Use Chapter 3 of K&F as a reference for CSI
Reading for parameter learning: Chapter 12 of K&F

Context-specific independence
Parameter learning: MLE

Graphical Models – 10708

Carlos Guestrin
Carnegie Mellon University

October 5th, 2005

Announcements

- **Homework 2:**
  - **Programming part in groups of 2-3**
- **Recitation on Thursday 5-6, Wean 4615A**
  - **Also covers details of programming part of HW2 & Matlab**
- **Class project**
  - **Teams of 2-3 students**
  - **Ideas on the class webpage, but you can do your own**
- **Timeline:**
  - **10/19: 1 page project proposal**
  - **11/14: 5 page progress report (20% of project grade)**
  - **12/2: poster session (20% of project grade)**
  - **12/5: 8 page paper (60% of project grade)**
  - **All write-ups in NIPS format (see class webpage)**

Clique trees versus VE

- Clique tree advantages
  - Multi-query settings
  - Incremental updates
  - Pre-computation makes complexity explicit

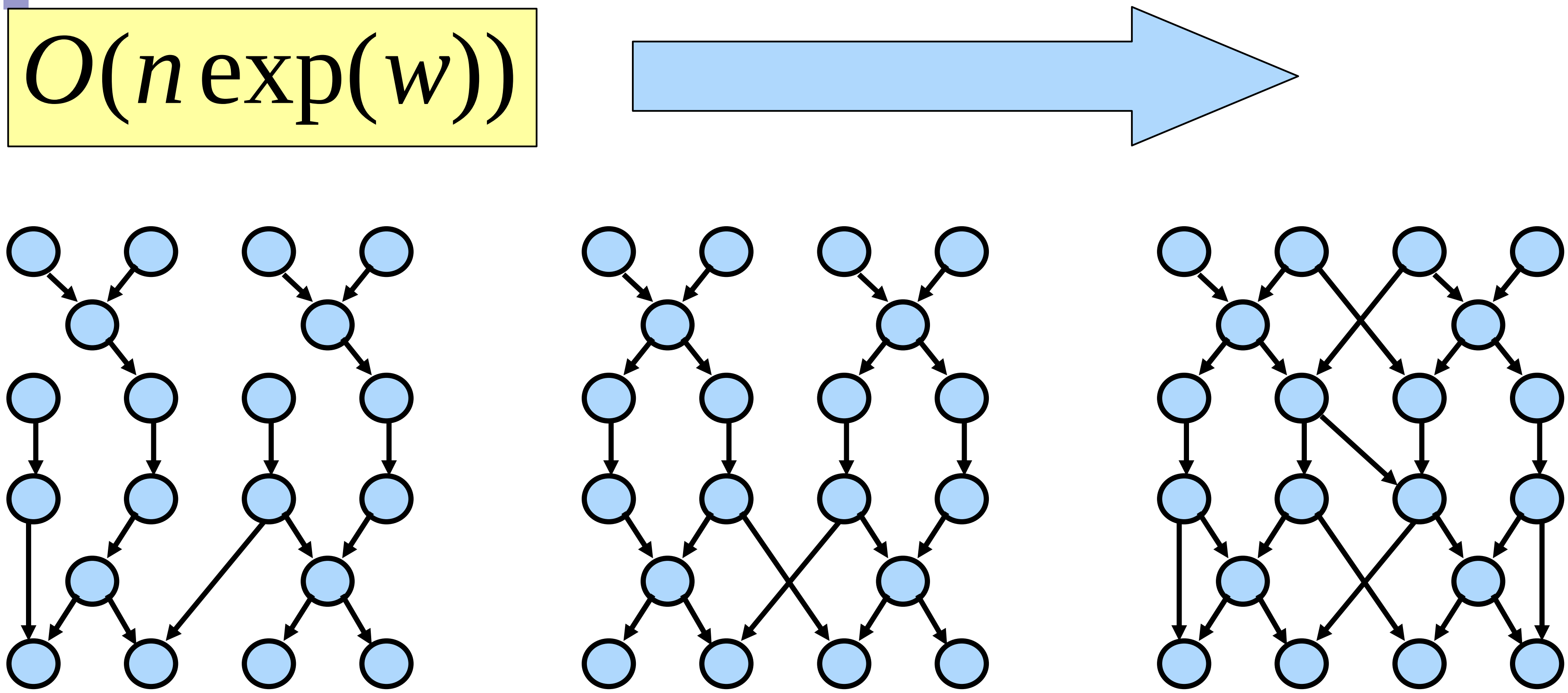
- Clique tree disadvantages
  - Space requirements – no factors are “deleted”
  - Slower for single query
  - Local structure in factors may be lost when they are multiplied together into initial clique potential

Clique tree summary

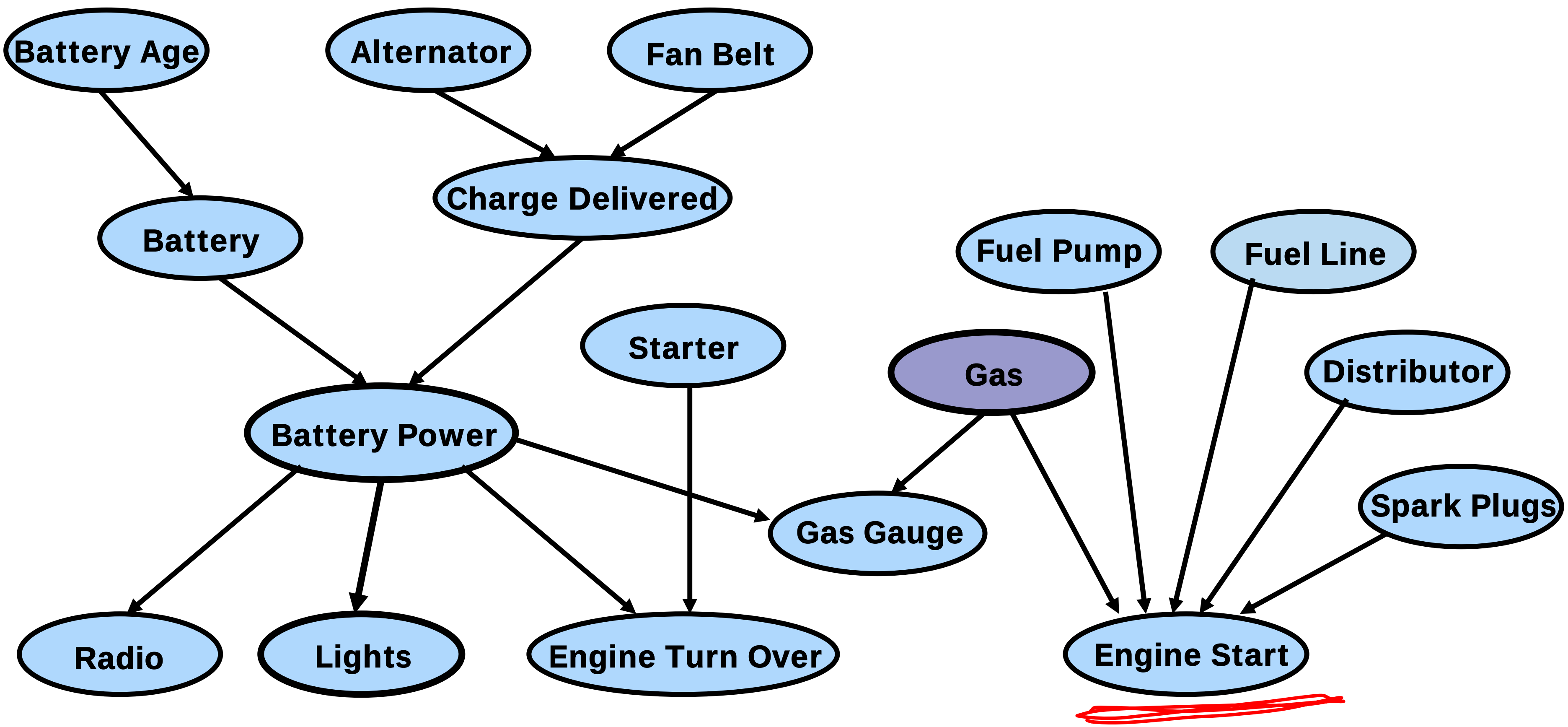
- Solve marginal queries for all variables in only twice the cost of query for one variable
- Cliques correspond to maximal cliques in induced graph
- Two message passing approaches
  - VE (the one that multiplies messages)
  - BP (the one that divides by old message)
- Clique tree invariant
  - Clique tree potential is always the same
  - We are only reparameterizing clique potentials
- Constructing clique tree for a BN
  - from elimination order
  - from triangulated (chordal) graph
- Running time (only) exponential in size of largest clique
  - Solve **exactly** problems with thousands (or millions, or more) of variables, and cliques with tens of nodes (or less)

