

# Graphical Models: Graphs, Boolean, and Bayesian Networks

COMP 572 (BIOS 572 / BIOE 564) - Fall 2011  
Luay Nakhleh, Rice University

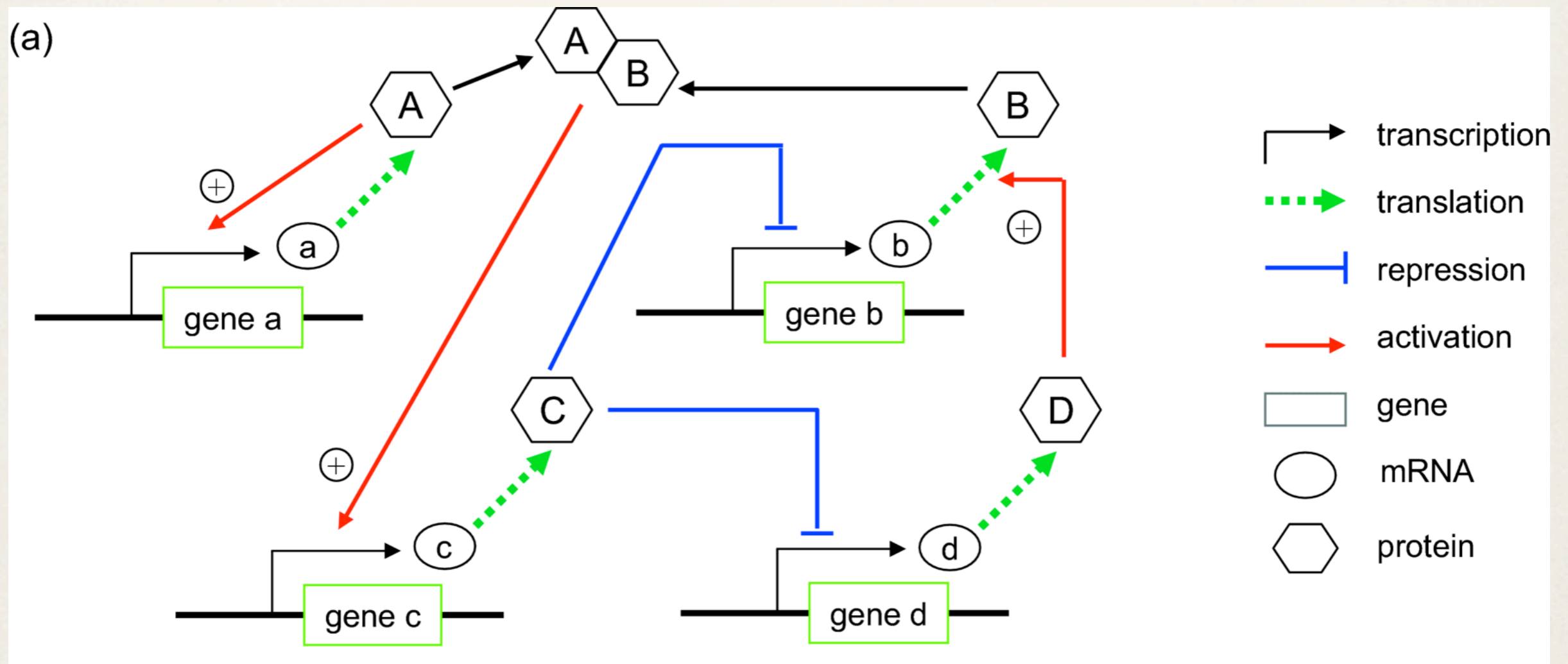
---

# Gene Regulatory Networks

---

- ❖ We'll illustrate some of the graphical models using gene regulatory networks (GRNs).
- ❖ *Gene regulatory networks* describe the molecules involved in gene regulation, as well as their interactions.
- ❖ *Transcription factors* are stimulated by upstream signaling cascades and bind on *cis-regulatory* positions of their target genes.
- ❖ Bound transcription factors *promote* or *inhibit* RNA polymerase assembly and thus determine whether and to what extent the target gene is *expressed*.

