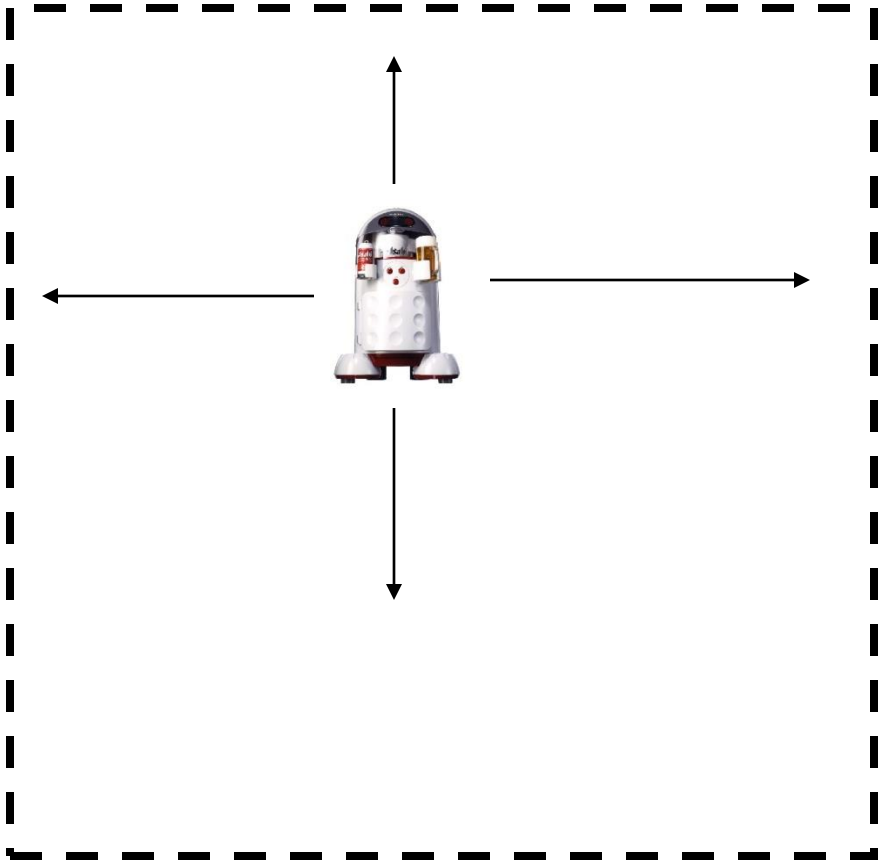
Advanced HMMs

- Factorial HMM's
- Input-output HMMs
- Dynamic Bayesian Networks (DBNs)

Coupling of hidden states

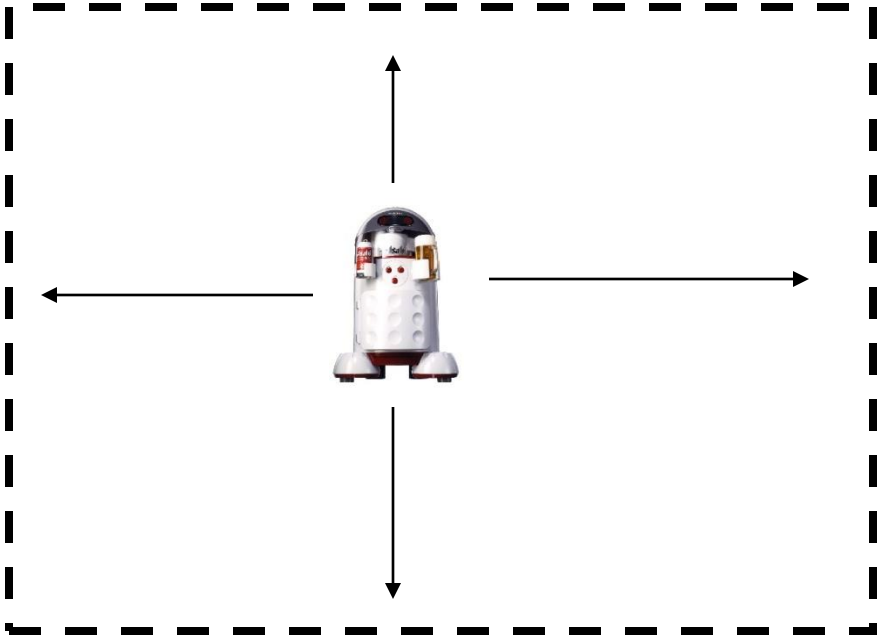
- A robot for tourists
- Location $\in$ {K by K grid}
- Language $\in$ {English, Spanish, French}
- Talk $\in$ {yes, no}

How do we design a HMM for this robot?



HMM for roboguide

- States: triplets {Loc, Lan, Tal}
- Emissions: based on location and presence / absence of a person
- Transitions: From one triplet to another



Example of transition:

P({Loc = (i,j), Lan = E, Tal = y} | ({Loc = (i,j-1), Lan = E, Tal = n})

