Machine Learning of Bayesian Networks

Peter van Beek
University of Waterloo

Collaborators

- Hella-Franziska Hoffmann, PhD student
- Colin Lee, NSERC USRA
- Andrew Li, NSERC USRA
- Alister Liao, PhD student
- Charupriya Sharma, PhD student

Outline

- Introduction
  - Machine learning
  - Bayesian networks
- Machine learning a Bayesian network
  - exact learning algorithms
  - approximate learning algorithms
- Extensions
  - generate all of the best networks
  - incorporate expert domain knowledge
- Conclusions

Outline

- Introduction
    - Machine learning
    - Bayesian networks
- Machine learning a Bayesian network
    - exact learning algorithms
    - approximate learning algorithms
- Extensions
    - generate all of the best networks
    - incorporate expert domain knowledge
- Conclusions

Machine learning: Supervised learning

- Training data $\mathcal{D}$, with N examples (instances):

Sex	Exercise	Age	Diastolic BP	…	Diabetes
male	no	middle-aged	high	…	yes
female	yes	elderly	normal	…	no
…	…	…	…	…	…

- *Supervised learning*: learn mapping from inputs **x** to outputs y, given a labeled set of input-output pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$, $i = 1, \ldots, N$
  - prediction
  - *here*: probabilistic models of the form P(y | **x**)
    - P(Diabetes = yes | Exercise = yes, Age = young)
    - P(Diabetes = no | Exercise = yes, Age = young)

Machine learning: Unsupervised learning

- Training data $\mathcal{D}$, with N examples (instances):

Sex	Exercise	Age	Diastolic BP	…	Diabetes
male	no	middle-aged	high	…	yes
female	yes	elderly	normal	…	no
…	…	…	…	…	…

- *Unsupervised learning*: learn hidden structure from unlabeled data $\mathcal{D} = \{(\mathbf{x}_i)\}$, $i = 1, \ldots, N$
  - knowledge discovery
  - density estimation (estimate underlying probability density function)
  - *here*: probabilistic models of the form P($\mathbf{x}$)
    - answer any probabilistic query; e.g., P(Exercise = yes | Diastolic BP = high)
    - representations that are useful for P($\mathbf{x}$) tend to be useful when learning P(y | $\mathbf{x}$)

Supervised vs unsupervised learning

- Supervised: Probabilistic models of the form P(y | $\mathbf{x}$)
  - discriminative models
    - model dependence of unobserved target variable y on observed variables $\mathbf{x}$
  - performance measure: predictive accuracy, cross-validation
- Unsupervised: Probabilistic models of the form P($\mathbf{x}$)
  - generative models
    - model probability distribution over all variables
  - performance measure: “fit” to the data

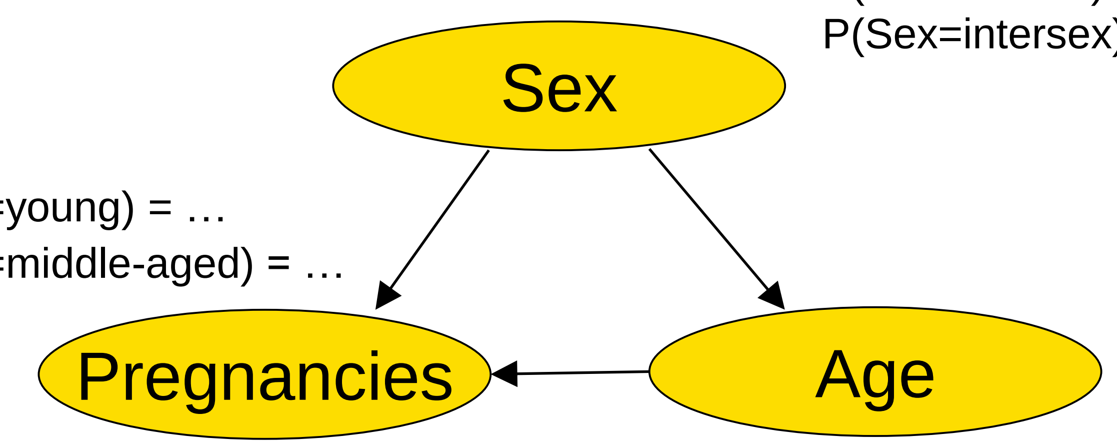
Bayesian networks

- A Bayesian network is a directed acyclic graph (DAG) where:
  - nodes are variables

  - directed arcs connect pairs of nodes, indicating direct influence, high correlation
  - each node has a conditional probability table specifying the effects parents have on the node