# Gene Regulatory Networks



# Outline

---

- ❖ Graph representation
- ❖ Boolean networks
- ❖ Bayesian networks

# Graph Representation

---

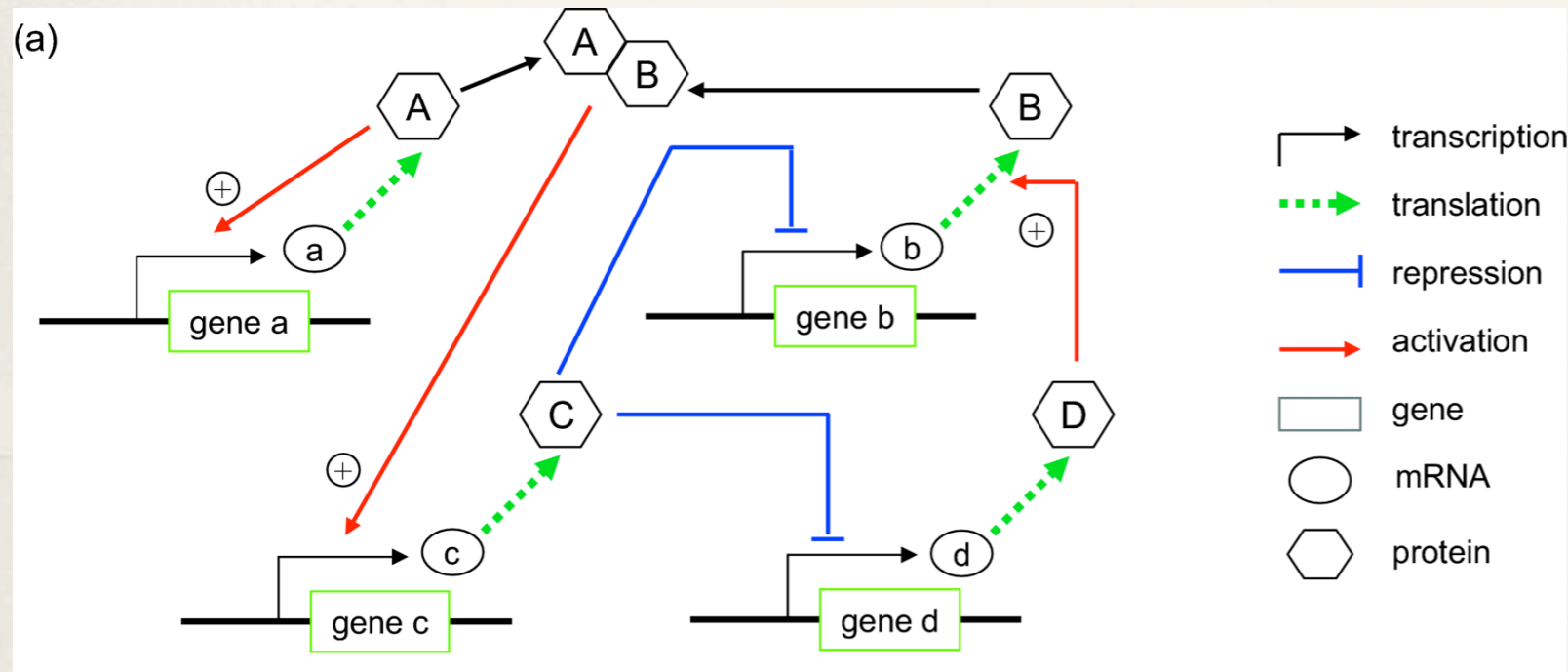
- ❖ A directed graph  $G=(V,E)$  is a tuple where  $V$  denotes a set of vertices (or nodes) and  $E$  a set of edges.
- ❖ An edge  $(i,j)$  in  $E$  indicates that  $i$  regulates the expression of  $j$ .
- ❖ Edges can have information about interactions. For example,  $(i,j,+)$  for “ $i$  activates  $j$ ” and  $(i,j,-)$  for “ $i$  inhibits  $j$ ”.
- ❖ Annotated directed graphs are the most commonly available type of data for regulatory networks.

# Graph Representation

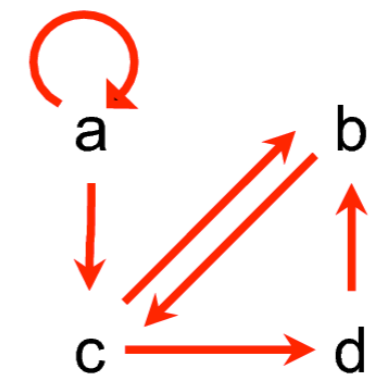
---

- ❖ Directed graphs do not suffice to describe the dynamics of a network, but they may contain information that allows certain predictions about network properties:
  - ❖ Tracing paths between genes yields sequences of regulatory events, shows redundancy in the regulation, or indicates missing regulatory interactions (that are, for example, known from experiments).
  - ❖ A cycle may indicate feedback regulation.
  - ❖ Comparison of GRNs of different organisms may reveal evolutionary relations and targets for bioengineering and pharmaceutical applications.
  - ❖ The network complexity can be measured by the connectivity.