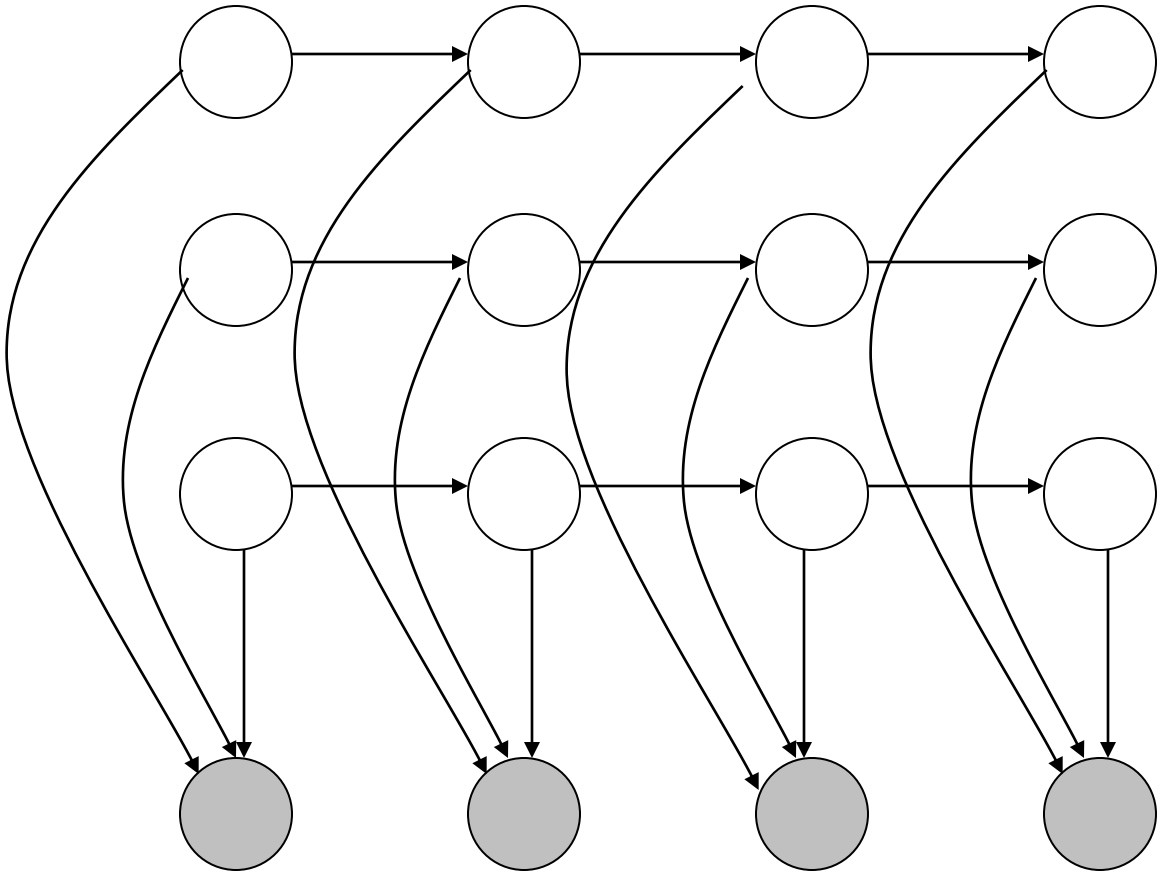
Problems?

Decoupling of states

- In many cases each state needs to be represented by a vector of attributes
- In these cases the transition between attributes may not depend on all the other attributes

  - For example, given a current location the next location is independent of the language being used
- In such cases it is better to use a different representation that is less complex and still captures the correct model

Factorial HMMs



Learning and inference in factorial HMMs

- M step: Same as in HMMs:

  - given expected state assignments compute initial, transition and emission probabilities (could couple states)

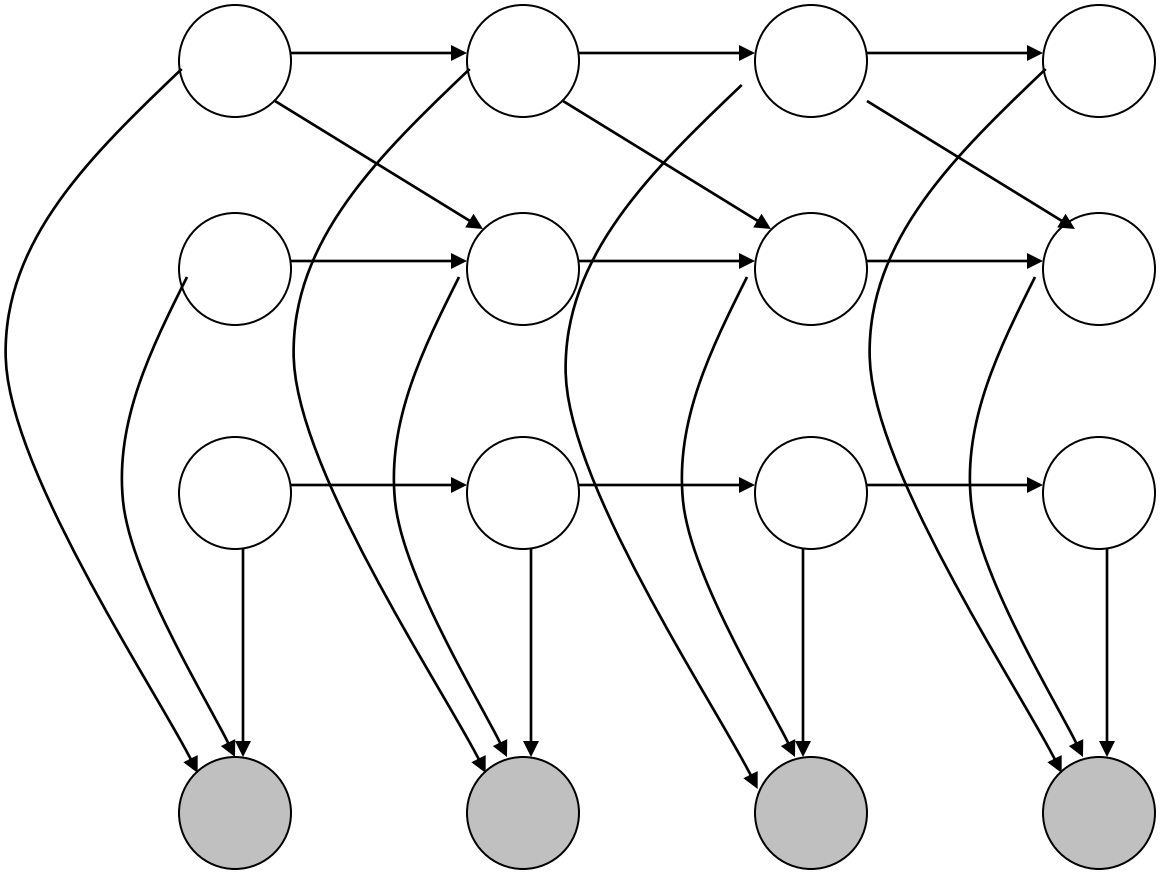
- E step: Harder, we cannot solve efficiently any more

  - The observations couple the states and the E step can be exponential in the number of different types of states

  - In practice people usually use a sampling (Monte Carlo) method for this task.