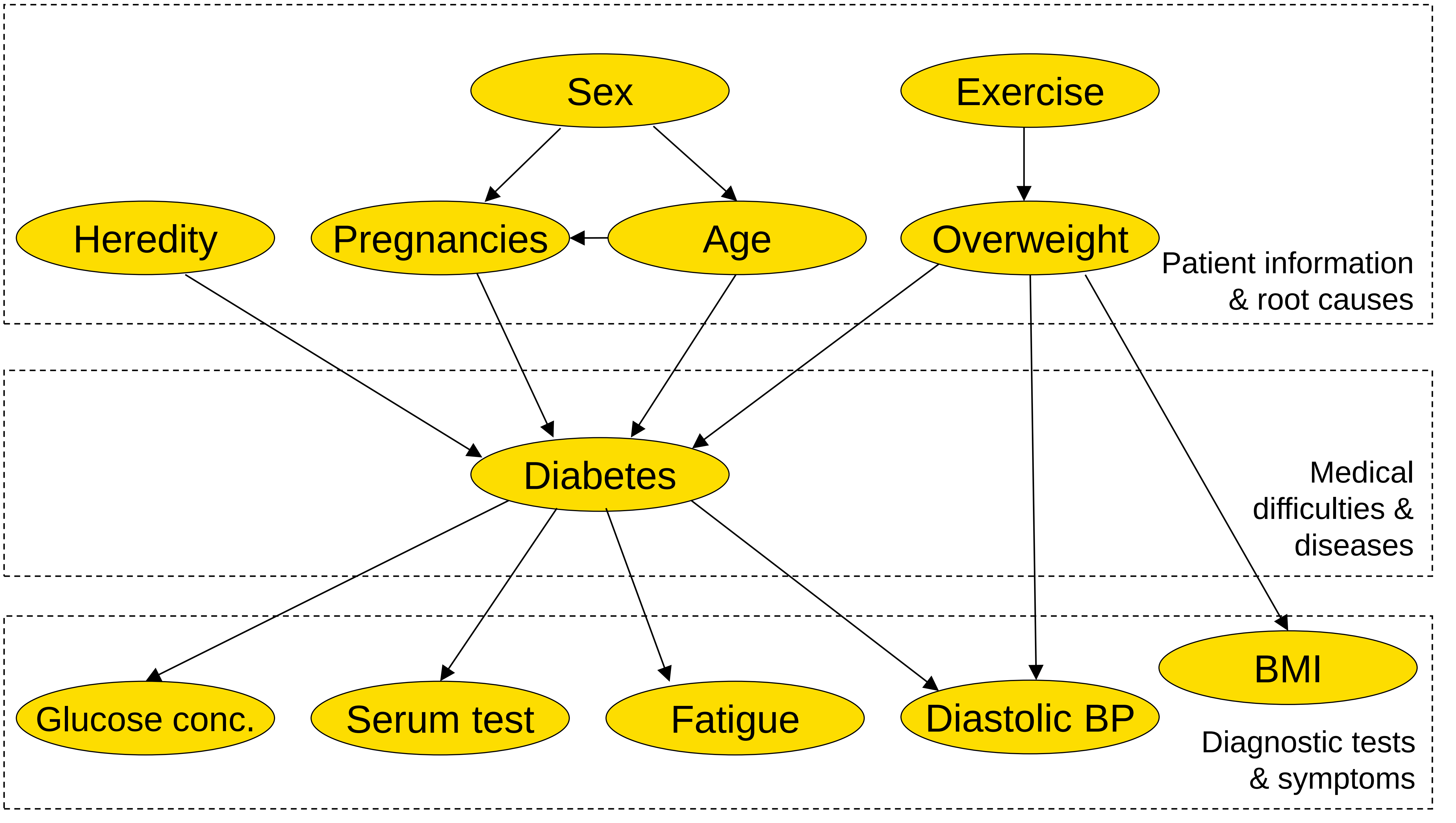
P(Sex=male) = 0.493
P(Sex=female) = 0.490
P(Sex=intersex) = 0.017

P(Preg=0 | Sex=male, Age=young) = …
P(Preg=0 | Sex=male, Age=middle-aged) = …
…



P(Age=young | Sex=male) = **…**
P(Age=middle-aged | Sex=male) = **…**
P(Age=elderly | Sex=male) = **…**
P(Age=young | Sex=female) = **…**
P(Age=middle-aged | Sex=female) = **…**
…

Example: Medical diagnosis of diabetes



Real-world examples

- Conflict analysis for groundwater protection (Giordano et al., 2013)
  - Bayesian network for farmers' behavior with regard to groundwater management
  - Analyze impact of policy on behavior and degree of conflict
- Safety risk assessment for construction projects (Leu & Chang, 2013)
  - Bayesian networks for four primary accident types
  - Site safety management and analyze causes of accidents
- Climate change adaption policies (Catenacci and Giupponi, 2009)
  - Bayesian network for ecological modelling, natural resource management, climate change policy
  - Analyze impact of climate change policies

Semantics of Bayesian networks (I)

- Training data $\mathcal{D}$, with N examples (instances):

Sex	Exercise	Age	Diastolic BP	…	Diabetes
male	no	middle-aged	high	…	yes
female	yes	elderly	normal	…	no
…	…	…	…	…	…