# Graph Representation



## Directed graphs



$$V = \{a, b, c, d\}$$

$$E = \{(a, a, +), (a, c, +), (b, c, +), (c, b, -), (c, d, -), (d, b, +)\}$$

# Boolean Networks

---

- ❖ Boolean networks are *qualitative* descriptions of gene regulatory interactions
- ❖ Gene expression has two states: on (1) and off (0)
- ❖ Let  $x$  be an  $n$ -dimensional binary vector representing the state of a system of  $n$  genes
- ❖ Thus, the state space of the system consists of  $2^n$  possible states

# Boolean Networks

---

- ❖ Each component,  $x_i$ , determines the expression of the  $i^{\text{th}}$  gene
- ❖ With each gene  $i$  we associate a Boolean rule,  $b_i$
- ❖ Given the input variables for gene  $i$  at time  $t$ , this function determines whether the regulated element is active (1) or inactive (0) at time  $t+1$ , i.e.,

$$x_i(t + 1) = b_i(x(t)), \quad 1 \leq i \leq n$$

# Boolean Networks

---

- ❖ The practical feasibility of Boolean networks is heavily dependent on the number of input variables,  $k$ , for each gene
- ❖ The number of possible input states of  $k$  inputs is  $2^k$
- ❖ For each such combination, a specific Boolean function must determine whether the next state would be on or off
- ❖ Thus, there are  $2^{2^k}$  possible Boolean functions (or rules)
- ❖ This number rapidly increases with the connectivity

# Boolean Networks

---

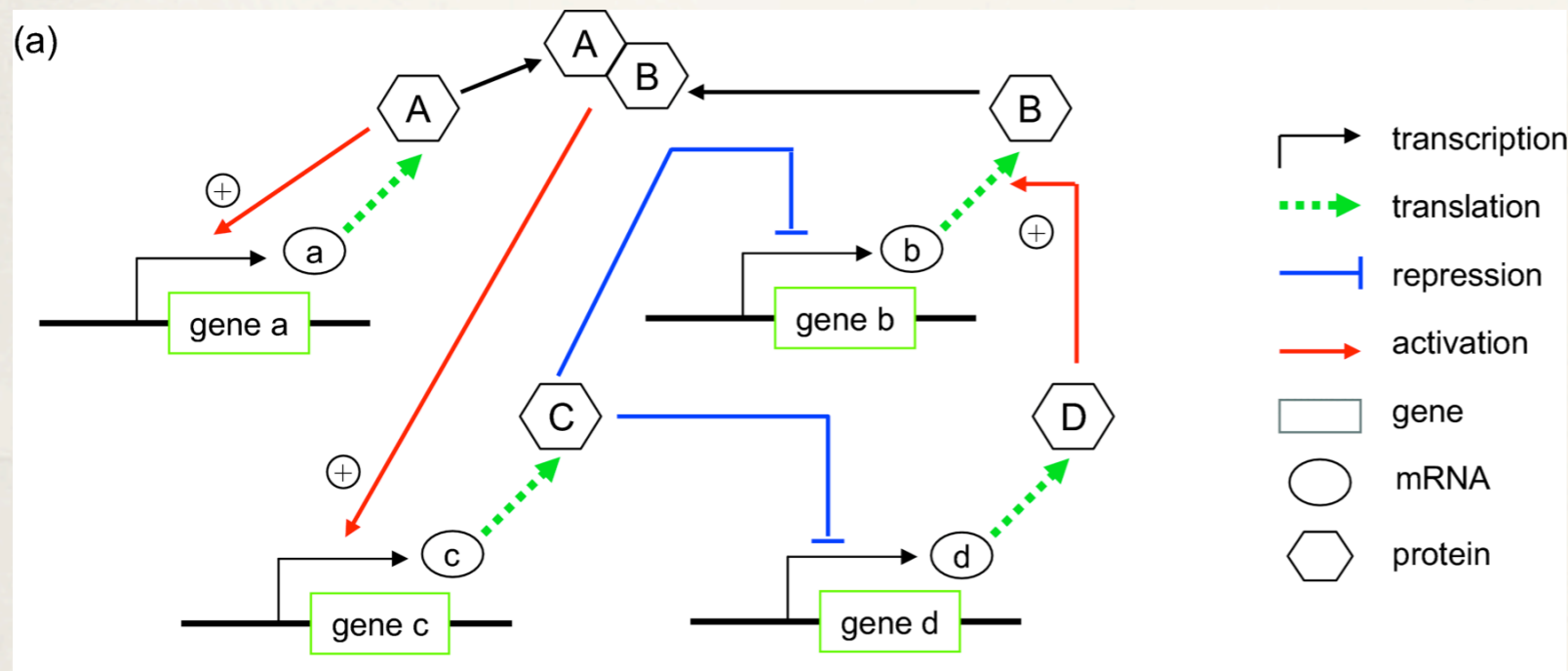
- ❖ In a Boolean network each state has a deterministic output state
- ❖ A series of states is called a *trajectory*
- ❖ If no difference occurs between the transitions of two states, i.e., output state equals input state, then the system is in a *point attractor*
- ❖ Point attractors are analogous to *steady states*
- ❖ If the system is in a cycle of states, then we have a *dynamic attractor*