Global Structure: Treewidth w

$O(n \exp(w))$

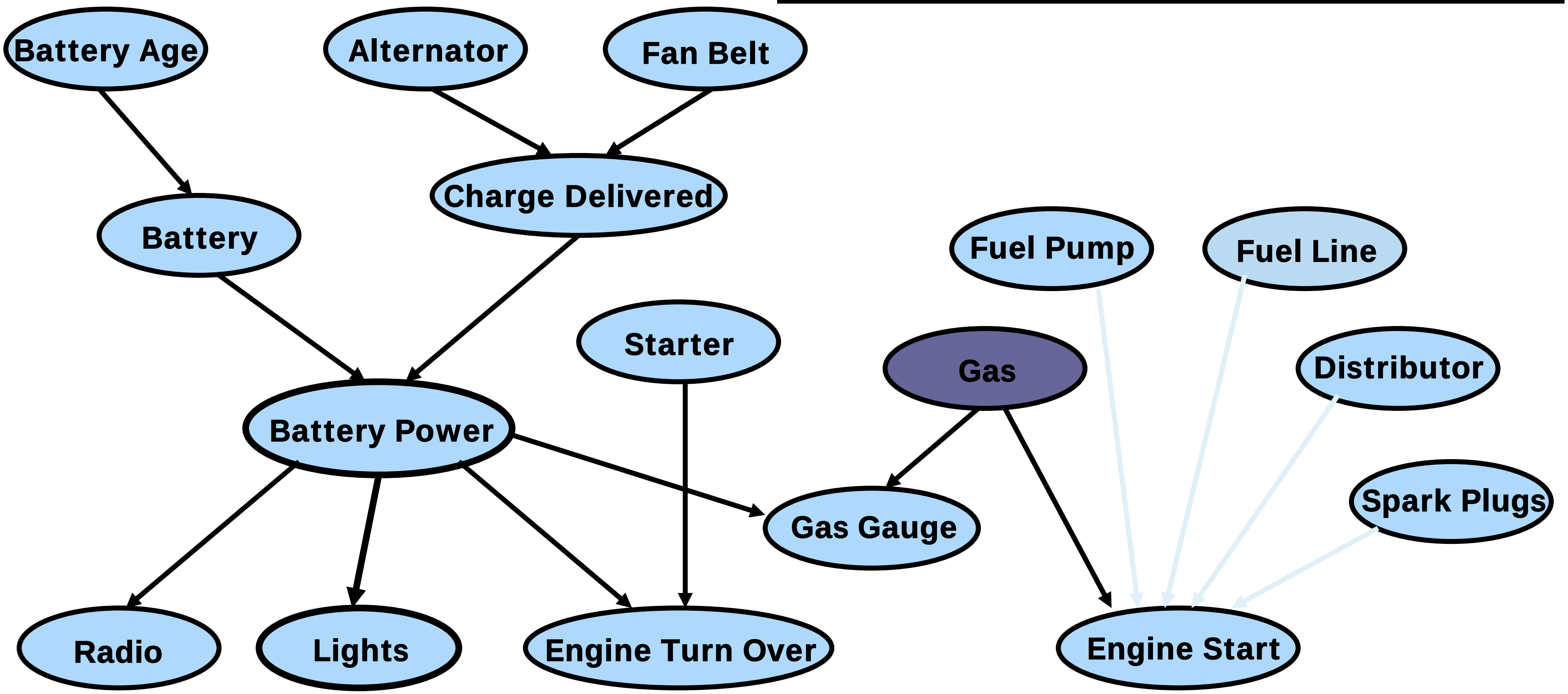


Local Structure 1: Context specific indepencence



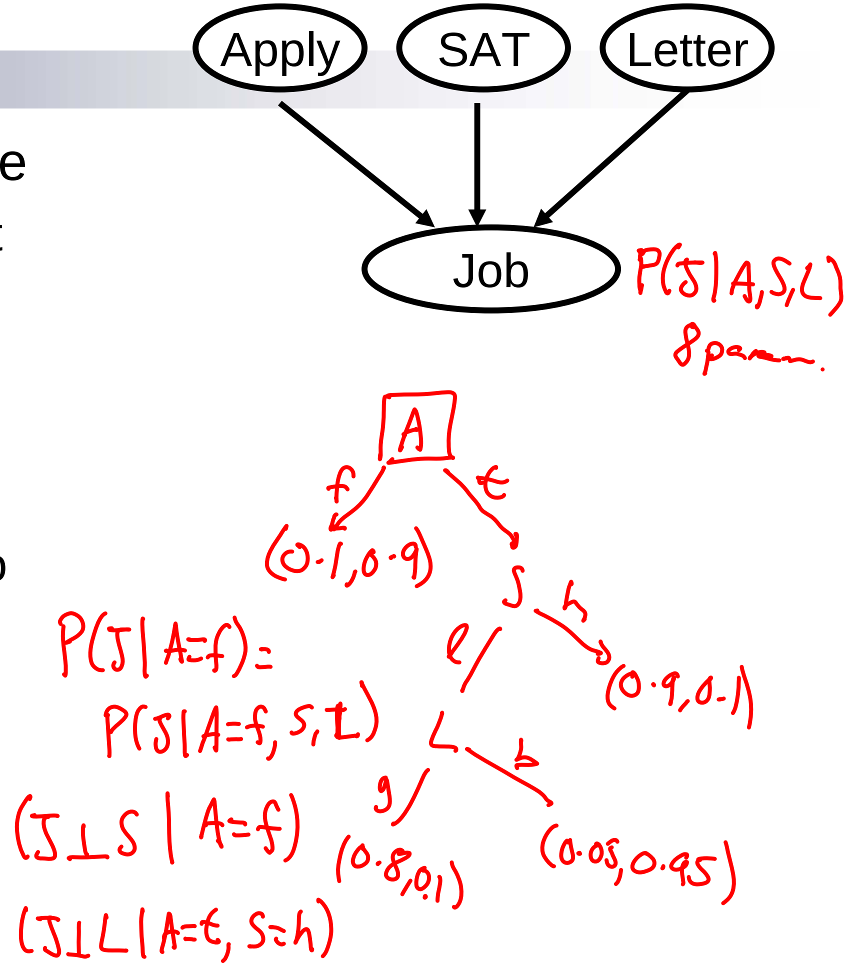
Local Structure 1: Context specific indepencence

Context Specific Independence (CSI)
After observing a variable, some vars become independent



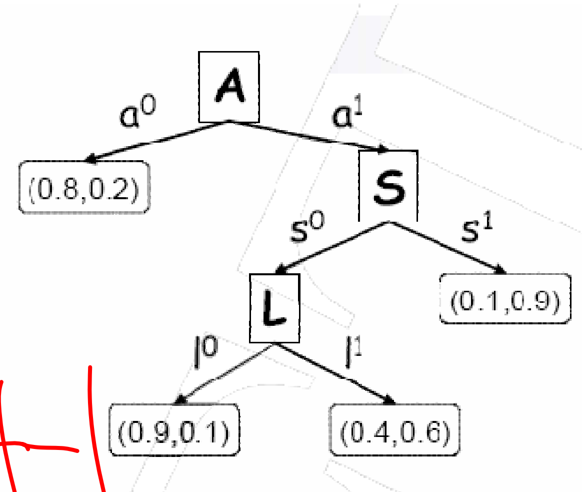
CSI example: Tree CPD

- Represent $P(X_i|\mathbf{Pa}_{X_i})$ using a decision tree
  - Path to leaf is an assignment to (a subset of) $\mathbf{Pa}_{X_i}$
  - Leaves are distributions over X_i given assignment of $\mathbf{Pa}_{X_i}$ on path to leaf
- **Interpretation of leaf**:
  - For specific assignment of $\mathbf{Pa}_{X_i}$ on path to this leaf – X_i is independent of other parents
- Representation can be exponentially smaller than equivalent table



Tabular VE with Tree CPDs

- If we turn a **tree CPD into table**
  - **“Sparsity” lost**!
- Need inference approach that **deals with tree CPD directly**!