- Representation of joint probability distribution
  - *Atomic event*: assignment of a value to each variable in the model
  - *Joint probability distribution*: assignment of a probability to each possible atomic event
  - Bayesian network is a succinct representation of the joint probability distribution

$$P(x_1, \ldots, x_n) = \prod P(x_i \mid \text{Parents}(x_i))$$

  - Can answer any and all probabilistic queries

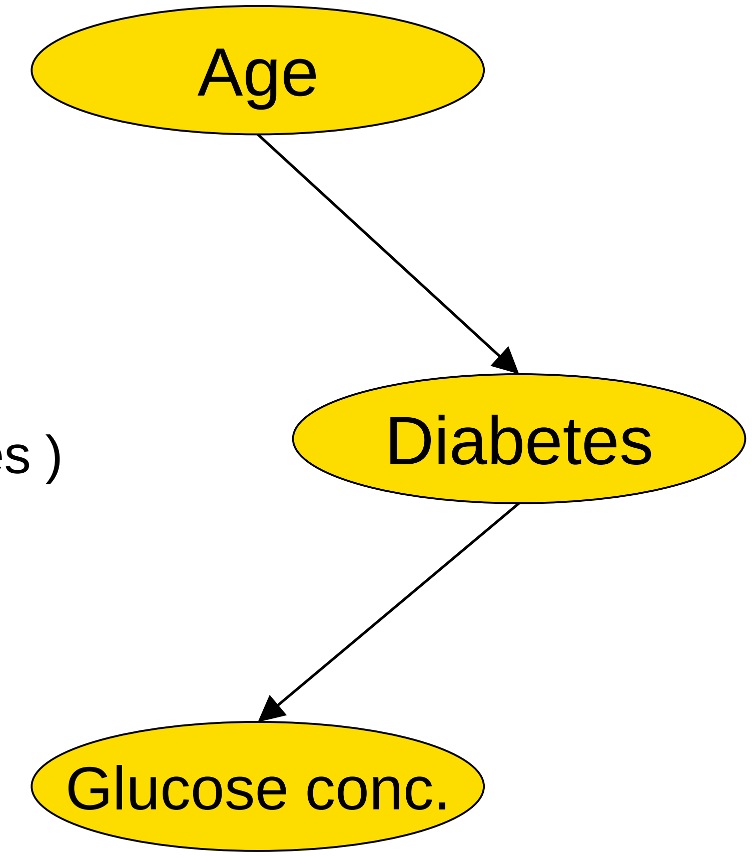
Semantics of Bayesian networks (II)

- Encoding of conditional independence assumptions
  - Conditional independence

    x is conditionally independent of *y* given *z* if

    $P(x \mid y, z) = P(x \mid z)$

  - "Missing" arcs represent conditional independence assumptions
    - E.g., P(Glucose | Age, Diabetes) = P(Glucose | Diabetes)



Advantages of Bayesian networks

- Declarative representation
  - separation of knowledge and reasoning
  - principled representation of uncertainty
- Interpretable
  - clear semantics, facilitate understanding a domain
  - explanation
- Learnable from data
  - can combine learning from data with prior expert knowledge
- Easily combinable with decision analytic tools
  - decision networks, value of information, utility theory

Outline

- Introduction
  - Machine learning
  - Bayesian networks
- Machine learning a Bayesian network
  - exact learning algorithms
  - approximate learning algorithms
- Extensions
  - generate all of the best networks
  - incorporate expert domain knowledge
- Conclusions

Structure learning from data: *measure fit to data*

- Training data $\mathcal{D}$, with N examples (instances):

Sex	Exercise	Age	Diastolic BP	…	Diabetes
male	no	middle-aged	high	…	yes
female	yes	elderly	normal	…	no
…	…	…	…	…	…

- First attempt: Maximize probability of observing data, given model G:
  - $P(\mathcal{D} \mid G)$
  - overfitting: complete network
- Scoring function: Add penalty term for complexity of model
  - Score(G) = likelihood + (penalty for complexity)
  - e.g., $\text{BIC}(G) = -\log_2 P(\mathcal{D} \mid G) + ½ (\log_2 N) \cdot \| G \|$
  - as N grows, more emphasis given to fit to data

Structure learning from data: *decomposability*

- Problem: Find a directed acyclic graph (DAG) *G* which minimizes:

$$\mathrm{Score}(G)$$

- *Decomposability*:

$$\mathrm{Score}(G) = \sum_{i=1}^{n} \mathrm{Score}(\text{ Parents}(x_i)\text{ })$$

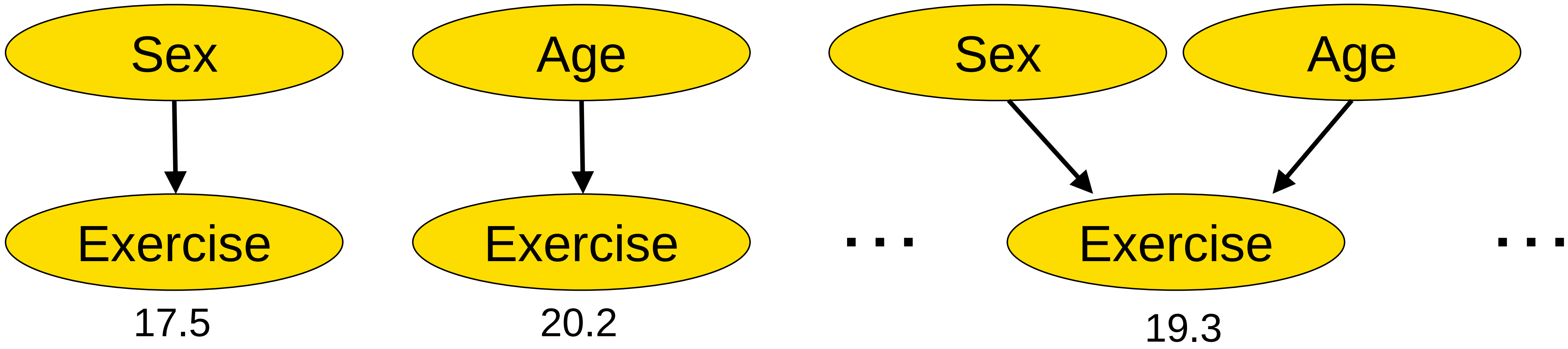
- Rephrased problem: Choose parent set for each variable so that Score(*G*) is minimized and resulting graph is acyclic