# Boolean Networks

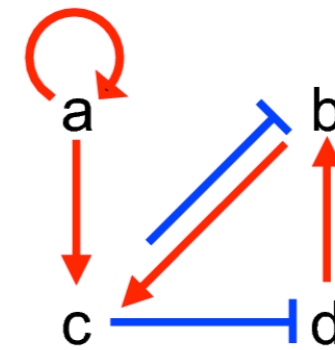
---

- ❖ Since the number of states in the state space is finite, the number of possible transitions is also finite.
- ❖ Therefore, each trajectory will lead either to a steady state or to a state cycle. These state sequences are called *attractors*.
- ❖ *Transient states* are those states that do not belong to an attractor.
- ❖ All states that lead to the same attractor constitute its *basin of attraction*.

# Boolean Networks



## Boolean network



$$a(t+1) = a(t)$$

$$b(t+1) = (\text{not } c(t)) \text{ and } d(t)$$

$$c(t+1) = a(t) \text{ and } b(t)$$

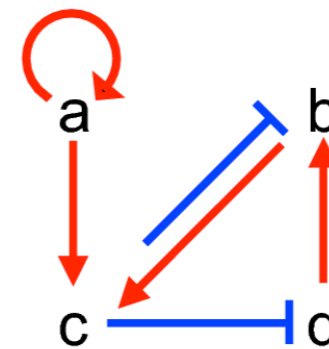
$$d(t+1) = \text{not } c(t)$$

# Boolean Networks

---

- ❖ The temporal behavior is determined by the sequence of states  $(a,b,c,d)$  given in an initial state.
- ❖ What happens if the initial state of  $a$  is 0? If the initial state of  $a$  is 1?

Boolean network



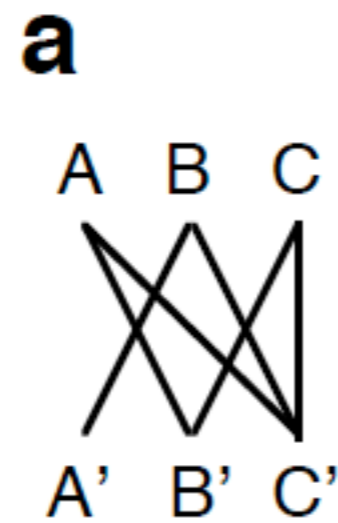
$$a(t+1) = a(t)$$

$$b(t+1) = (\text{not } c(t)) \text{ and } d(t)$$

$$c(t+1) = a(t) \text{ and } b(t)$$

$$d(t+1) = \text{not } c(t)$$

# Boolean Networks



**b**

$A' = B$   
 $B' = A \text{ or } C$   
 $C' = (A \text{ and } B) \text{ or } (B \text{ and } C) \text{ or } (A \text{ and } C)$

**c**

input			output		
A	B	C	A'	B'	C'
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	1	1	1
1	1	1	1	1	1

Fig. 1 A simple Boolean network. a) Wiring diagram. b) Logical (Boolean) rules. c) Complete state transition table defining network. The input column corresponds to the state at time= $t$ , the output column (elements marked by prime) corresponds to the state at time= $t+1$ .