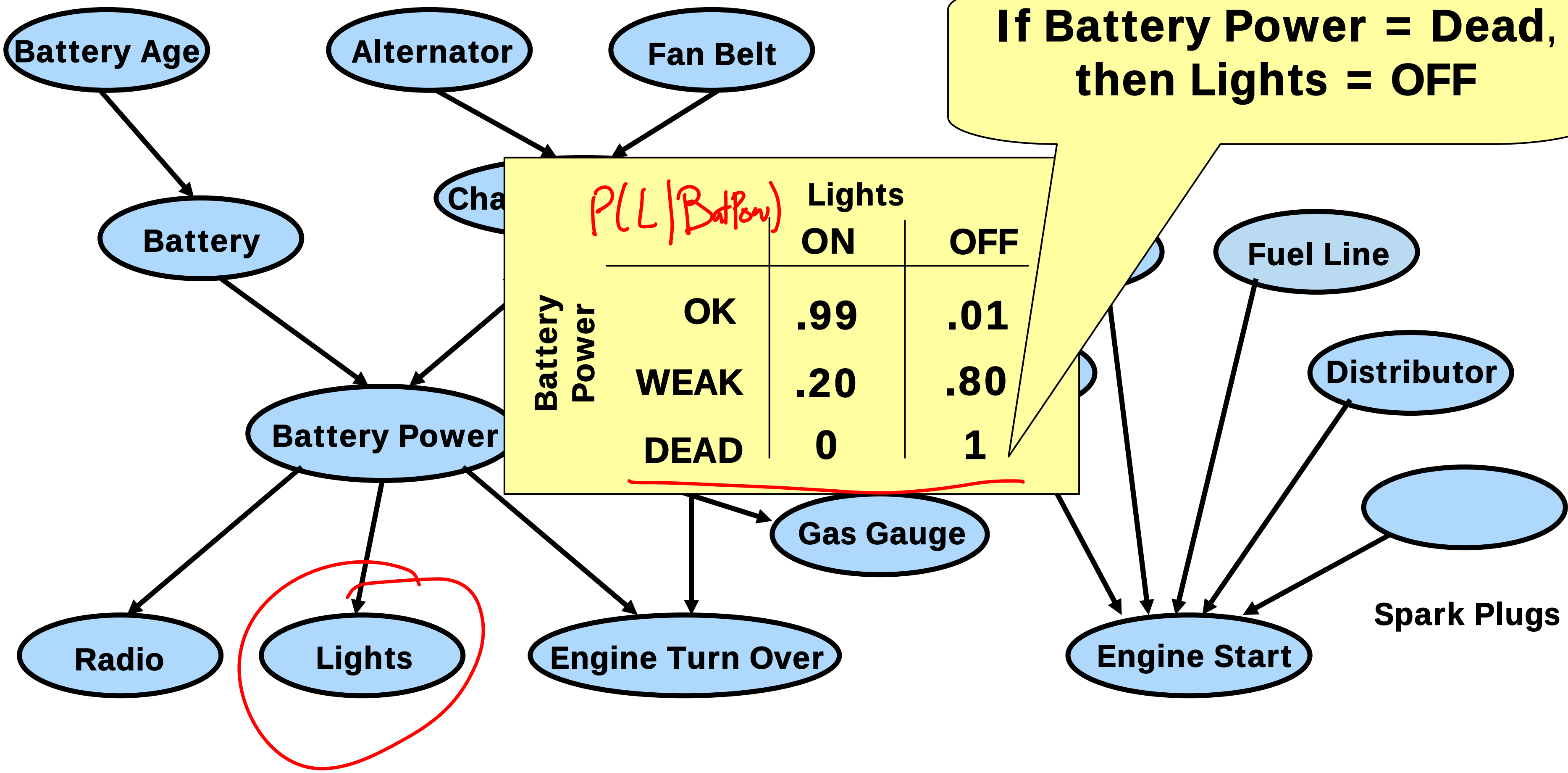


$P(J \mid A,S,L)$	$\bar{a},\bar{s},\bar{l}$	$\bar{a},\bar{s},l$	$\bar{a},s,\bar{l}$	$\bar{a},s,l$	$a,\bar{s},\bar{l}$			
J=t	0.2	0.2	0.2	0.2	0.1			
J=f	0.8	0.8	0.8	0.8	0.9			

repeated

Local Structure 2: Determinism

Determinism



If Battery Power = Dead, then Lights = OFF

P(L | BatPow)

Battery Power	Lights ON	Lights OFF
OK	.99	.01
WEAK	.20	.80
DEAD	0	1

Determinism and inference

- Determinism gives a little sparsity in table, but much bigger impact on inference
- Multiplying deterministic factor with other factor introduces many new zeros
  - Operations related to theorem proving, e.g., unit resolution

Battery Power	Lights: ON	Lights: OFF
OK	.99	.01
WEAK	.20	.80
DEAD	0	1

$$g(A,B,C,D) = f(A,B) \cdot f(B,C,D)$$

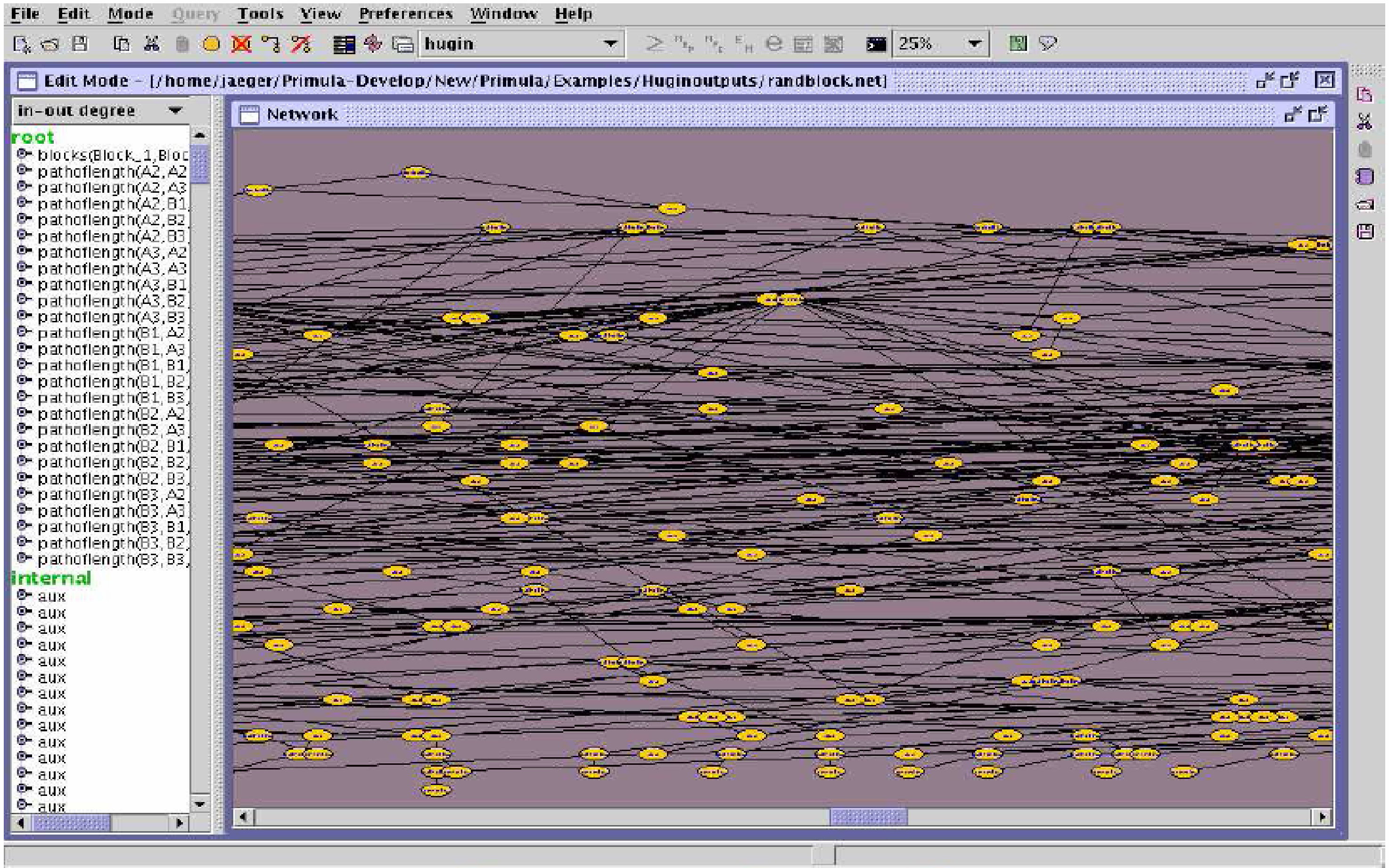
$$f(A,B=t) = 0$$

$$\Rightarrow g(A,B=t,C,D) = 0 \quad \forall A, C, D$$

Today's Models …

- **Often characterized by:**
  - Richness in local structure (determinism, CSI)
  - Massiveness in size (10,000's variables)
  - High connectivity (treewidth)
- **Enabled by:**
  - High level modeling tools: relational, first order
  - Advances in machine learning
  - New application areas (synthesis):
    - Bioinformatics (e.g. linkage analysis)
    - Sensor networks
- **Exploiting local structure a must!**