Structure learning from data:
score-and-search approach

1. Training data $\mathcal{D}$, with *N* examples (instances):

Sex	Exercise	Age	Diastolic BP	…	Diabetes
male	no	middle-aged	high	…	yes
female	yes	elderly	normal	…	no
…	…	…	…	…	…

2. Scoring function (BIC/MDL, BDeu) gives possible parent sets:



3. Combinatorial optimization problem:
   - find a directed acyclic graph (DAG) over the variables that minimizes the total score

Outline

- Introduction
  - Machine learning
  - Bayesian networks
- Machine learning a Bayesian network
  - exact learning algorithms
  - approximate learning algorithms
- Extensions
  - generate all of the best networks
  - incorporate expert domain knowledge
- Conclusions

Exact learning: Global search algorithms

Dynamic programming	Koivisto & Sood, 2004 Silander & Myllymäki, 2006 Malone, Yuan & Hansen, 2011
Integer linear programming	Jaakkola et al., 2010 Bartlett & Cussens, 2013, 2017 (GOBNILP)
A* search	Yuan & Malone, 2013 Fan, Malone & Yuan, 2014 Fan & Yuan, 2015
Breadth-first branch-and-bound search	Suzuki, 1996 Campos & Ji, 2011 Fan, Malone & Yuan, 2014, 2015
Depth-first branch-and-bound search	Tian, 2000 Malone & Yuan, 2014 van Beek & Hoffman, 2015 (CPBayes)

Constraint programming

- A constraint model is defined by:
  - a set of variables $\{x_1, \ldots, x_n\}$
  - a set of values for each variable $dom(x_1), \ldots, dom(x_n)$
  - a set of constraints $\{C_1, \ldots, C_m\}$
- A solution to a constraint model is a complete assignment to all the variables that satisfies the constraints

Global constraints

- A *global constraint* is a constraint that can be specified over an arbitrary number of variables
- Advantages:
  - captures common constraint patterns
  - efficient, special purpose constraint propagation algorithms can be designed

Example global constraint: alldifferent

- Consists of:
  - set of variables $\{x_1, \ldots, x_n\}$
- Satisfied iff:
  - each of the variables is assigned a different value
- Constraint propagation:
  - suppose alldifferent(x_1, x_2, x_3) where:
    - $dom(x_1) = \{b, c, d, e\}$
    - $dom(x_2) = \{b, d\}$
    - $dom(x_3) = \{b, d\}$

Bayesian network structure learning: Constraint model (I)

- Notation:

V	set of variables
n	number of variables in data set
cost(v)	cost (score) of variable *v*
dom(v)	domain of variable *v*

- Vertex (possible parent set) variables: $v_1, \ldots, v_n$
  - $dom(v_i) \subseteq 2^V$ consists of possible parent sets for v_i
  - assignment $v_i = p$ denotes vertex v_i has parents p in the graph
  - global constraint: acyclic($v_1, \ldots, v_n$)
    - satisfied iff the graph designated by the parent sets is acyclic

Bayesian network structure learning: Constraint model (II)

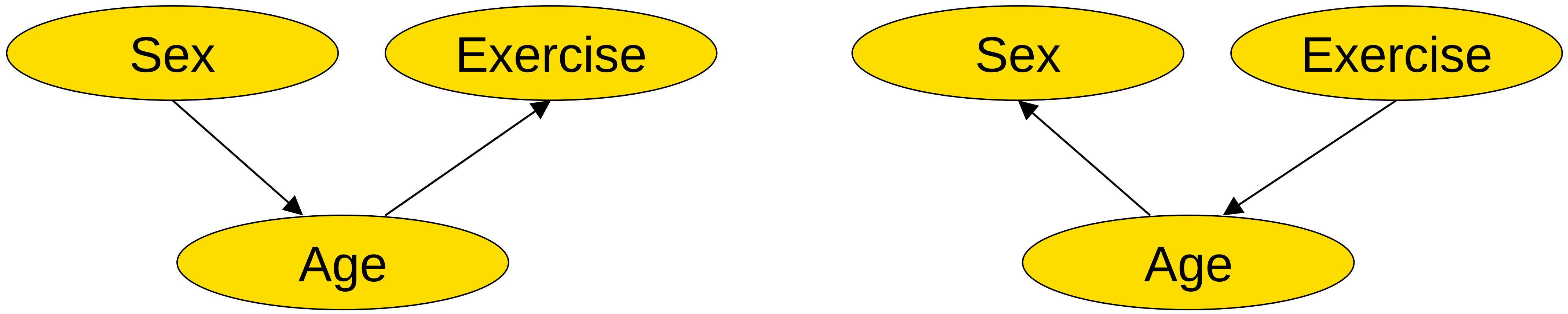
- Ordering (permutation) variables: $o_1, \ldots, o_n$
  - $dom(o_i) = \{1, \ldots, n\}$
  - assignment $o_i = j$ denotes vertex v_j is in position i in the total ordering
  - global constraint: alldifferent($o_1, \ldots, o_n$)
  - given a permutation, it is easy to determine the minimum cost DAG
- Depth auxiliary variables: $d_1, \ldots, d_n$
  - $dom(d_i) = \{0, \ldots, n-1\}$
  - assignment $d_i = k$ denotes that depth of vertex variable v_j that occurs at position i in the ordering is k
- Channeling constraints connect the three types of variables