# Boolean Networks: The REVEAL Algorithm

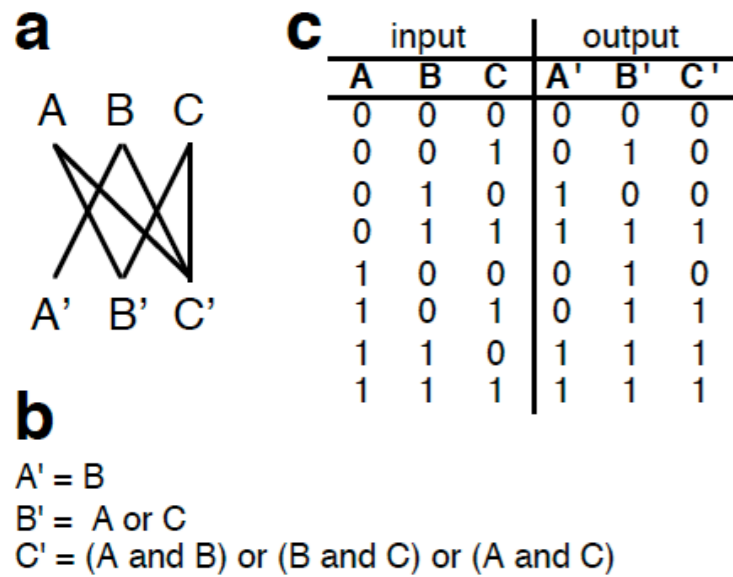


Fig. 1 A simple Boolean network. a) Wiring diagram. b) Logical (Boolean) rules. c) Complete state transition table defining network. The input column corresponds to the state at time=t, the output column (elements marked by prime) corresponds to the state at time=t+1.

“REVEAL, A general reverse engineering algorithm for inference of genetic network architectures”

Liang et al., PSB 1998

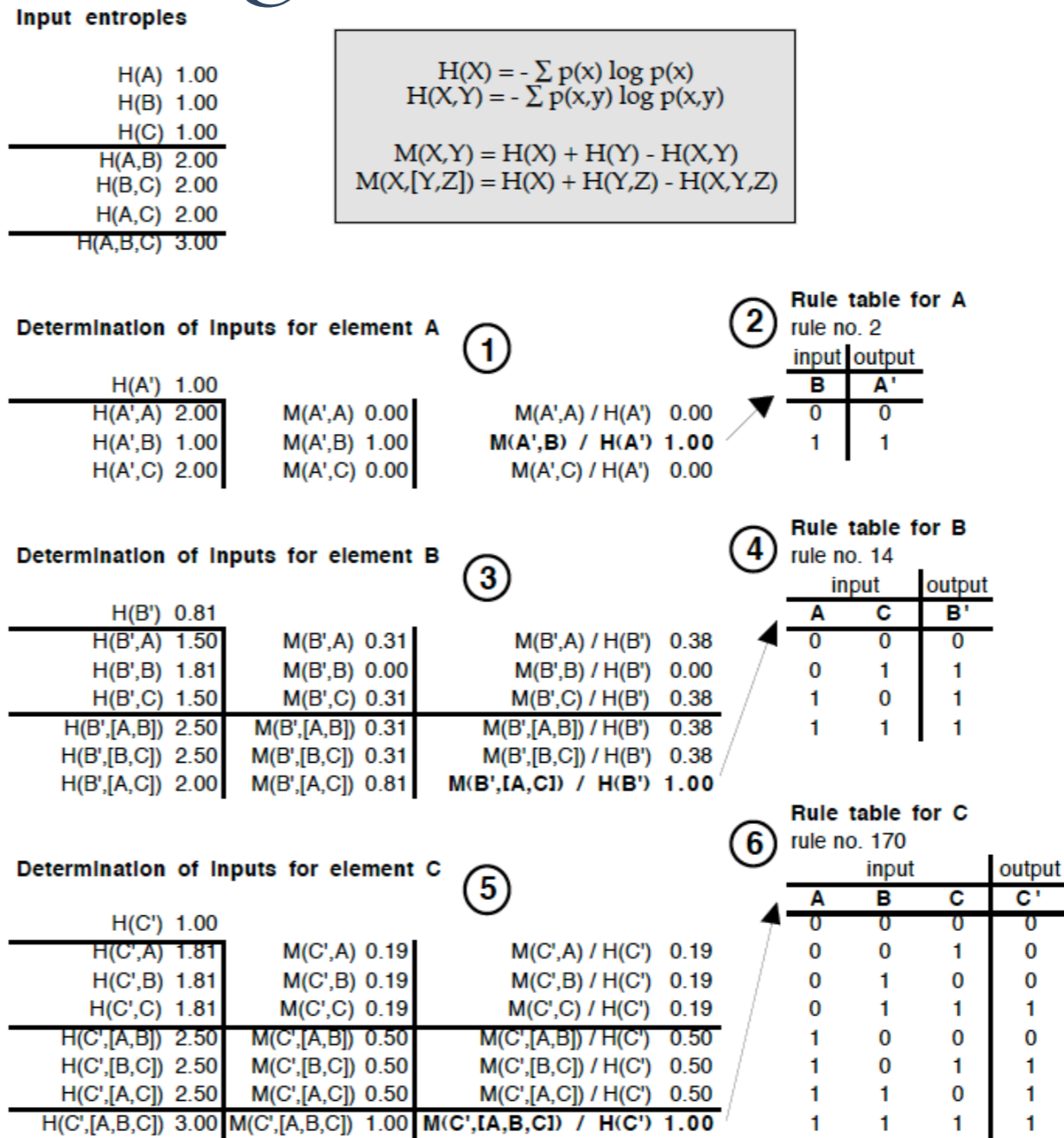


Fig. 5 Outline of progressive M-analysis underlying REVEAL for network example shown in Fig. 1. Hs and Ms are calculated from the look-up tables according to the definitions (shaded). The network wiring is extracted by M-analysis (left, odd steps). Rule tables are then determined directly from the trajectory (right, even steps).