Factorial HMMs

Can also be used to represent relationships between different types of states. Again, inference is hard but if the states are known (estimated) we can compute transition and other probabilities (but we need more data).

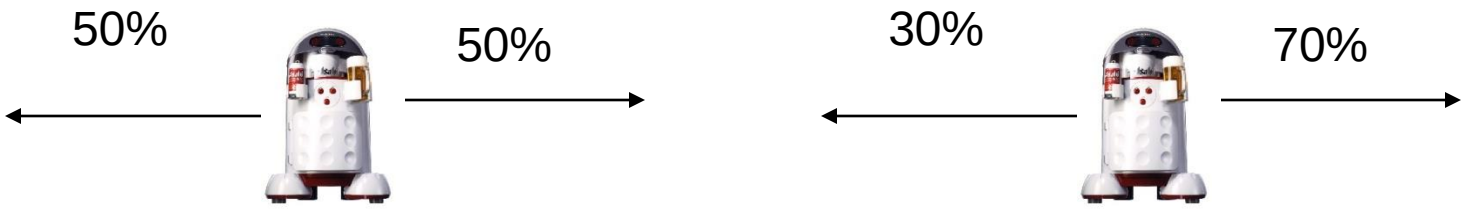


Advanced HMMs

- Factorial HMM's √
- Input-output HMMs
- Dynamic Bayesian Networks (DBNs)

Static input

- Assume we are building a robot tracker which determines the location of a robot based on sensor information
- The model is to be used for 3 different types of robots, each has a slightly different speed and preference as to the next location



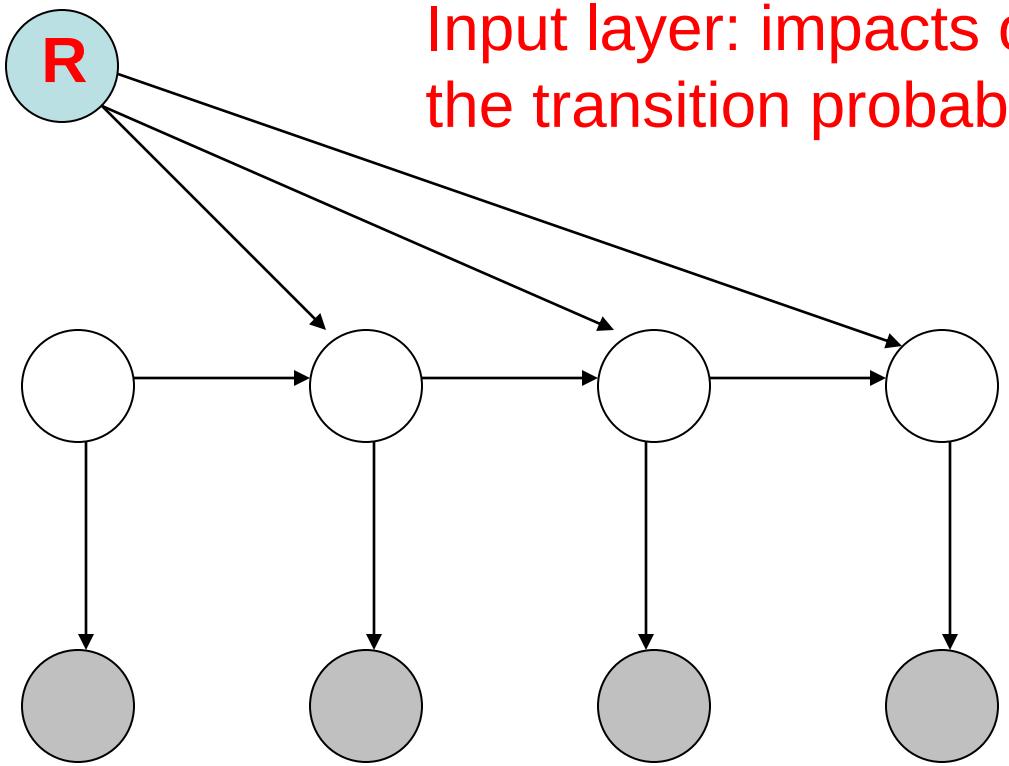
Handling the robotype model

- We can always build separate models for the three different robots
- But this is a waste

  - All robots share the same output probabilities

  - Perhaps they also share other properties (language, etc.).
- Instead, it would be better if we can design one model that will fit all of them

Input-output HMMs



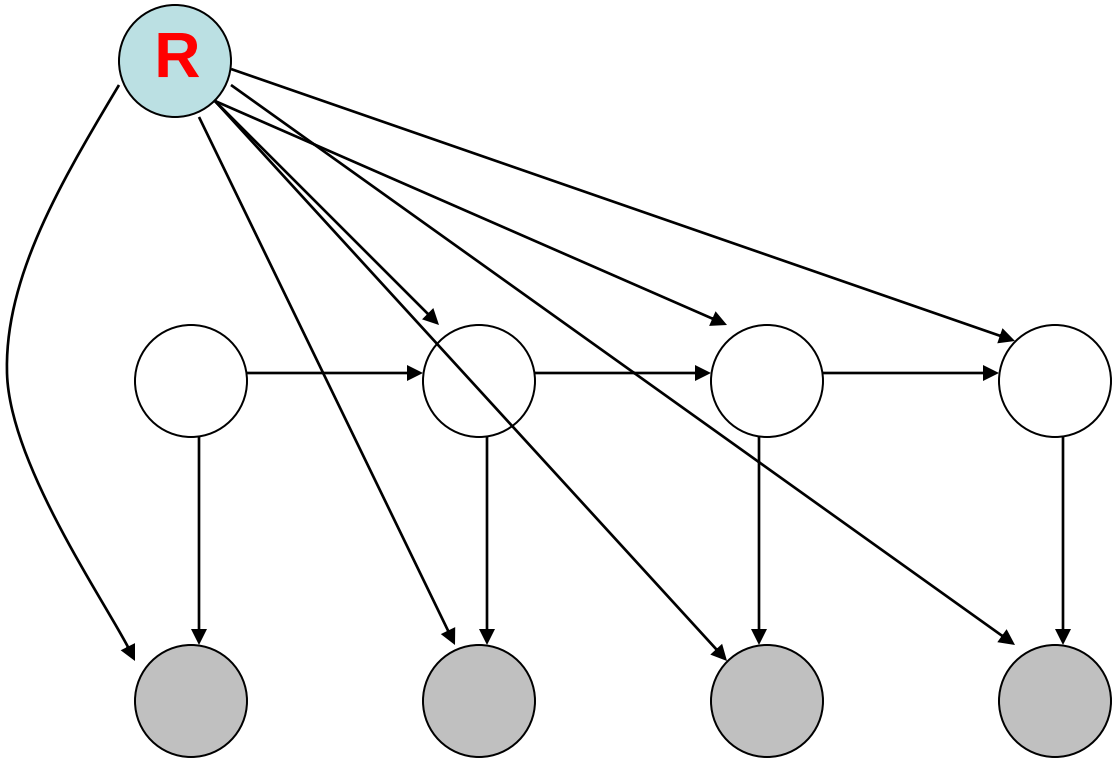
Input layer: impacts only the transition probabilities