Bayesian network structure learning: Improving the constraint model

- Add constraints to increase constraint propagation (e.g., Smith 2006)
  - *symmetry-breaking constraints*: preserve one among a set of symmetric solutions
  - *dominance constraints*: preserve an optimal solution

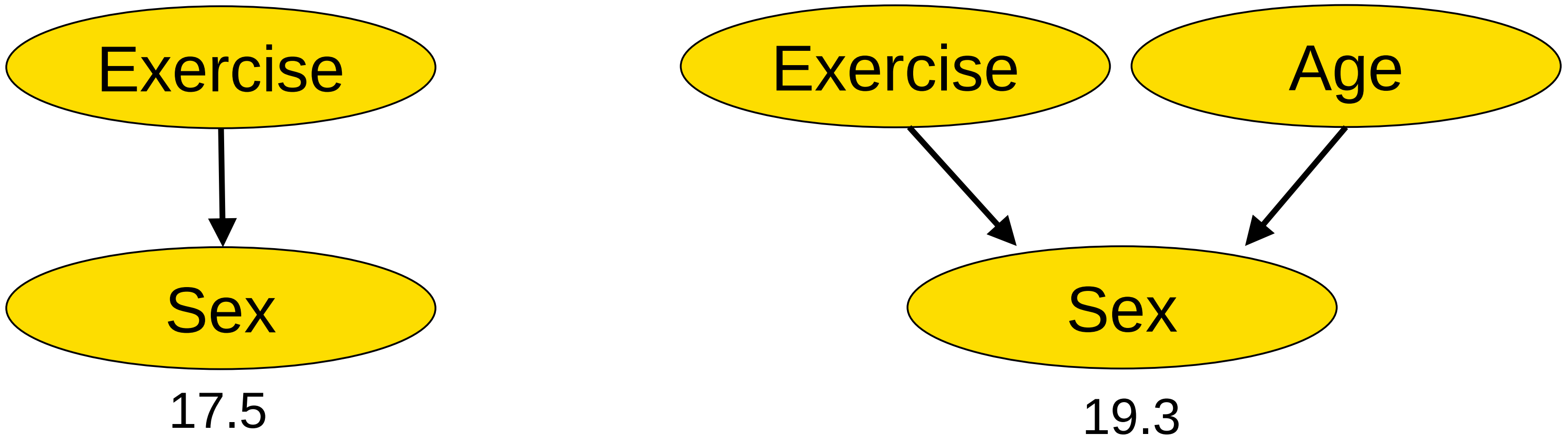
Example: Symmetry-breaking constraints

- I-equivalent networks:
  - two DAGs are said to be I-equivalent if they encode the same set of conditional independence assumptions
- Chickering (1995, 2002) provides a local characterization:
  - sequence of “covered” edges that can be reversed
- Example:



Example: Dominance constraints

- Teyssier and Koller (2005) present a cost-based pruning rule
  - only applicable before search begins
  - routinely used in score-and-search approaches



- We generalize the pruning rule
  - applicable during search
  - takes into account ordering information induced by the partial solution so far

Constraint-based search variant (CPBayes)

- Constraint-based depth-first branch-and-bound search
  - branching over ordering (permutation) variables $o_1, \ldots, o_n$
  - cost function $z = \text{cost}(v_1) + \ldots + \text{cost}(v_n)$
  - lower bound based on Fan and Yuan (2015) using pattern databases
  - initial upper bound based on Lee and van Beek (2017) using local search

Outline

- Introduction
  - Machine learning
  - Bayesian networks
- Machine learning a Bayesian network
  - exact learning algorithms
  - approximate learning algorithms
- Extensions
  - generate all of the best networks
  - incorporate expert domain knowledge
- Conclusions

Approximate learning: Local search algorithms

Genetic algorithm	Larrañaga et al., 1996
Greedy search	Chickering et al., 1997
Tabu search	Teyssier & Koller, 2005
Ant colony optimization	De Campos et al., 2002
Memetic search	Lee and van Beek, 2017
Space of network structures	Cooper and Herskovits, 1992 Chickering et al., 1997
Space of equivalent network structures	Chickering, 2002
Space of variable orderings, permutations	Larrañaga et al., 1996 Teyssier & Koller, 2005 (OBS) Scanagatta et al., 2015 (ASOBS) Lee and van Beek, 2017 (MINOBS)