# Bayesian Networks

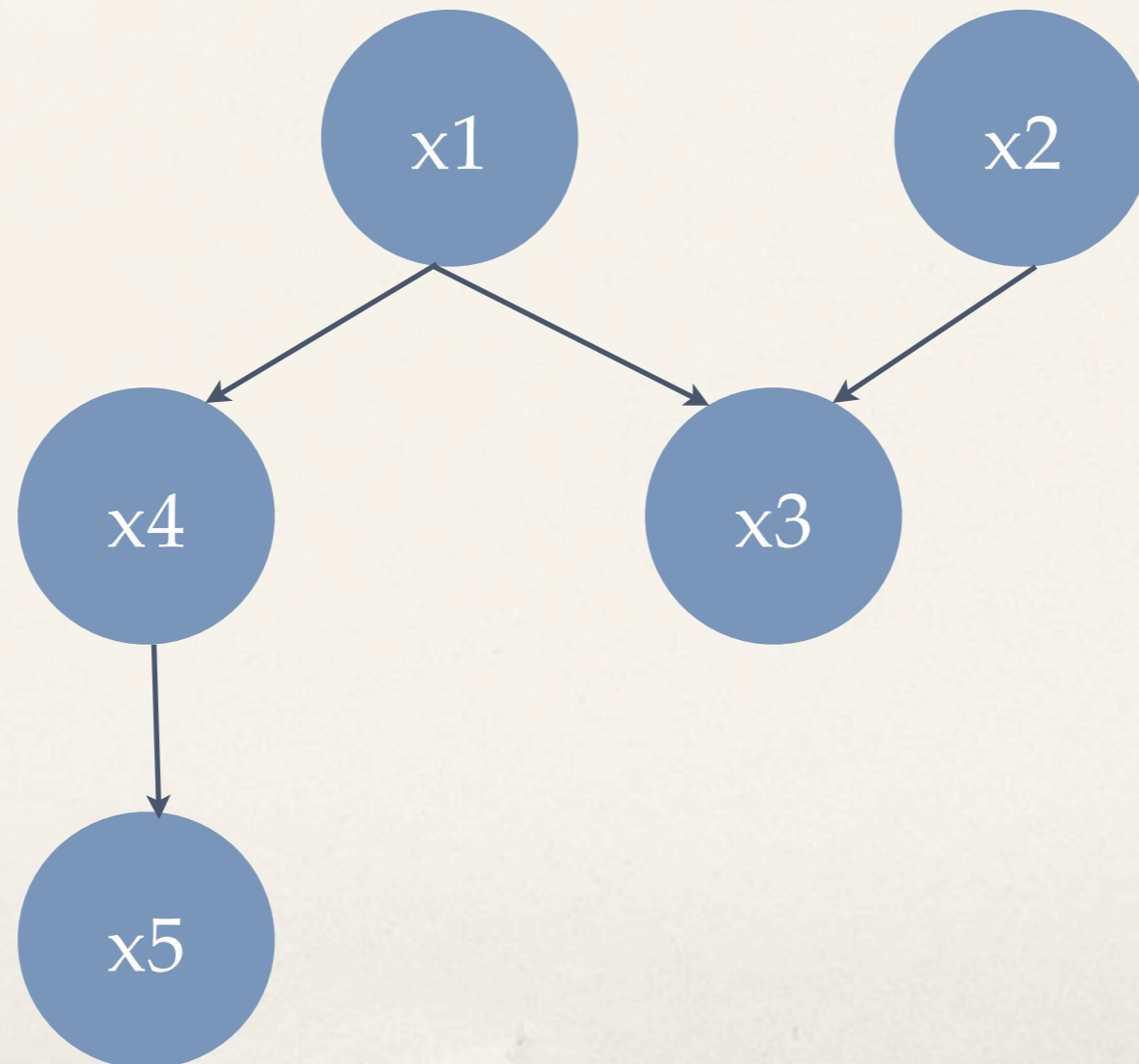
---

- ❖ Bayesian networks are probabilistic descriptions of the regulatory network.
- ❖ A Bayesian network consists of (1) a directed, acyclic graph,  $G=(V,E)$ , and (2) a set of probability distributions.
- ❖ The  $n$  vertices ( $n$  genes) correspond to random variables  $x_i$ ,  $1 \leq i \leq n$ .
- ❖ For example, the random variables describe the gene expression level of the respective gene.