$$p(Q,O \mid M) = \pi(q_1)\, p(o_1 \mid q_1) \prod_t p(q_t \mid q_{t-1}, R)\, p(o_t \mid q_t)$$

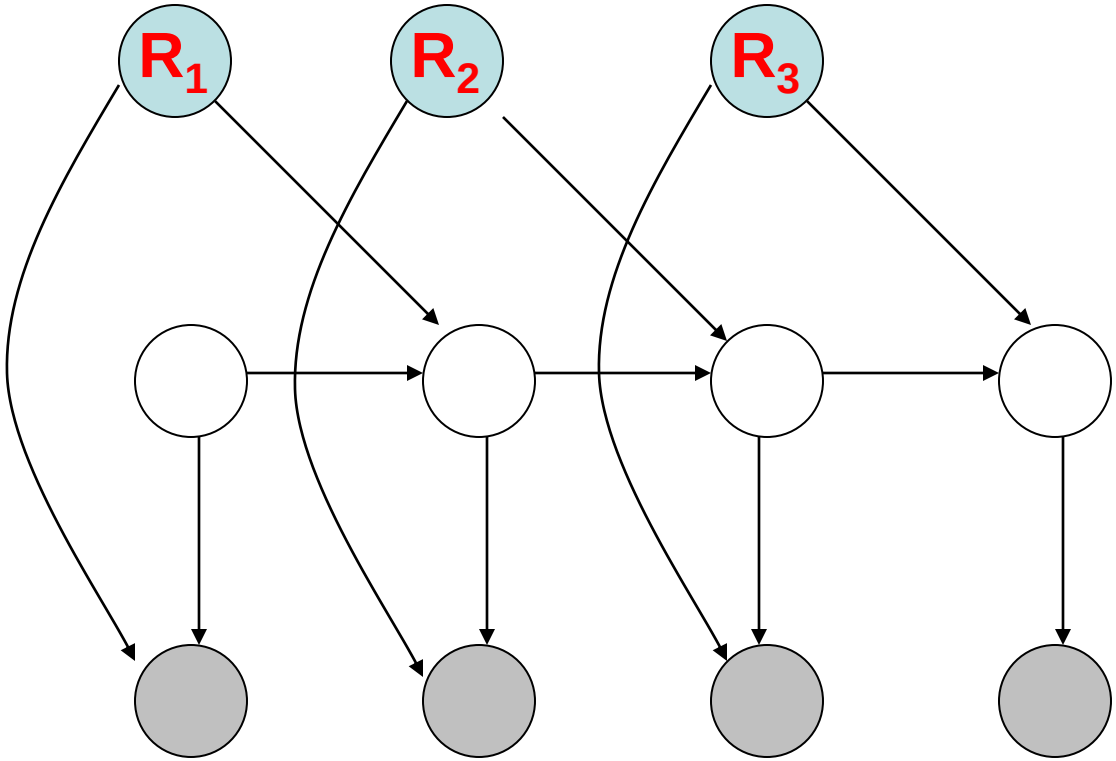
Learning and inference in input-output HMMs

- Depends on how the transition probability is modeled. Since R is given we can just learn a new transition table for each R value (note that the emission probabilities are still the same regardless of R).
- In some cases the transition probability may take on a different format (for example, logistic regression classifier). For such transitions learning is harder.

More general Input-output HMMs



More general Input-output HMMs



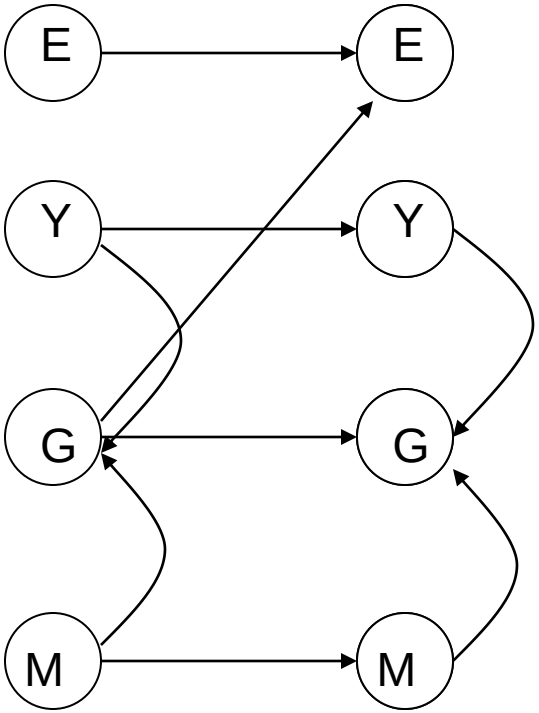
Advanced HMMs

- Factorial HMM's √
- Input-output HMMs √
- Dynamic Bayesian Networks (DBNs)

Predicting stock prices

- If we knew the price for Microsoft, Yahoo and Ebay today and the price of Google yesterday, could we predict the new price for Google?
- In these and other cases there is no hidden states but there is a strong dependency over time