Permutation-based local search

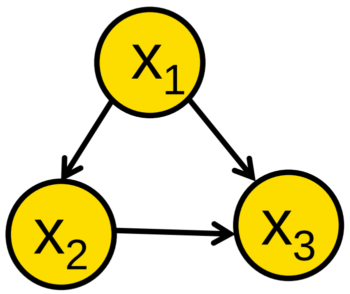
- Local search over the space of permutations of vertices
  - best score for a permutation is easily found
  - find permutation that gives best score overall
- Example: Ordering O = x_1, x_2, x_3

Candidate parent sets:

x_1:	4, $\{x_2, x_3\}$	12, {}
x_2:	5, $\{x_1\}$	10, {}
x_3:	3, $\{x_1, x_2\}$	4, {}

Optimal parent set for x_1 : {}	Score: 12
Optimal parent set for x_2 : $\{x_1\}$	Score: 5
Optimal parent set for x_3 : $\{x_1, x_2\}$	Score: 3

Score of network (Score(O)): 12 + 5 + 3 = 20



Greedy search over orderings

```
O ← randomOrdering();
curScore ← sc(O);
while neighbours(O) contains an ordering which improves curScore do
    O ← selectImprovingNeighbour(O);
    curScore ← sc(O);
return O
```

- Basic local search algorithm
- Output is a local minima in the search space
- Need to design functions neighbours(O), selectImprovingNeighbour(O)

Neighborhoods

Consider a permutation representation

<1, 2, 3, 4, 5, 6, 7, 8>

What could be its neighbors?

Transpose: swap two adjacent
e.g., <1, 2, $\underline{4}$, $\underline{3}$, 5, 6, 7, 8> is a neighbor $O(n)$

Swap: swap two (not necessarily adjacent)
e.g., <1, $\underline{6}$, 3, 4, 5, $\underline{2}$, 7, 8> is a neighbor $O(n^2)$

Insert: move
e.g., <1, $\underline{5}$, 2, 3, 4, 6, 7, 8> is a neighbor $O(n^2)$

Block insert: move a subsequence of queens
e.g., <1, $\underline{4}$, $\underline{5}$, 2, 3, 6, 7, 8> is a neighbor $O(n^3)$

Memetic search variant (MINOBS)

- Population-based approach with local improvement procedures
- At start of algorithm, create a population of locally optimal orderings
- For each iteration:
  - Add new local optima to the population by crossing/perturbing members of the population and applying local search
  - Prune members of the population so that it returns to the original size
- Parameters tuned from small training set using ParamILS

Experimental evaluation

- Algorithms evaluated in our study:
  - GOBNILP, version 1.6.2 (Bartlett and Cussens 2013; Bartlett et al., 2017)
    - global search, based on integer linear programming
  - CPBayes, version 1.2 (van Beek and Hoffman, 2015)
    - global search, based on constraint programming
  - ASOBS, version of December 2016 (Scanagatta et al., 2015)
    - local search, based on space of variable orderings, swap neighborhood, and improved search of neighborhood
  - MINOBS, version 0.2 (Lee and van Beek, 2017)
    - local search, based on space of variable orderings, insertion neighborhood, and memetic or population based approach
    - report median of 10 runs with different random seeds

Experimental setup

- Instances:

size	*variables*	*scoring functions*	*remarks*
small	$n \leq 20$	BIC BDeu	• data sets obtained from J. Cussens, B. Malone, UCI ML Repository • local scores computed from data sets using code from B. Malone • larger BDeu instances have indegree restricted to be between 6 and 8
medium	$20 < n \leq 60$		
large	$60 < n \leq 100$	BIC	• data sets and local scores obtained from Bayesian Network Learning and Inference Package (BLIP) by M. Scanagatta • maximum indegree of parents sets restricted to be 6
very large	$100 < n \leq 1000$		
massive	$n > 1000$		

Experimental results

- Notation:

n	number of variables in data set
N	number of instances in data set
d	total number of possible parent sets for variables
—	indicates method did not report any solution within given time bound
opt	indicates method found the known optimal solution within given time bound
*benchmark**	indicates optimal value for benchmark is not known; in such cases percentage from optimal is calculated using best value found within 24 hours of CPU time

Experimental results

Percentage from optimal, BIC scoring function, small networks ($n \leq 20$ variables)

				1 minute			5 minutes			10 minutes		
benchmark	*n*	*N*	*d*	GO	CP	MI	GO	CP	MI	GO	CP	MI
nltcs	16	3,236	7,933	0.2%	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
msnbc	17	58,265	47,229	—	*opt*	*opt*	0.4%	*opt*	*opt*	0.0%	*opt*	*opt*
letter	17	20,000	4,443	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
voting	17	435	1,848	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
zoo	17	101	554	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
tumour	18	339	219	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
lympho	19	148	143	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
vehicle	19	846	763	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
hepatitis	20	155	266	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
segment	20	2,310	1,053	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*

Experimental results

Percentage from optimal, BDeu scoring function,
small networks ($n \leq 20$ variables)