# Bayesian Networks

---

- ❖ For each  $x_i$ , a conditional probability  $p(x_i | L(x_i))$  is defined, where  $L(x_i)$  denotes the parents of gene  $i$ , i.e., the set of genes that have a direct regulatory influence on gene  $i$ .



# Bayesian Networks

---

- ❖ The set of random variables is completely determined by the joint probability distribution.
- ❖ Under the Markov assumption, i.e., the assumption that each  $x_i$  is conditionally independent of its non-descendants given its parents, this joint probability distribution can be determined by the factorization via

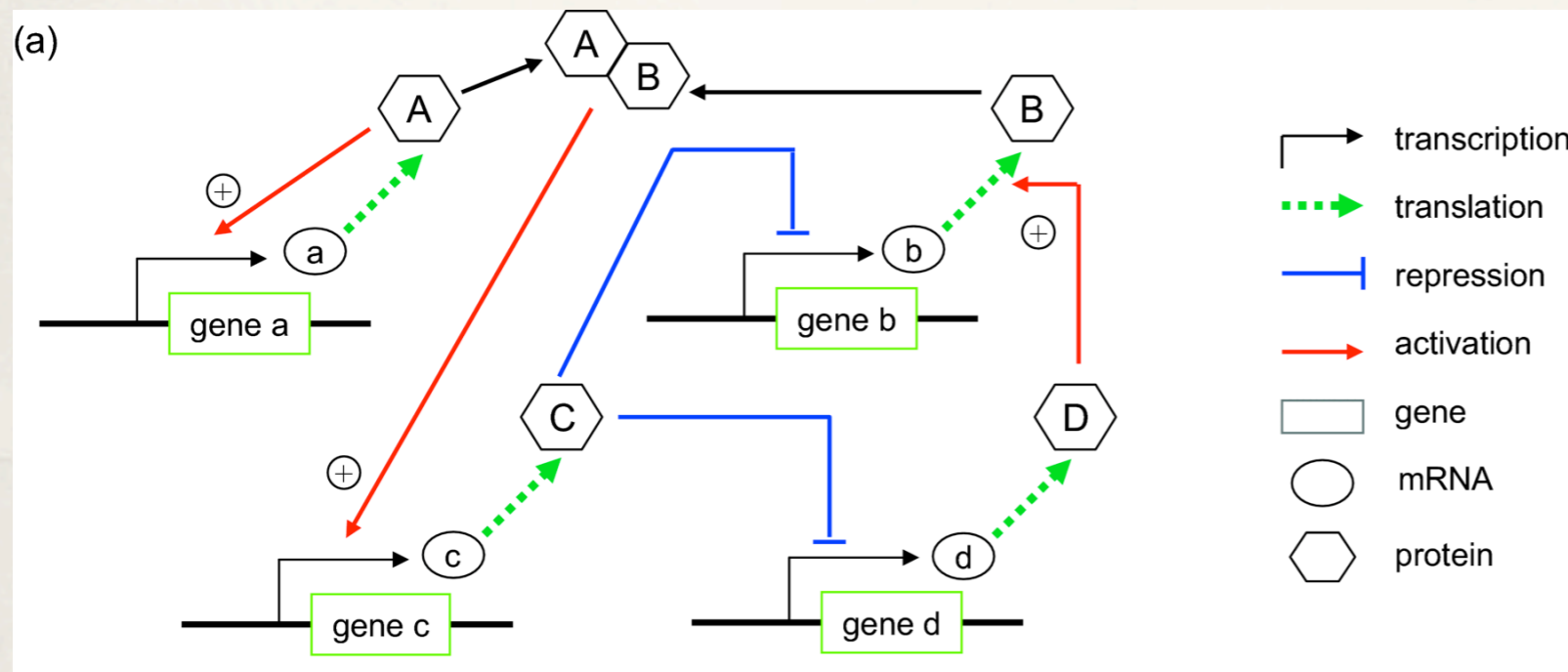
$$p(x) = \prod_{i=1}^n p(x_i | L(x_i))$$