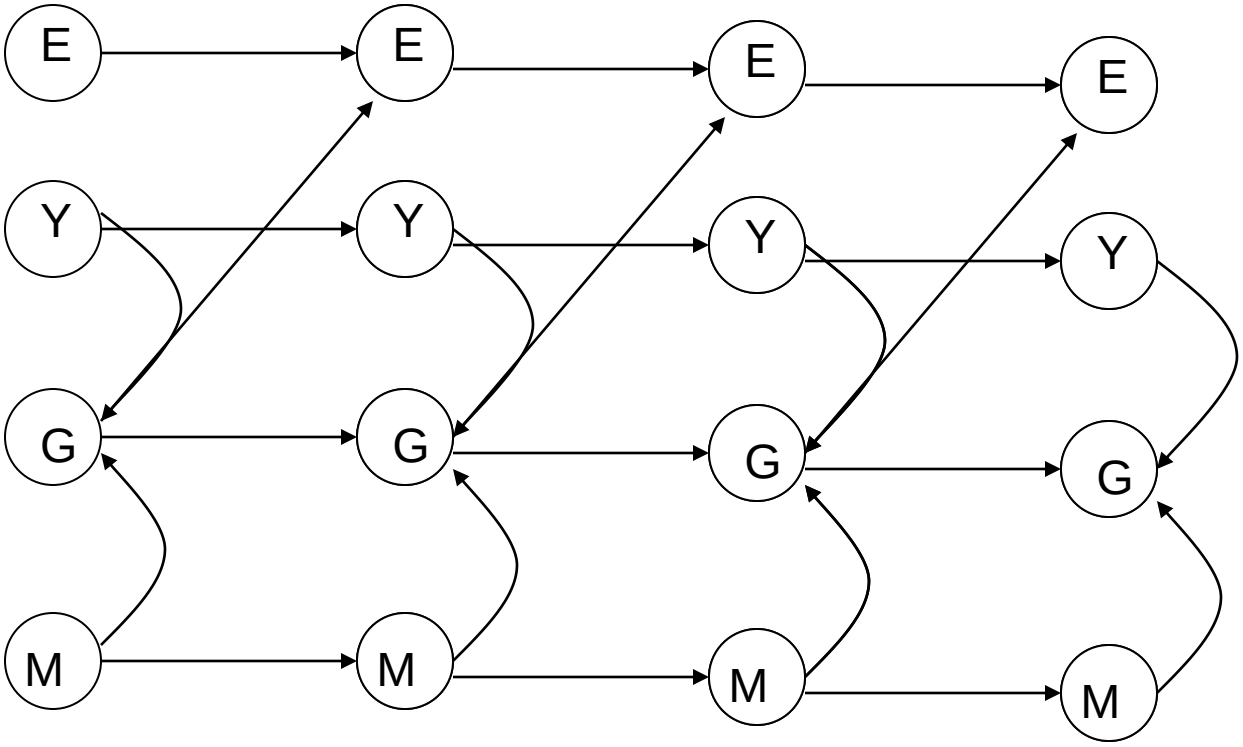
Dynamic Bayesian networks (DBNs)



- DBNs are an extension of Bayesian networks
- They follow the same semantics
- But they are repeated over time and so loops (between time units) are allowed.

$$P(X) = \prod_i p(x_i \mid Pa(x_i))$$

Dynamic Bayesian networks (DBNs)



$$P(X) = \prod_i p(x_i \mid Pa(x_i))$$

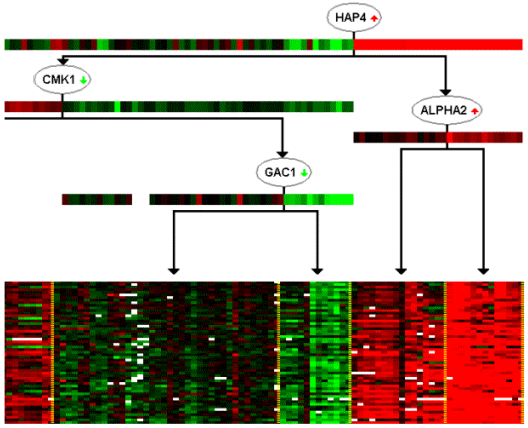
Learning and inference in DBNs

- This is really more similar to a Bayesian network (BN) than to a HMM
- Like all BNs, learning and inference is NP hard
- Which leads to several approximation methods:
  - Hill climbing
  - Annealing
  - Greedy algorithms
  - etc.

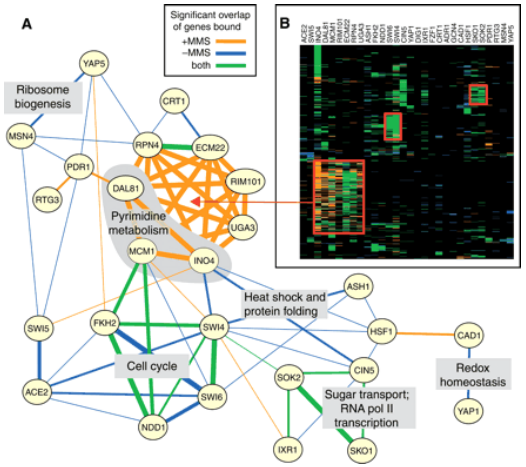
What you should know

- Why HMMs? Which applications are suitable?
- Learning HMMs: EM algorithm (Baum-Welch)
- Extensions of HMMs

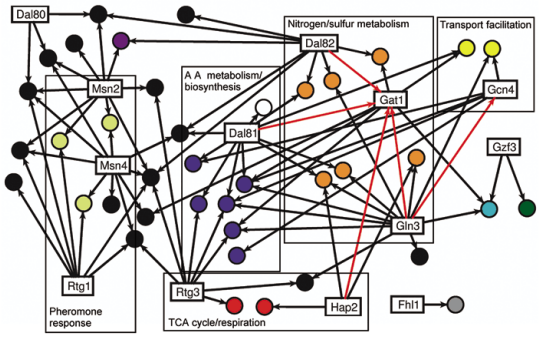
reconstructed networks are static …



Segal et al *Nature Genetics* 2003



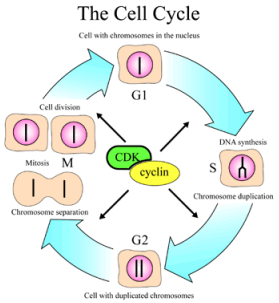
Workman et al *Science* 2006



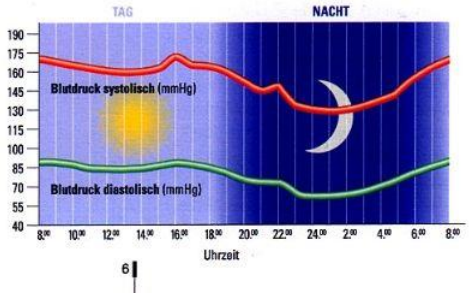
Bar-Joseph et al *Nature Biotechnology* 2003