Exact inference in large models is possible…

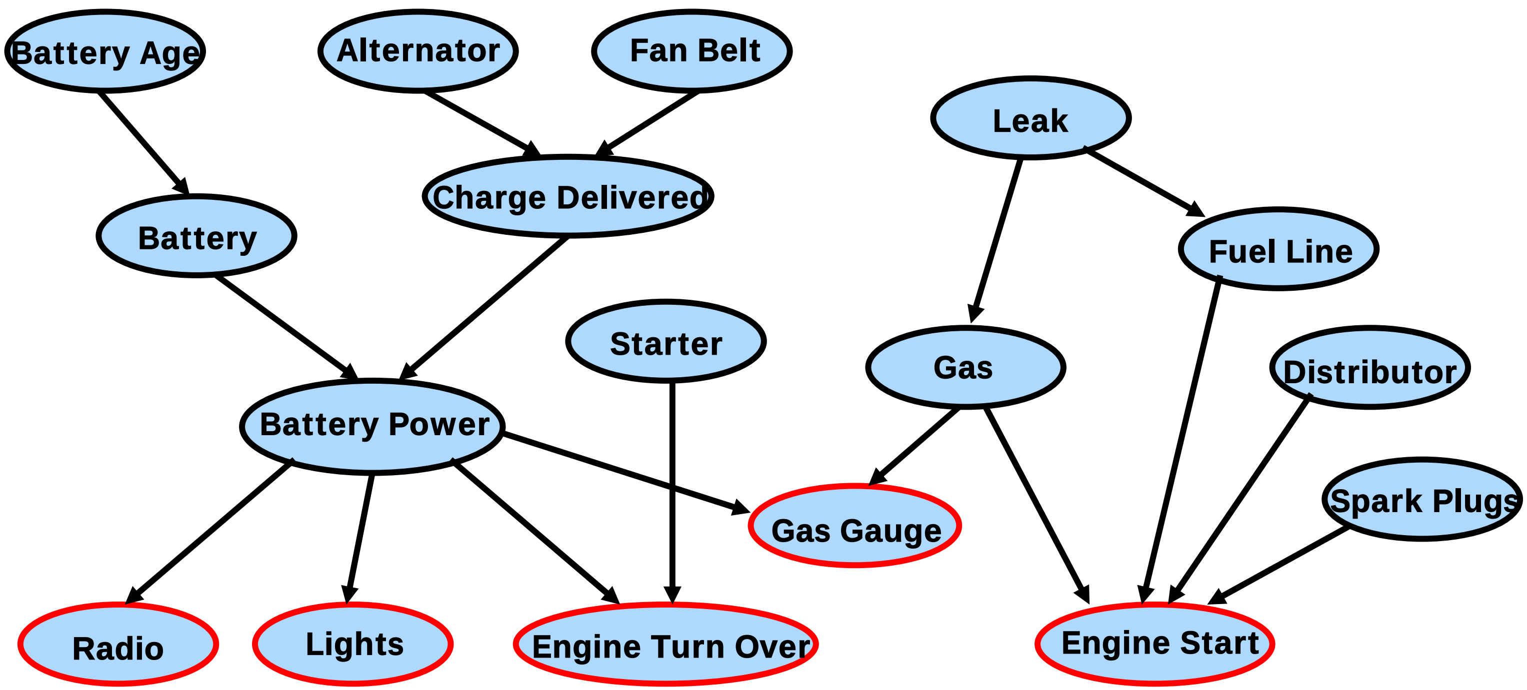


- BN from a relational model

Recursive Conditioning

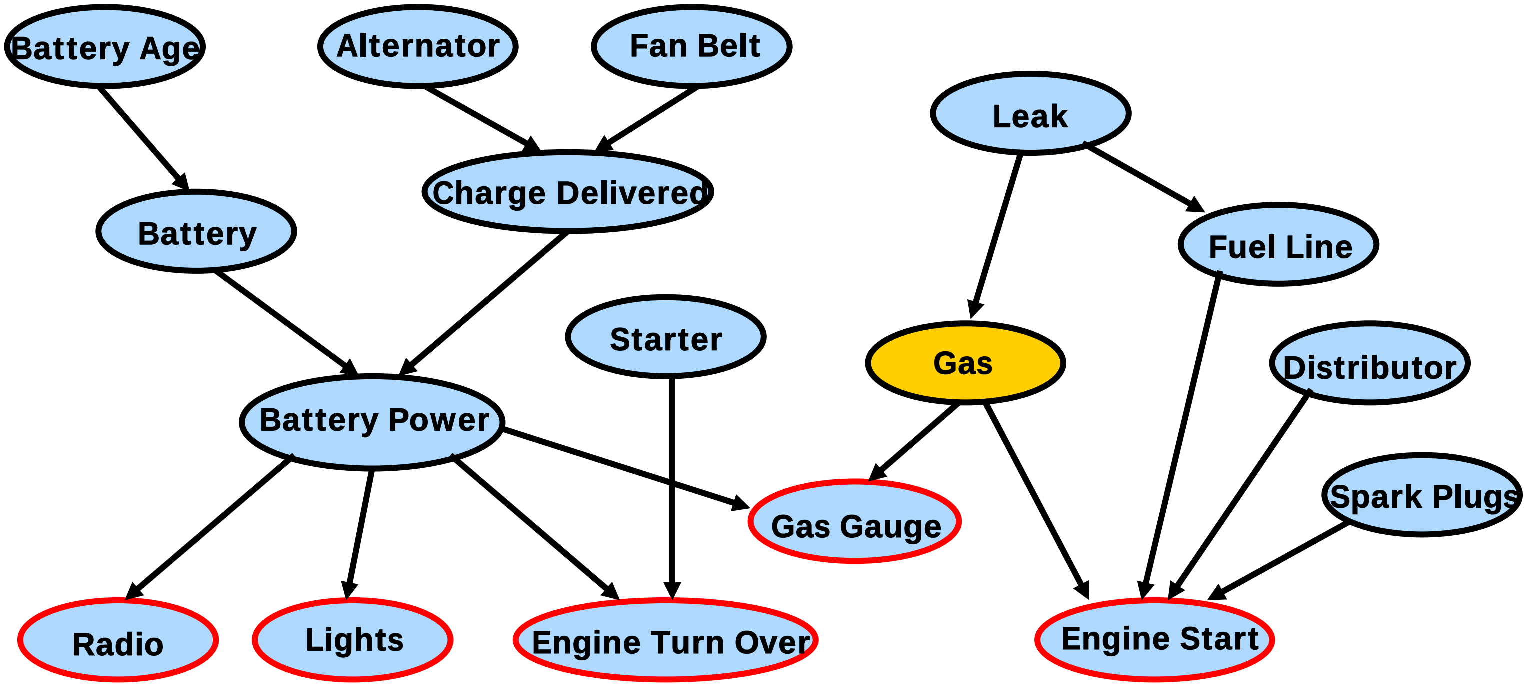
- Treewidth complexity (worst case)
- Better than treewidth complexity with local structure
- Provides a framework for time-space tradeoffs
- Only quick intuition today, details:
  - Koller&Friedman: 3.1-3.4, 6.4-6.6
  - “Recursive Conditioning”, Adnan Darwiche. In *Artificial Intelligence Journal,* 125:1, pages 5-41