				1 minute			5 minutes			10 minutes		
benchmark	*n*	*N*	*d*	GO	CP	MI	GO	CP	MI	GO	CP	MI
nltcs	16	3,236	8,091	0.0%	*opt*	*opt*	0.0%	*opt*	*opt*	*opt*	*opt*	*opt*
msnbc	17	58,265	50,921	—	*opt*	*opt*	0.2%	*opt*	*opt*	0.1%	*opt*	*opt*
letter	17	20,000	18,841	1.3%	*opt*	*opt*	0.1%	*opt*	*opt*	0.0%	*opt*	*opt*
voting	17	435	1,940	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
zoo	17	101	2,855	1.7%	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
tumour	18	339	274	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
lympho	19	148	345	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
vehicle	19	846	3,121	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
hepatitis	20	155	501	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
segment	20	2,310	6,491	0.3%	*opt*	*opt*	0.3%	*opt*	*opt*	0.0%	*opt*	*opt*

Experimental results

Discussion of results for BIC and BDeu scoring functions, small networks ($n \leq 20$ variables)

- CPBayes and MINOBS are able to consistently find optimal solutions within a 1 minute time bound, whereas GOBNILP sometimes has not yet found an optimal solution with a 10 minute time bound

- CPBayes and GOBNILP, being global search methods, may terminate earlier than time bound, whereas MINOBS and ASOBS, being local search methods, terminate only when a time bound is reached

- The parameter *d* is a relatively good predictor for instances that GOBNILP finds difficult; it strongly correlates with size of integer programming model

Experimental results

Percentage from optimal, BIC scoring function, medium networks (20 < $n \leq 60$ variables)

				1 minute			5 minutes			10 minutes		
benchmark	*n*	*N*	*d*	GO	CP	MI	GO	CP	MI	GO	CP	MI
mushroom	23	8,124	13,025	1.1%	*opt*	*opt*	0.6%	*opt*	*opt*	0.6%	*opt*	*opt*
autos	26	159	2,391	1.5%	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
insurance	27	1,000	506	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
horse colic	28	300	490	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
steel	28	1,941	93,026	—	0.0%	0.0%	0.9%	*opt*	*opt*	0.7%	*opt*	*opt*
flag	29	194	741	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
wdbc	31	569	14,613	0.7%	*opt*	*opt*	0.2%	*opt*	*opt*	0.2%	*opt*	*opt*
water	32	1,000	159	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
mildew	35	1,000	126	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
soybean	36	266	5,926	1.6%	*opt*	*opt*	1.6%	*opt*	*opt*	*opt*	*opt*	*opt*
alarm	37	1,000	1,002	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
bands	39	277	892	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
spectf	45	267	610	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
sponge	45	76	618	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
barley	48	1,000	244	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
hailfinder	56	100	50	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
hailfinder	56	500	43	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
lung cancer	57	32	292	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
carpo	60	100	423	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
carpo	60	500	847	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*

Experimental results

Percentage from optimal, BDeu scoring function,
medium networks (20 < *n* ≤ 60 variables)

				5 minutes			1 hour			12 hours		
benchmark	*n*	*N*	*d*	GO	CP	MI	GO	CP	MI	GO	CP	MI
mushroom	23	8,124	438,185	—	0.0%	0.0%	0.5%	*opt*	0.0%	0.1%	*opt*	*opt*
autos	26	159	25,238	4.3%	0.0%	0.0%	1.2%	*opt*	0.0%	*opt*	*opt*	*opt*
insurance	27	1,000	792	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
horse colic	28	300	490	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
steel	28	1,941	113,118	2.0%	0.0%	*opt*	0.5%	*opt*	*opt*	0.4%	*opt*	*opt*
flag	29	194	1,324	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
wdbc	31	569	13,473	0.6%	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
water	32	1,000	261	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
mildew	35	1,000	166	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
soybean*	36	266	212,425	—	0.1%	0.1%	3.1%	0.1%	0.1%	1.8%	0.0%	0.0%
alarm	37	1,000	2,113	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
bands	39	277	1,165	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
spectf	45	267	316	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
sponge	45	76	10,790	0.4%	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
barley	48	1,000	364	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
hailfinder	56	100	199	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
hailfinder	56	500	447	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*
lung cancer*	57	32	22,338	6.7%	0.3%	0.1%	6.7%	0.0%	0.0%	0.9%	0.0%	0.0%
carpo	60	100	15,408	2.1%	*opt*	*opt*	0.5%	*opt*	*opt*	*opt*	*opt*	*opt*
carpo	60	500	3,324	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*	*opt*

Experimental results

Discussion of results for BIC and BDeu scoring functions, medium networks (20 < *n* ≤ 60 variables)

- BDeu scoring leads to instances that are significantly harder to solve than BIC scoring (max time bound of 12 hours for BDeu vs. 10 minutes for BIC)
- By the shortest time bounds, CPBayes and MINOBS are able to consistently find optimal or near-optimal solutions
- By the largest time bounds, CPBayes and MINOBS found an optimal solution in all cases where it was known, whereas for five of these instances GOBNILP found high-quality solutions but not optimal solutions
- The parameter *d* is once again a relatively good predictor for instances that GOBNILP finds difficult (GOBNILP is able to prove the optimality of larger instances than CPBayes, and thus scales better on the parameter *n*)

Experimental results

Percentage from optimal, BIC scoring function, large networks (60 < n ≤ 100 variables)