… but biological systems are dynamic

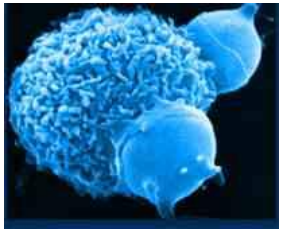
Cell cycle



Circadian rhythm

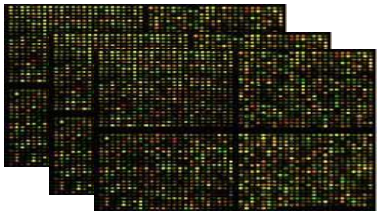


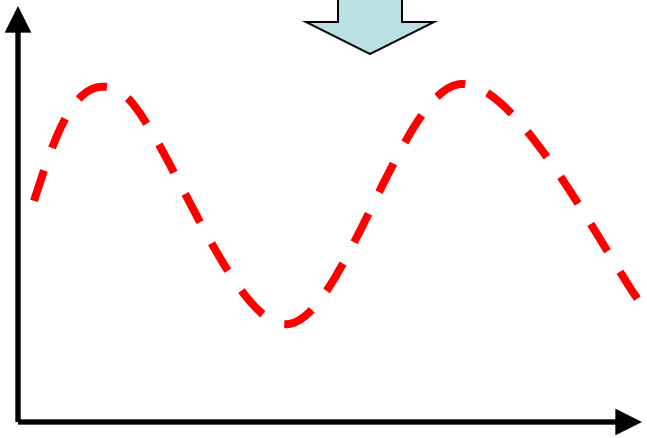
Immune response



Key problem: Most high-throughput data is static

Time-series measurements



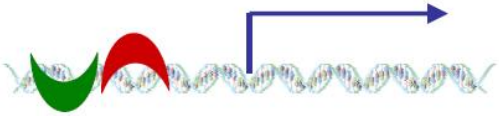