The Computational Power of Assumptions



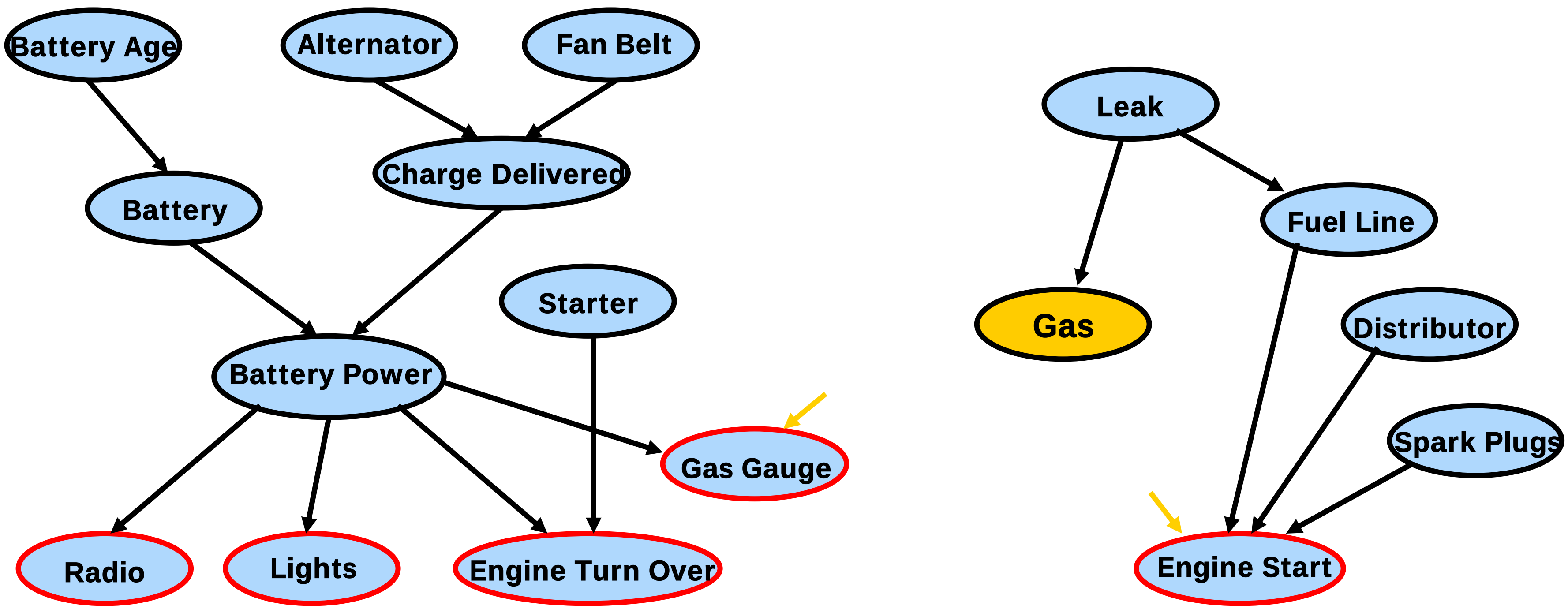
A. Darwiche

The Computational Power of Assumptions

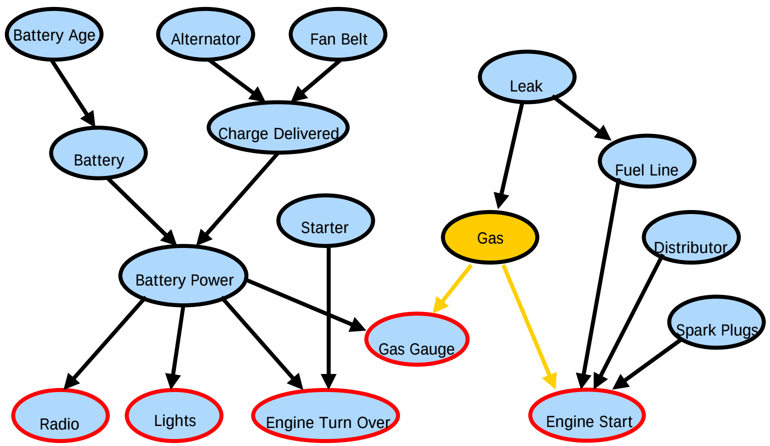


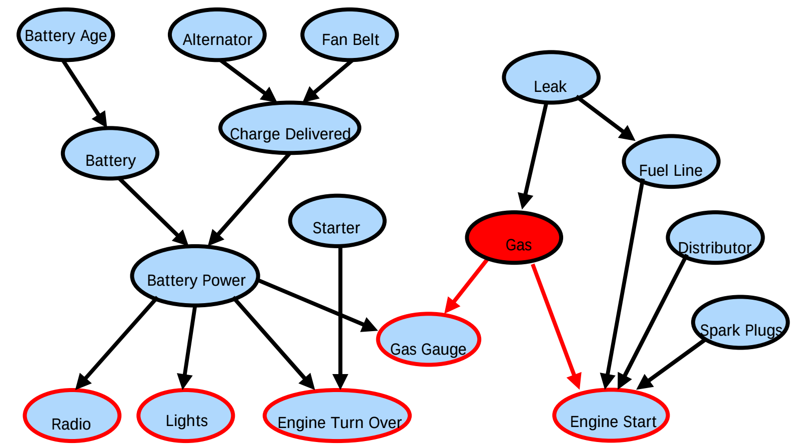
A. Darwiche

Decomposition



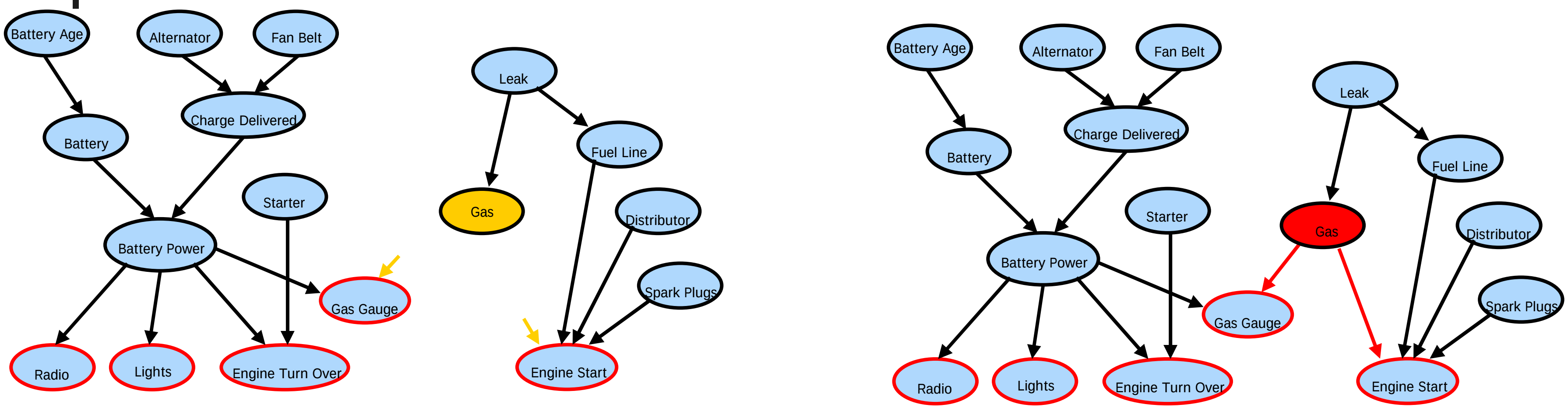
Case Analysis





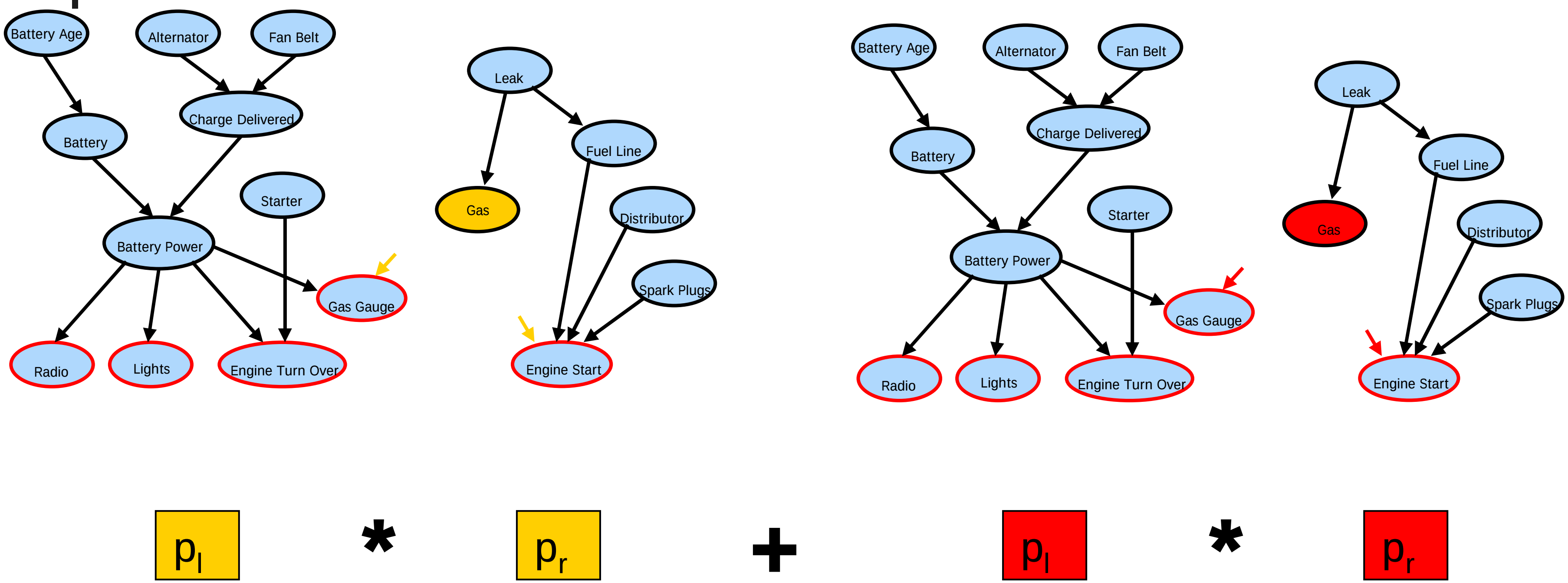
p + p

Case Analysis