				1 hour				12 hours			
benchmark	*n*	*N*	*d*	GO	CP	AS	MI	GO	CP	AS	MI
kdd	64	34,955	152,873	3.4%	*opt*	0.5%	0.0%	3.3%	*opt*	0.5%	*opt*
plants*	69	3,482	520,148	44.5%	0.1%	17.5%	0.0%	33.0%	0.0%	14.8%	0.0%
bnetflix	100	3,000	1,103,968	—	*opt*	3.7%	*opt*	—	*opt*	2.2%	*opt*

Experimental results

Percentage from optimal, BIC scoring function,
very large networks (100 < *n* ≤ 1000 variables)

				1 hour				12 hours			
benchmark	*n*	*N*	*d*	GO	CP	AS	MI	GO	CP	AS	MI
accidents*	111	2,551	1,425,966	—	0.6%	325.6%	0.3%	—	0.0%	155.9%	0.0%
pumsb_star*	163	2,452	1,034,955	320.7%	—	24.0%	0.0%	277.2%	—	18.9%	0.0%
dna*	180	1,186	2,019,003	—	—	7.3%	0.4%	—	—	5.8%	0.0%
kosarek*	190	6,675	1,192,386	—	—	8.4%	0.1%	—	—	8.0%	0.0%
msweb*	294	5,000	1,597,487	—	—	1.5%	0.0%	—	—	1.3%	0.0%
diabetes*	413	5,000	754,563	—	—	0.8%	0.0%	—	—	0.7%	0.0%
pigs*	441	5,000	1,984,359	—	—	16.8%	1.8%	—	—	16.8%	0.1%
book*	500	1,739	2,794,588	—	—	9.9%	0.8%	—	—	9.1%	0.1%
tmovie*	500	591	2,778,556	—	—	36.1%	5.5%	—	—	33.4%	0.2%
link*	724	5,000	3,203,086	—	—	28.4%	0.2%	—	—	17.1%	0.1%
cwebkb*	839	838	3,409,747	—	—	32.4%	2.3%	—	—	25.5%	0.2%
cr52*	889	1,540	3,357,042	—	—	25.9%	2.2%	—	—	23.5%	0.1%
c20ng*	910	3,764	3,046,445	—	—	16.3%	1.0%	—	—	14.6%	0.0%

Experimental results

Percentage from optimal, BIC scoring function,
massive networks (n > 1000 variables)

				1 hour				12 hours			
benchmark	n	N	d	GO	CP	AS	MI	GO	CP	AS	MI
bbc*	1,058	326	3,915,071	—	—	26.0%	4.5%	—	—	24.4%	0.5%
ad*	1,556	487	6,791,926	—	—	15.2%	3.2%	—	—	15.0%	0.5%

Experimental results

Discussion of results for BIC scoring function, large, very large, and massive networks

- Global search solvers GOBNILP and CPBayes are not competitive on these large to massive networks
  - GOBNILP: for all but three instances, memory requirements exceed 30 GB limit
  - CPBayes: able to solve four instances but this is only due to high-quality initial upper bound found by MINOBS (as well, note that CPBayes can only handle instances for $n \leq 128$)
- Local search solvers ASOBS and MINOBS are able to scale to these large to massive networks within reasonable time bounds
  - MINOBS performs exceptionally well, *consistently* finding high-quality solutions within 1 hour and very high-quality solutions within 12 hours (over ten tests, standard deviation less than 0.3 for 12 hour time bound)
  - ASOBS often reports solutions that are quite far from optimal

Outline

- Introduction
  - Bayesian networks (BNs)
  - applications and advantages
- Machine learning a Bayesian network
  - exact learning algorithms
  - approximate learning algorithms
- Extensions
  - generate all of the best networks
  - incorporate expert domain knowledge
- Conclusions

Generate all of the best networks

- Selecting a single Bayesian network may not be best choice
  - there may be many other Bayesian networks with scores that are close to optimal
  - posterior probability of even the best-scoring Bayesian network often close to zero
- Alternative: some form of Bayesian model averaging
- Previous work:
  - generate k-best Bayesian networks for some k
  - disadvantage: how to choose k?
- We are extending CPBayes to generate all Bayesian networks such that,
  - $OPT \leq score(G) \leq \rho \, OPT$
  - advantages: pruning rules can be generalized and applied, scaling, principled way to chose ρ

Incorporate expert domain knowledge

- Bayesian networks are either:
  - fully specified by a domain expert
    - difficult as number of variables grows
  - learned from data
    - not so reliable when data is limited
- Hybrid method:
  - incorporate both expert knowledge (side constraints) and data
- We have extended MINOBS to handle side constraints:
    - existence of an arc
    - absence of an arc
    - ordering constraints: assert *x* comes before *y* in some ordering of the nodes
    - ancestral constraints: there exists a directed path from *x* to *y*

Conclusion

- Unsupervised learning of structure of Bayesian network
  - formulated as a combinatorial optimization problem
- Viewpoint leads to state-of-the-art algorithms
  - CPBayes: exact algorithm based on constraint-based global search
  - MINOBS: approximate algorithm based on local search
- Viewpoint leads to generalization of existing algorithms
  - generate all of the best networks
  - incorporate expert domain knowledge