Static data sources

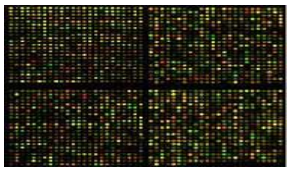
DNA

motif

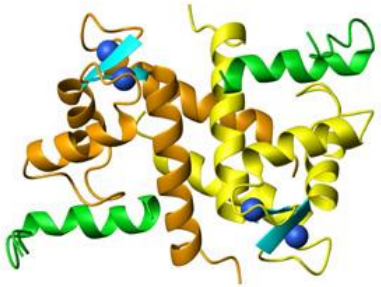
CHIP-chip



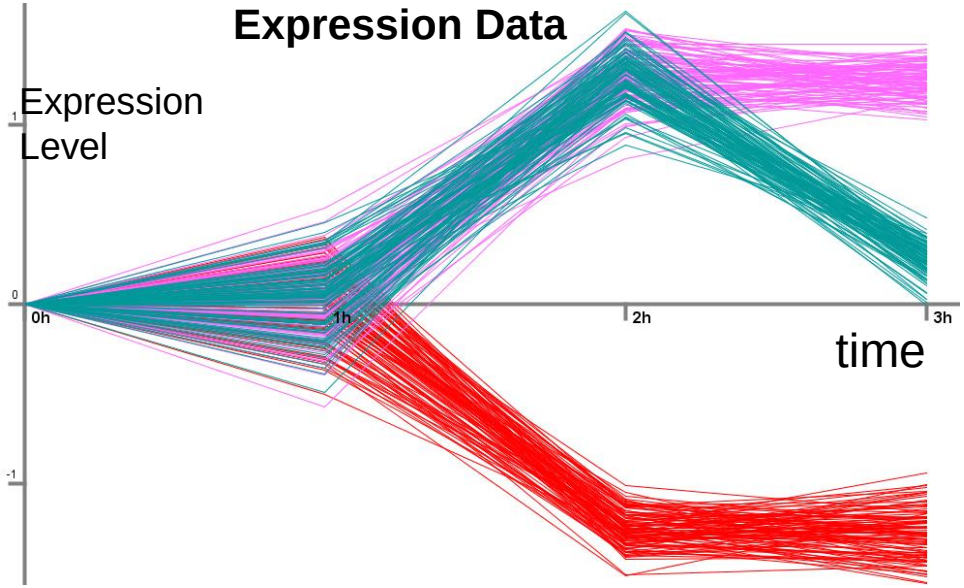
microarray



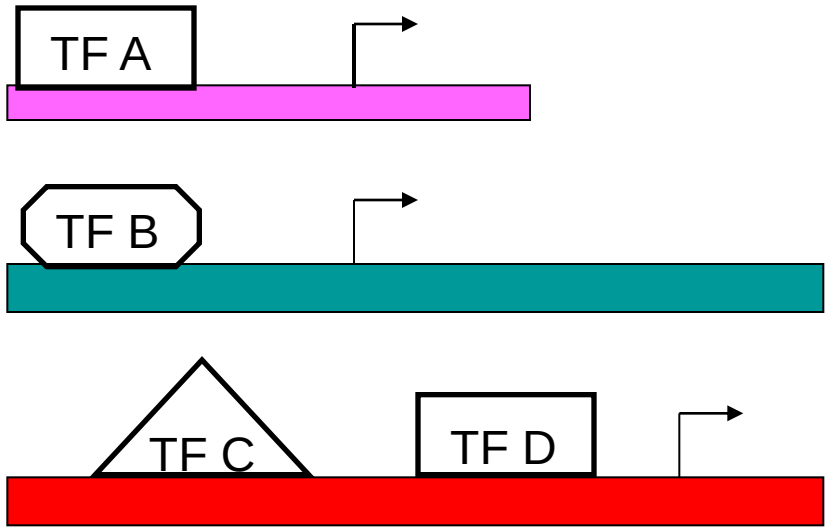
PPI



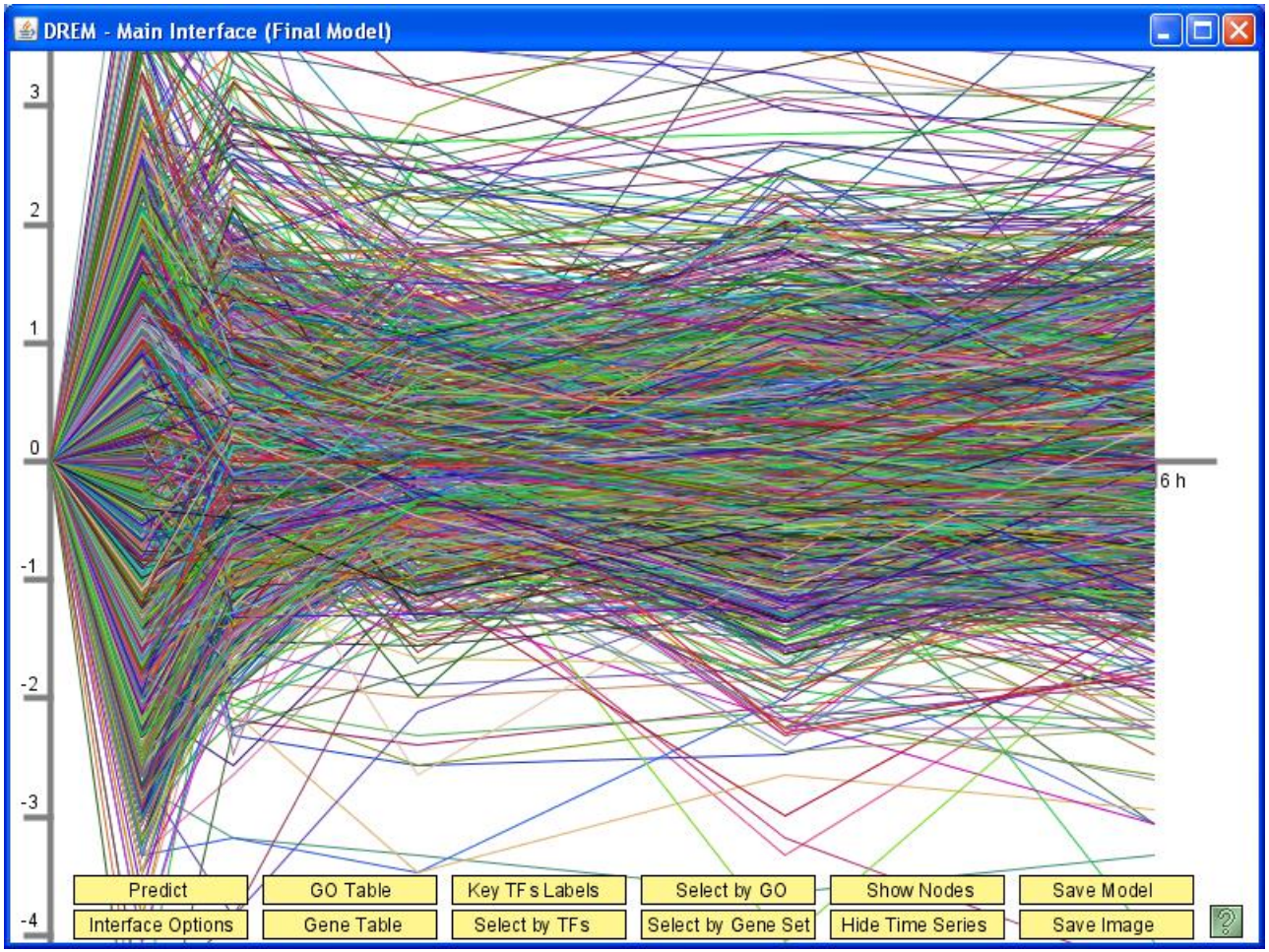
Expression Data



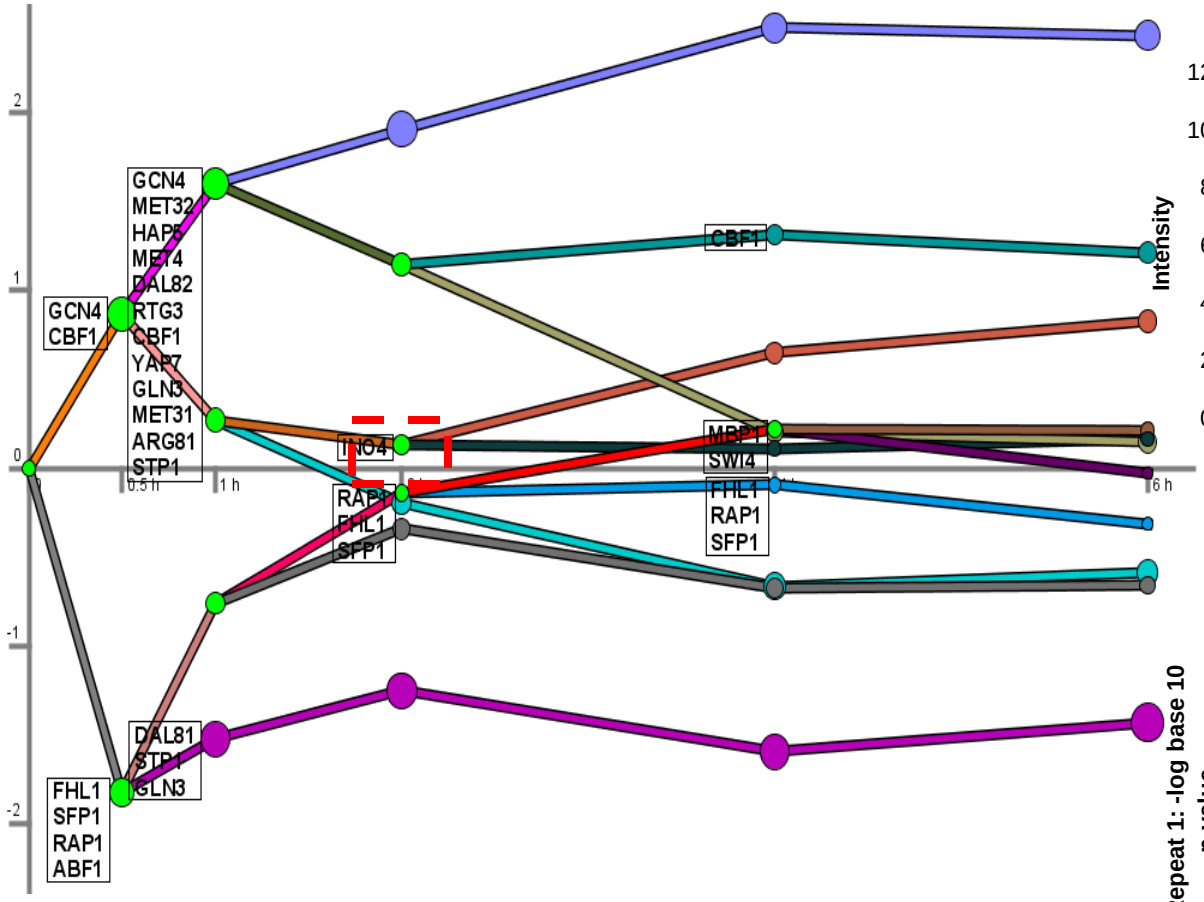
