Progetto su Incertezza

IALab A.A. 2018/2019

MPE e MAP

- abbiamo visto in classe:
  - **Simple Query** con algoritmo di **Variable Elimination**
  - definizioni di **MAP** e **MPE**
- la libreria **aima-core** fornisce:
  - rappresentazione BN
  - algoritmo di **Variable Elimination** capace di rispondere a **Conjunctive Simple Query**

MPE e MAP

- **MPE** si resolve con un metodo simile a VE ma:
  - anziché fare il "sumout" delle variabili facciamo il "maxout"
  - dobbiamo "ricordarci" gli assegnamenti alle variabili che massimizzano la probabilità
- **MAP** è un po' più complicato (perché è più generale):
  - prima dobbiamo processare le variabili non-MAP con il sumout
  - poi processare le variabili MAP con il maxout

MPE e MAP: implementazione

Implementare le inferenze MPE e MAP

1. estendere AIMA core

MPE e MAP: esperimenti

Eseguire esperimenti su una o più BN di esempio confrontando MPE e MAP al variare di:

1. dimensioni e "complessità" ("lontananza" da un polytree) della BN
2. variabili evidenza
3. variabili MAP

   NOTA: MAP può essere **molto più inefficiente** di MPE a causa del fatto che deve "processare" le variabili MAP dopo quelle non-MAP: cercate di farlo vederlo

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

Kalman Filter

- abbiamo visto in classe:
  - Kalman Filter su una variabile per transizioni di stato/osservazioni lineari
  - Kalman Filter su più variabili per transizioni di stato/osservazioni lineari
- esistono diverse librerie per l'implementazione di KF, es.:
  - Apache commons ne fornisce una implementazione nella classe filter.KalmanFilter

https://commons.apache.org/proper/commons-math/userguide/filter.html

  - la libreria EJML offre operazioni su matrici e vari modi per implementare i KF:

https://ejml.org/wiki/index.php?title=Example_Kalman_Filter

Kalman Filter: implementazione

Implementare l'inferenza con i Kalman Filter generali

1. l'estensione può essere "organica" alla libreria AIMA nel senso che tenta di usare ed estendere le interfacce e classi già presenti in AIMA core
2. oppure, se lo preferite, essere scollegata dalla libreria AIMA

Scrivere un programma/libreria che:

1. possa simulare un processo con transizioni di stato e osservazioni lineari in 2 variabili (es: coordinate x,y di un oggetto; posizione e velocità di un oggetto) **NOTA**: simulate anche il "rumore" Gaussiano per transizioni di stato e osservazioni
2. tracci lo stato del sistema con un KF
3. stampi a video:
   a. lo stato "vero" e quello stimato del Sistema
   b. l'errore (stimato) di processo
   c. il Kalman gain

Kalman Filter: esperimenti

Usare il programma per eseguire esperimenti da analizzare criticamente nella relazione:

1. variare il ”rumore” di transizione e di osservazione (usate pure matrici di covarianza diagonali)
   a. provare errore nullo, medio, molto alto
   b. provare varie combinazioni: es. errore di processo nullo e di osservazione molto alto, tutti e due nulli, ecc.
2. variare la P(.) iniziale dello stato
3. (facoltativo) simulare un processo (più o meno) non lineare e vedere come si comporta il KF

Non è un esercizio di programmazione!

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