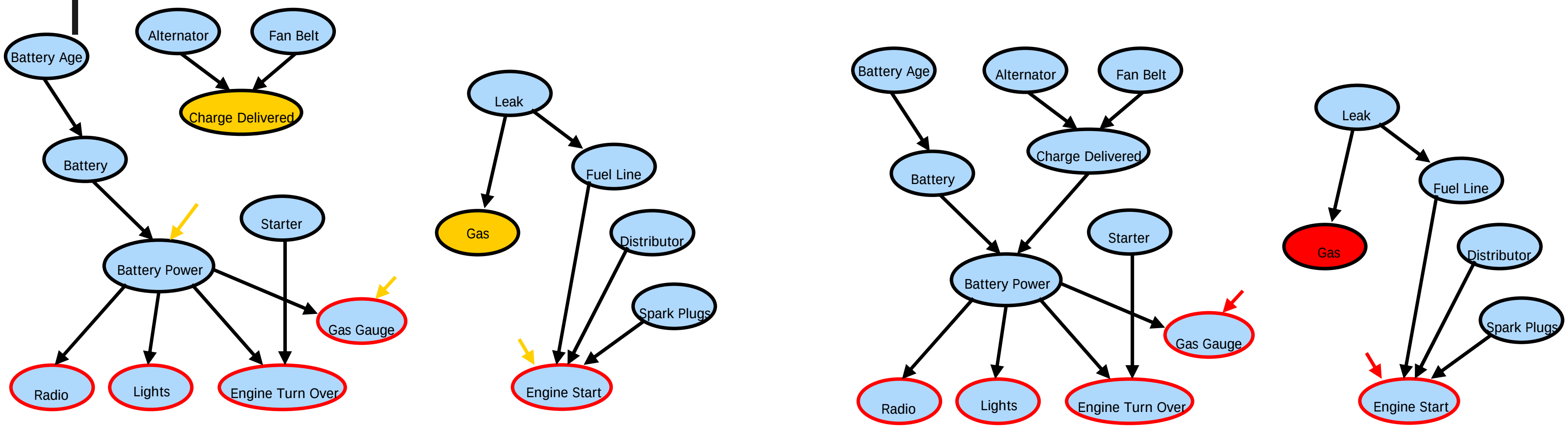


$p_l * p_r + p$

Case Analysis



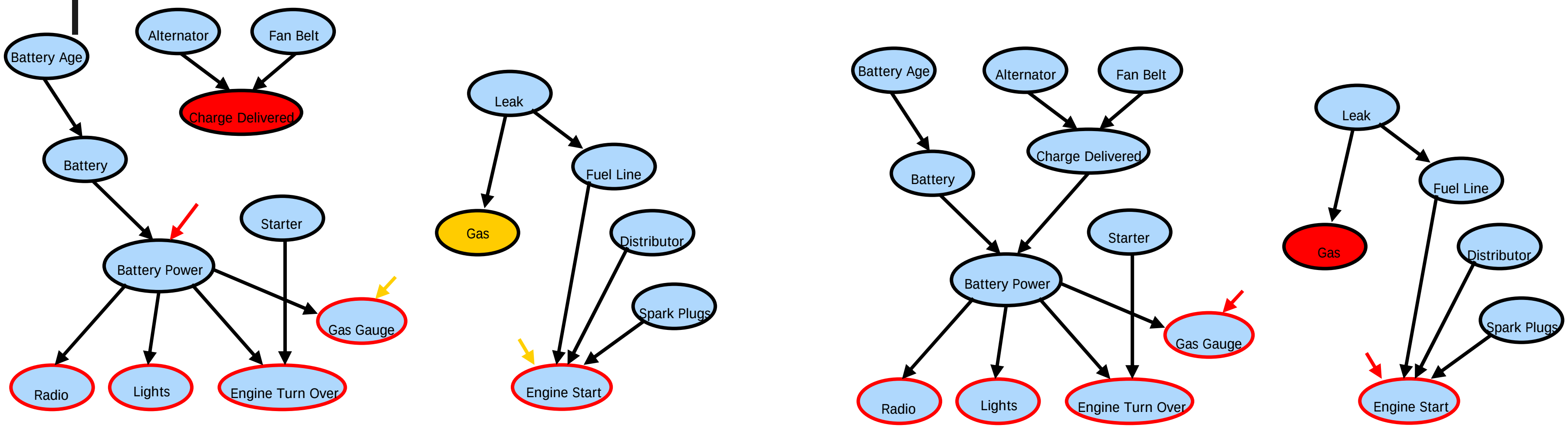
Case Analysis



p_l * p_r + p_l * p_r

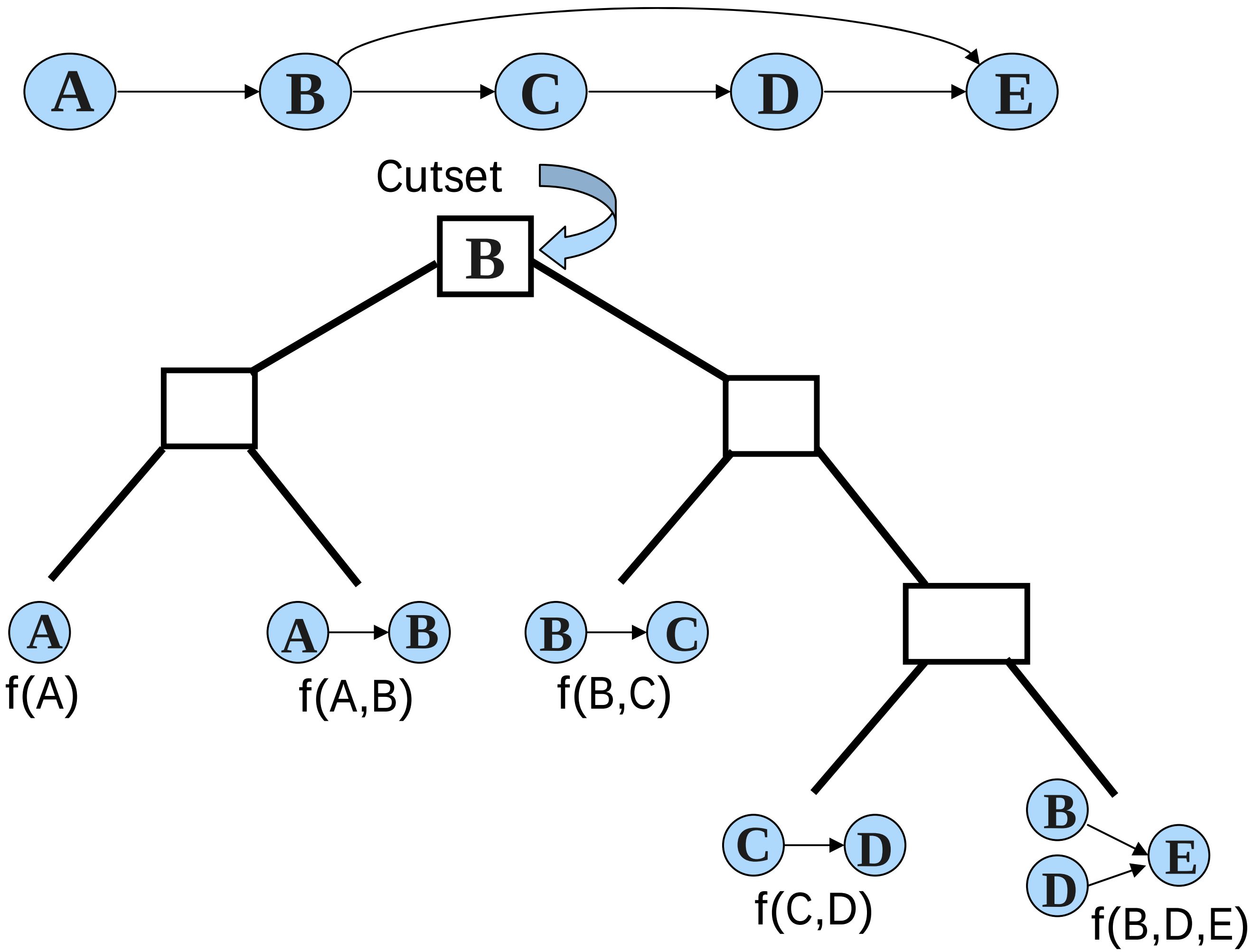
A. Darwiche

Case Analysis



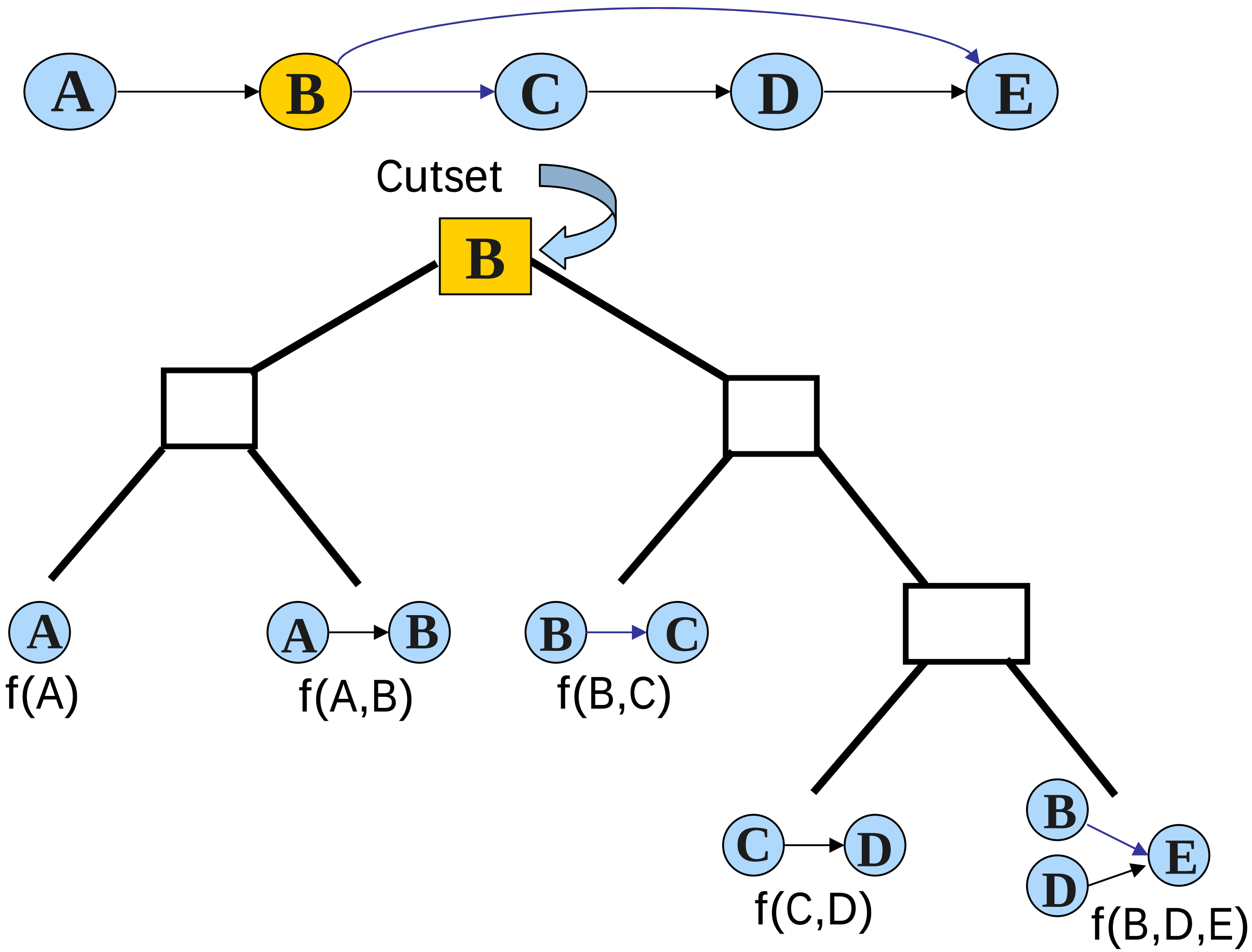
p_l * p_r + p_l * p_r

Decomposition Tree



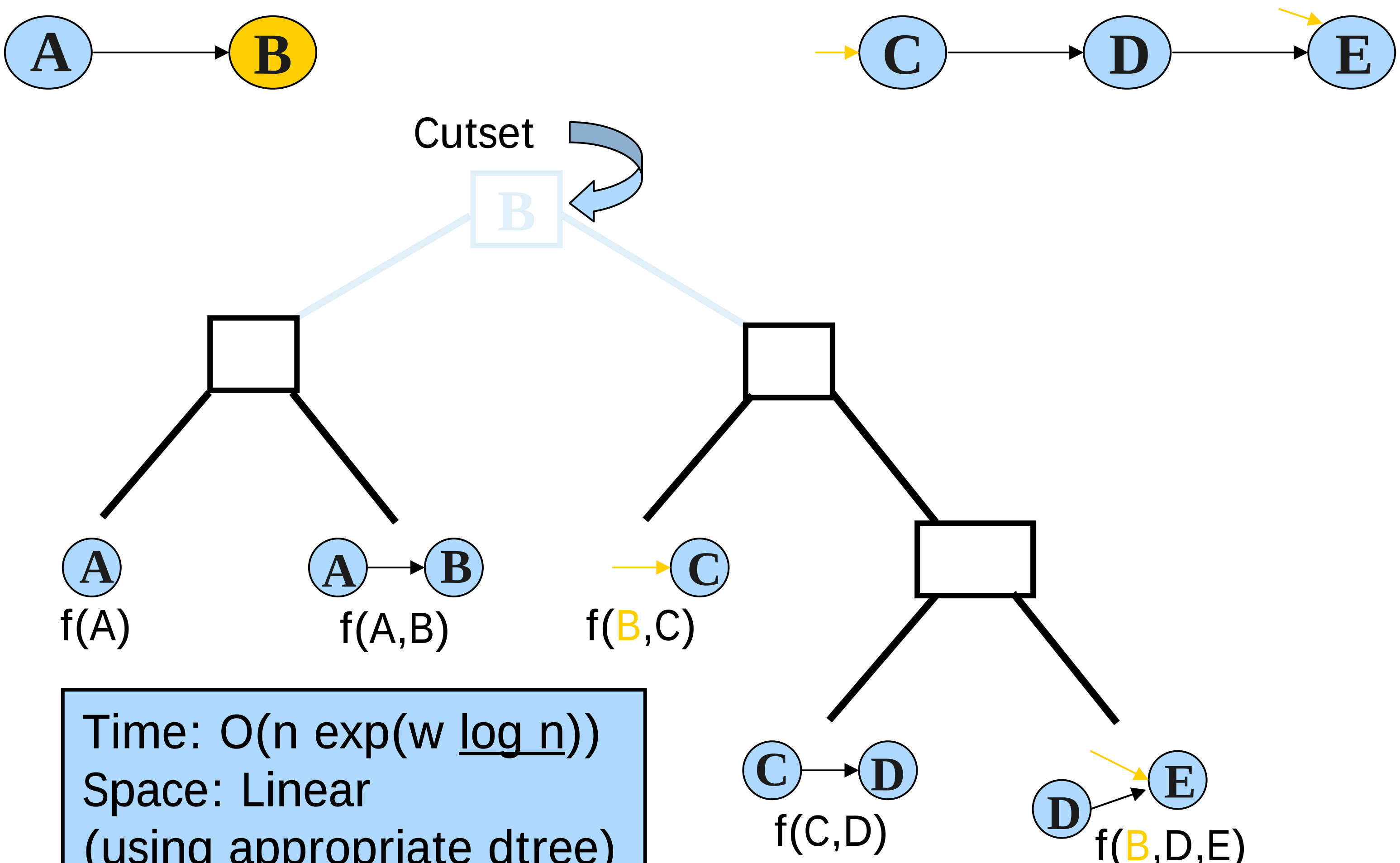
A. Darwiche

Decomposition Tree



A. Darwiche

Decomposition Tree



Time: O(n exp(w log n))
Space: Linear
(using appropriate dtree)

A. Darwiche

RC1

```
RC1(T,e)
    // compute probability of evidence e on dtree T

    If T is a leaf node
    Return Lookup(T,e)
    Else
        p := 0
        for each instantiation c of cutset(T)-E do
            p := p + RC1(Tl,ec) RC1(Tr,ec)
        return p
```

