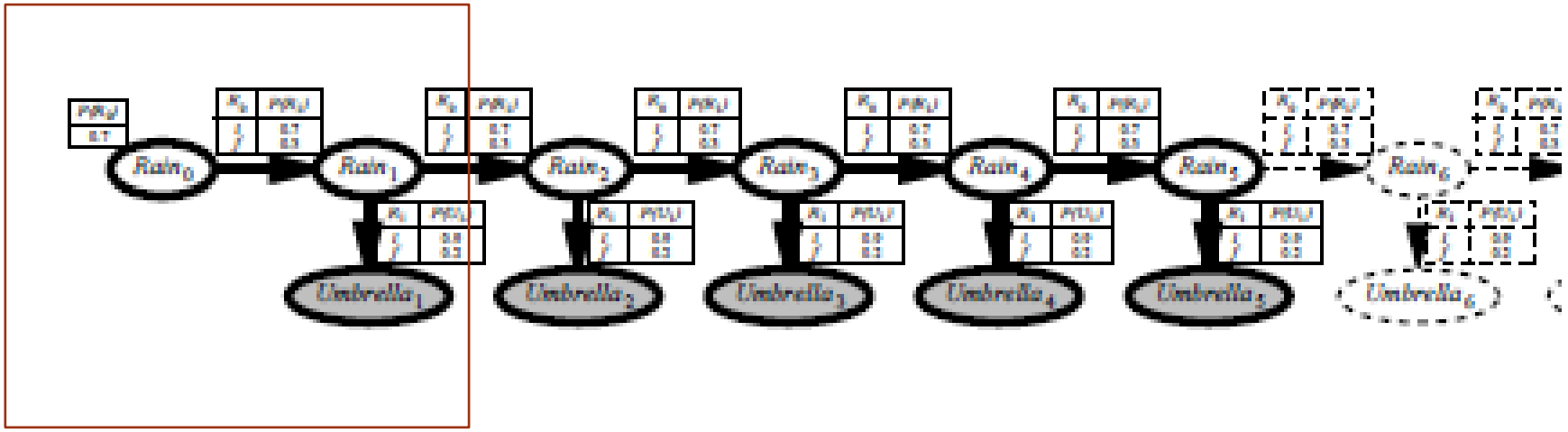
Progetto su DBN

- sulle BN statiche abbiamo visto in classe:
  - **Simple Query** con algoritmo di **Variable Elimination**
- sulle DBN abbiamo visto in classe:
  - algoritmo di **Rollup filtering**
- la libreria **aima-core** fornisce:
  - rappresentazione **BN** statiche
  - rappresentazione **DBN**
  - algoritmo di **Variable Elimination** capace di rispondere a **Conjunctive Simple Query** su BN statiche
  - algoritmo **approssimato** di inferenza su **DBN** (lo usiamo per verifica di correttezza)

Inferenza su DBN con Rollup filtering

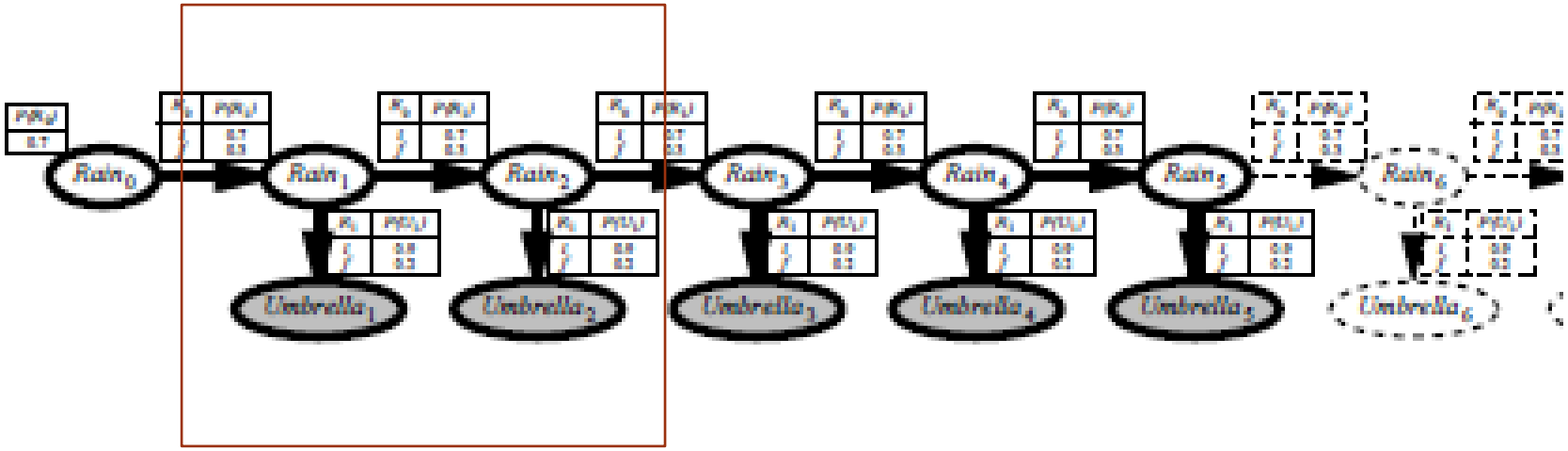
1. partire dall'algoritmo di Variable Elimination per BN statiche
2. modificarlo in modo che effettui il Rollup filtering su una DBN (considerando due "slice" per volta)
3. **attenzione**: conviene restituire il risultato come insieme di fattori, senza moltiplicarli tutti in un unico fattore (ultime operazioni della *eliminationAsk*)
4. oltre alla DBN, occorre naturalmente fornire all'algoritmo una sequenza di osservazioni $\mathbf{e}_1, \ldots, \mathbf{e}_m$ (dato che parliamo di DBN, le variabili evidenza possono essere più di una a ciascun istante di tempo)