# Bayesian Networks

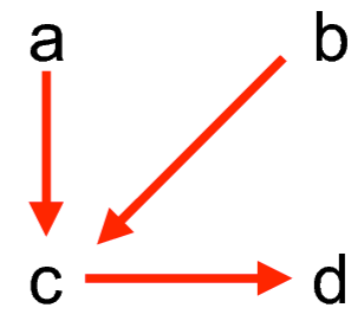
---

- ❖ Conditional independence of two random variables  $x_i$  and  $x_j$  given a random variable  $x_k$  means that  $p(x_i, x_j | x_k) = p(x_i | x_k)p(x_j | x_k)$ , or, equivalently,  $p(x_i | x_j, x_k) = p(x_i | x_k)$ .
- ❖ The conditional distributions  $p(x_i | L(x_i))$  are typically assumed to be linearly normally distributed, i.e.,  $p(x_i | L(x_i)) \sim N\left(\sum_k a_k x_k, \sigma^2\right)$ , where  $x_k$  is in the parent set of  $x_i$ .

# Bayesian Networks



## Bayesian network



$$p(x_a)$$

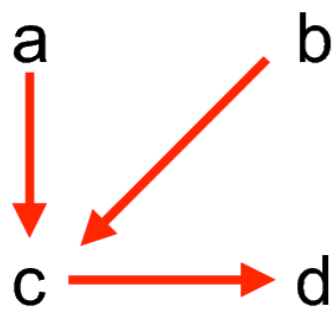
$$p(x_b)$$

$$p(x_c | x_a, x_b),$$

$$p(x_d | x_c),$$

# Bayesian Networks

Bayesian network



$p(x_a)$   
 $p(x_b)$   
 $p(x_c|x_a, x_b)$ ,  
 $p(x_d|x_c)$ ,

a	b	P(c=1)
0	0	0.02
0	1	0.08
1	0	0.06
1	1	0.88

c	P(d=1)
0	0.03
1	0.92

**Inputs:** a,b

**Outputs:** d

**Hidden:** c

# Bayesian Networks

---

- ❖ Given a network structure and a conditional probability table (CPT) for each node, we can calculate the output of the system by simply looking up the relevant input condition (row) in the CPT of the inputs, generating a “1” with the output probability specified for that condition, then using these newly generated node values to evaluate the outputs of nodes that receive inputs from these, and so on.
- ❖ We can also go backwards, asking what input activity patterns could be responsible for a particular observed output activity pattern.

# Bayesian Networks

---

- ❖ To construct a Bayesian network, we need to estimate two sets of parameters:
  - ❖ the connectivity pattern (dependencies between variables), and
  - ❖ the values of the CPT entries.
- ❖ The usual approach to learning both sets of parameters simultaneously is to first search for network structures, and evaluate the performance of each candidate network structure after estimating its optimum conditional probability values.

# Bayesian Networks

---

- ❖ *Learning conditional probabilities from full data*
  - ❖ If we have full data, i.e., for every combination of inputs to every node we have several measurements of node output value, then we can estimate the node output probabilities by simply counting the proportion of outputs at each level (e.g., on, off). These can be translated to CPTs, which together with the network structure fully define the Bayesian network.

# Bayesian Networks

---

- ❖ *Learning conditional probabilities from incomplete data*
  - ❖ If we do not have data for all possible combinations of inputs to every node, or when some individual data values are missing, we start by giving all missing CPT values equal probabilities. Next, we use an optimization algorithm (Expectation Maximization, Markov Chain Monte Carlo search, etc.) to curve-fit the missing numbers to the available data. When we find parameters that improve the network's overall performance, we can replace the previous "guess" with the new values and repeat the process.

# Bayesian Networks

---

- ❖ *Learning conditional probabilities from incomplete data*
  - ❖ Another case of incomplete data pertains to hidden nodes: no data is available for certain nodes in the network.
  - ❖ A solution is to iterate over plausible network structures, and to use a “goodness” score to identify the Bayesian network.

# Bayesian Networks

---

- ❖ Given a training set  $T$  of independent realizations of the  $n$  random variables  $x^1, \dots, x^n$ , the problem is to find a Bayesian network that best matches  $T$
- ❖ A common solution is to assign a score to each calculated network using the *a posteriori* probability of the calculated network,  $N$ , given the training data by

$$\log P(N|T) = \log \frac{P(T|N)P(N)}{P(T)} = \log P(T|N) + \log P(N) + \text{const}$$

where the constant is independent of the calculated network and

$$P(T|N) = \int P(T|N, \Theta)P(\Theta|N)d\Theta$$

is the marginal likelihood that averages the probability of the data over all possible parameter assignments to the network.

# Dynamic Bayesian Networks

---

- ❖ Feedback loops in a graphical representation of dependencies between different quantities correspond to networks that are not Bayesian networks, according to the definition above.
- ❖ To specify the behavior of networks with feedback, we need to introduce an explicit concept of time and “unfold” the edges in a new time direction.