Static TF-DNA Binding Data

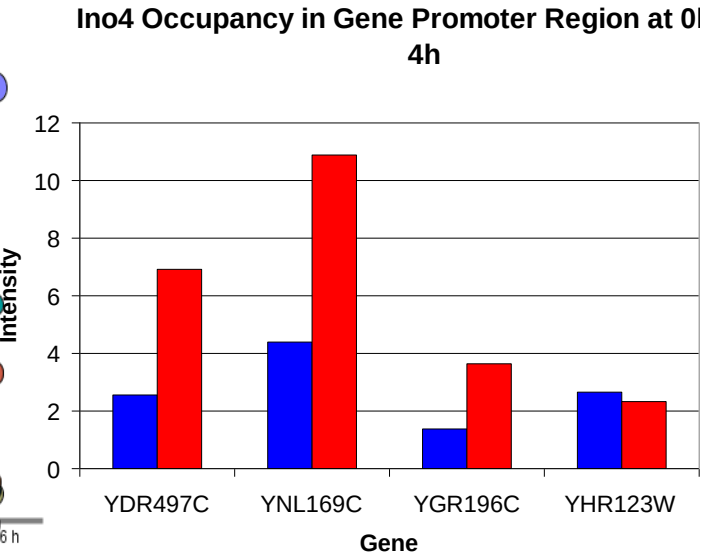


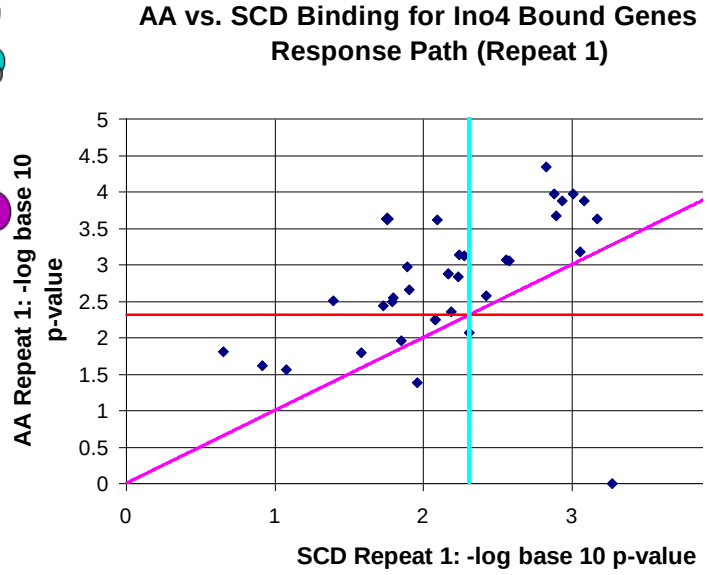
Things are a bit more complicated: Real data



Final model and validations

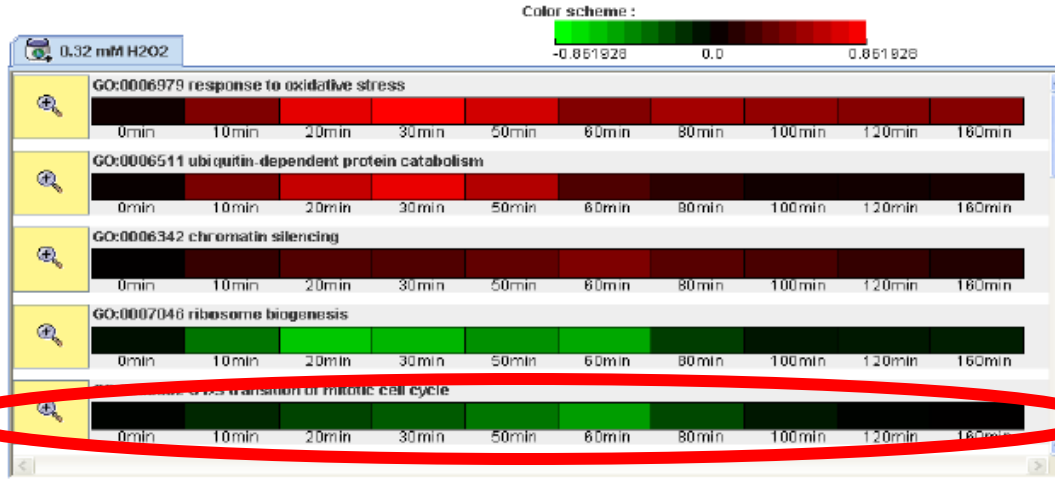






Continuous hidden process models

CHPM



G1/S transition of mitotic cell cycle