Inferenza su DBN con Rollup filtering



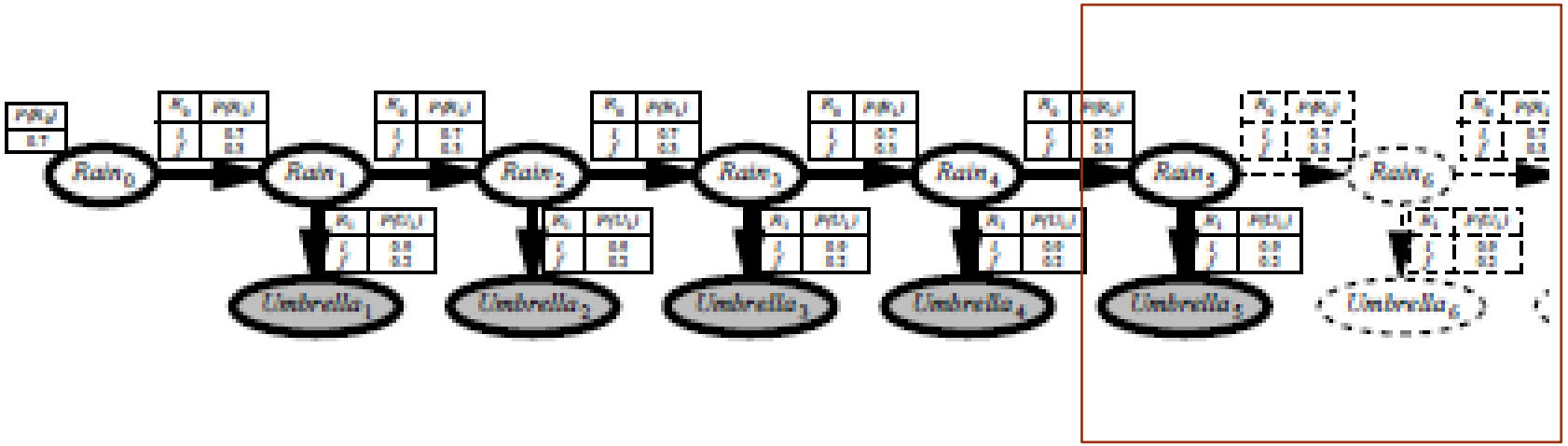
slice 0 e 1

Inferenza su DBN con Rollup filtering



slice 1 e 2

Inferenza su DBN con Rollup filtering



slice t e (t+1)

Esperimenti

Eseguire esperimenti su **diverse DBN** confrontando il tempo di filtering al variare di:

1. numero di variabili di stato $\mathbf{X}_t$ e complessità della relazione tra $\mathbf{X}_t$ e $\mathbf{X}_{t+1}$ e tra $\mathbf{X}_t$ e $\mathbf{E}_t$
2. ordine variabili (topologico inverso, min-degree, min-fill)

Non è un esercizio di programmazione!

E' importante fare degli esperimenti ragionati e valutare criticamente i risultati

DBN di test per il Progetto

- per testare il vostro Progetto DBN potete in prima istanza usare la semplice DBN (di fatto, un HMM) Umbrella visto a teoria e nell'esercizio su HMM
- per fare prove più significative potete:
  - "dinamizzare" le BN statiche usate nel primo Progetto
  - non devono essere reti molto grandi, ma permettervi di avere fino a 10-20 variabili di stato
  - se vi è comodo possono esserci variabili hidden oltre agli stati $\mathbf{X}_t$ e alle evidenze $\mathbf{E}_t$