Lookup(T,**e**)

$\Theta_{X|\mathbf{U}}$: CPT associated with leaf T

If X is instantiated in **e**, then

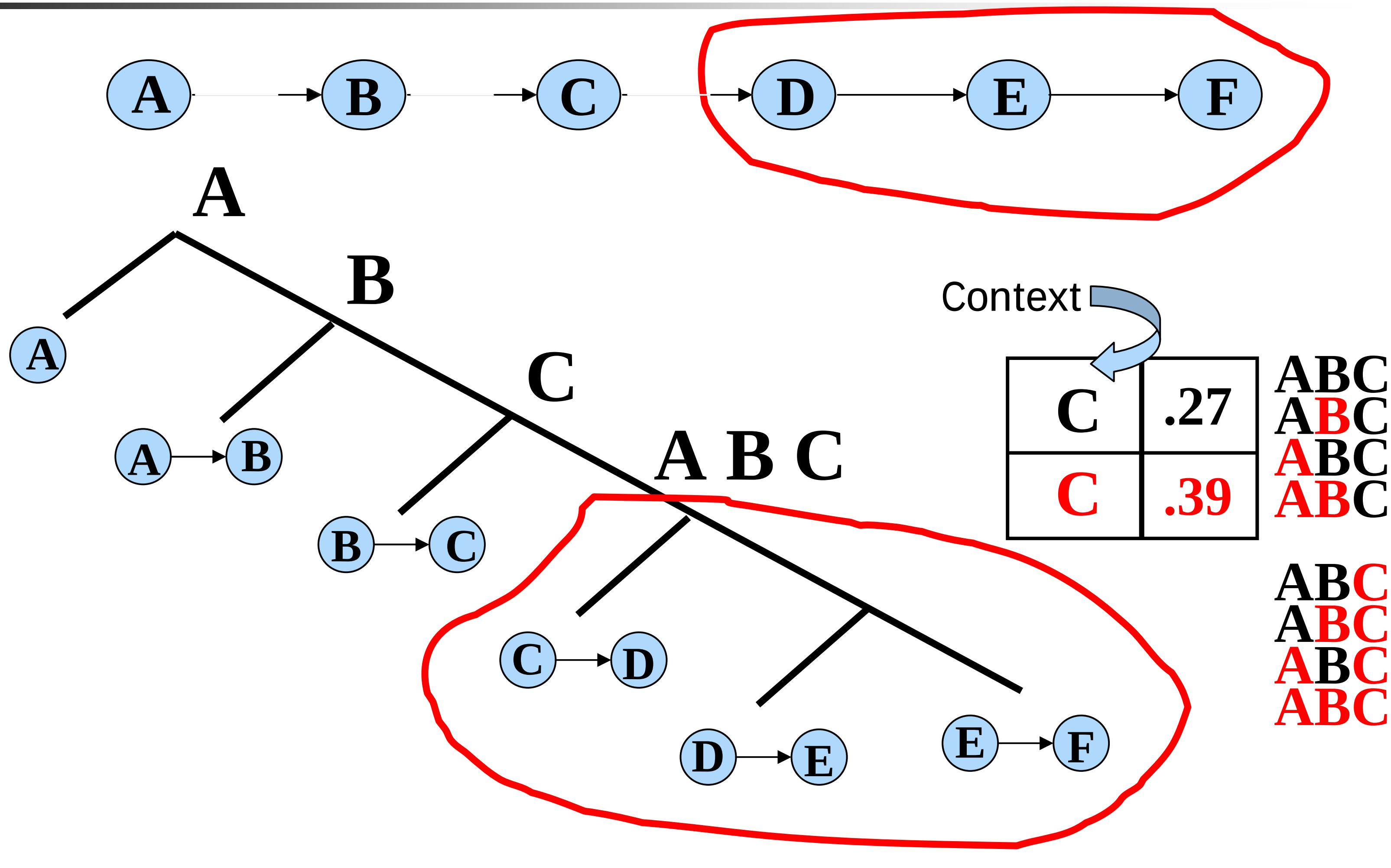
x: value of X in **e**

u: value of **U** in **e**

Return $\theta_{x|\mathbf{u}}$

Else return $1 = \Sigma_x \ \theta_{x|\mathbf{u}}$

Caching


A
B
C
D
E
F
A
B
C
A B C
A
A
B
B
C
C
D
D
E
E
F
Context
C
.27
C
.39
ABC
ABC
ABC
ABC
ABC
ABC
ABC
ABC

Caching

Recursive Conditioning

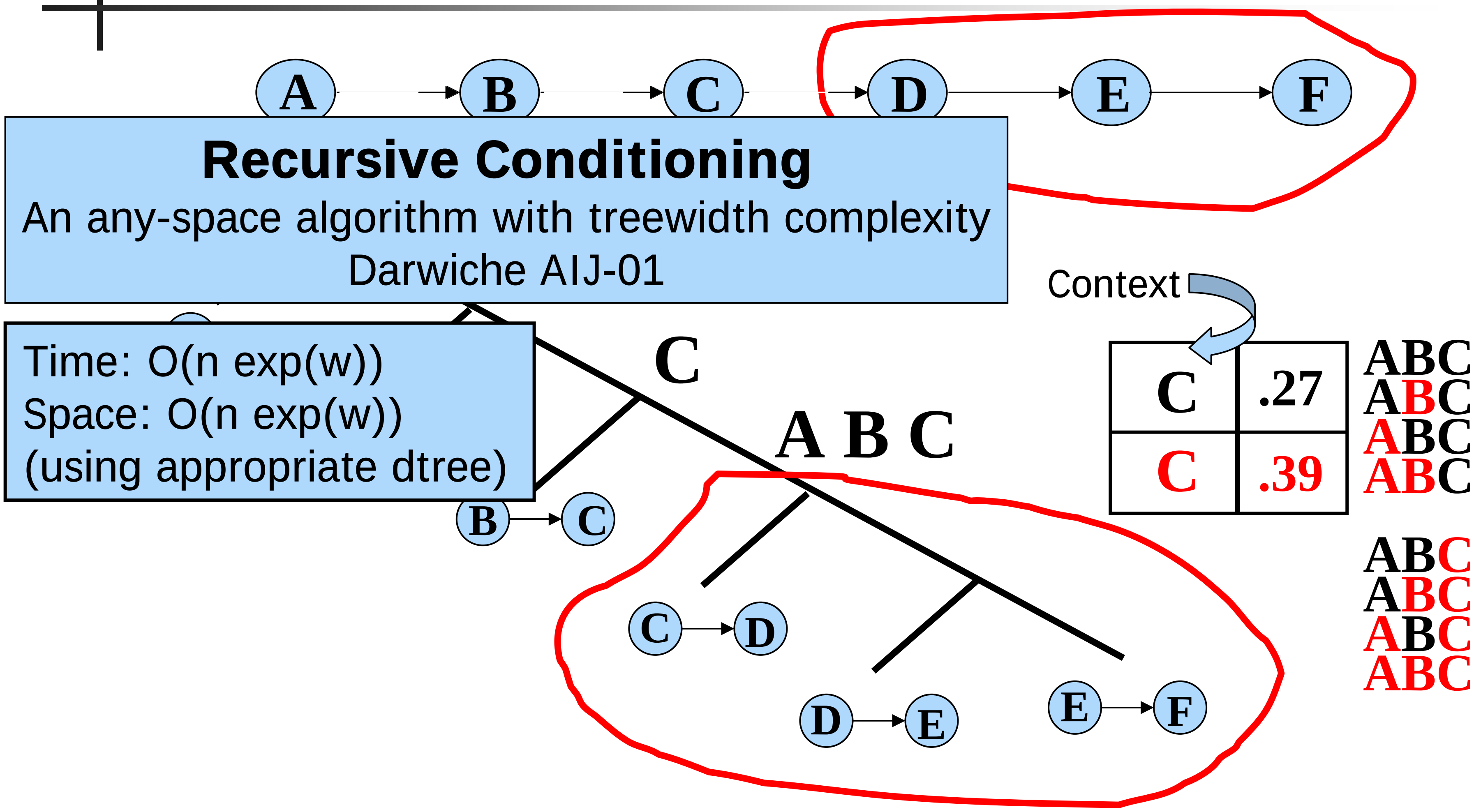
An any-space algorithm with treewidth complexity

Darwiche AIJ-01

Time: O(n exp(w))

Space: O(n exp(w))

(using appropriate dtree)



RC2

RC2(T,**e**)

If T is a leaf node, return Lookup(T,e)

y := instantiation of context(T)

If $cache_T$[**y**] <> nil, return $cache_T$[**y**]

p := 0

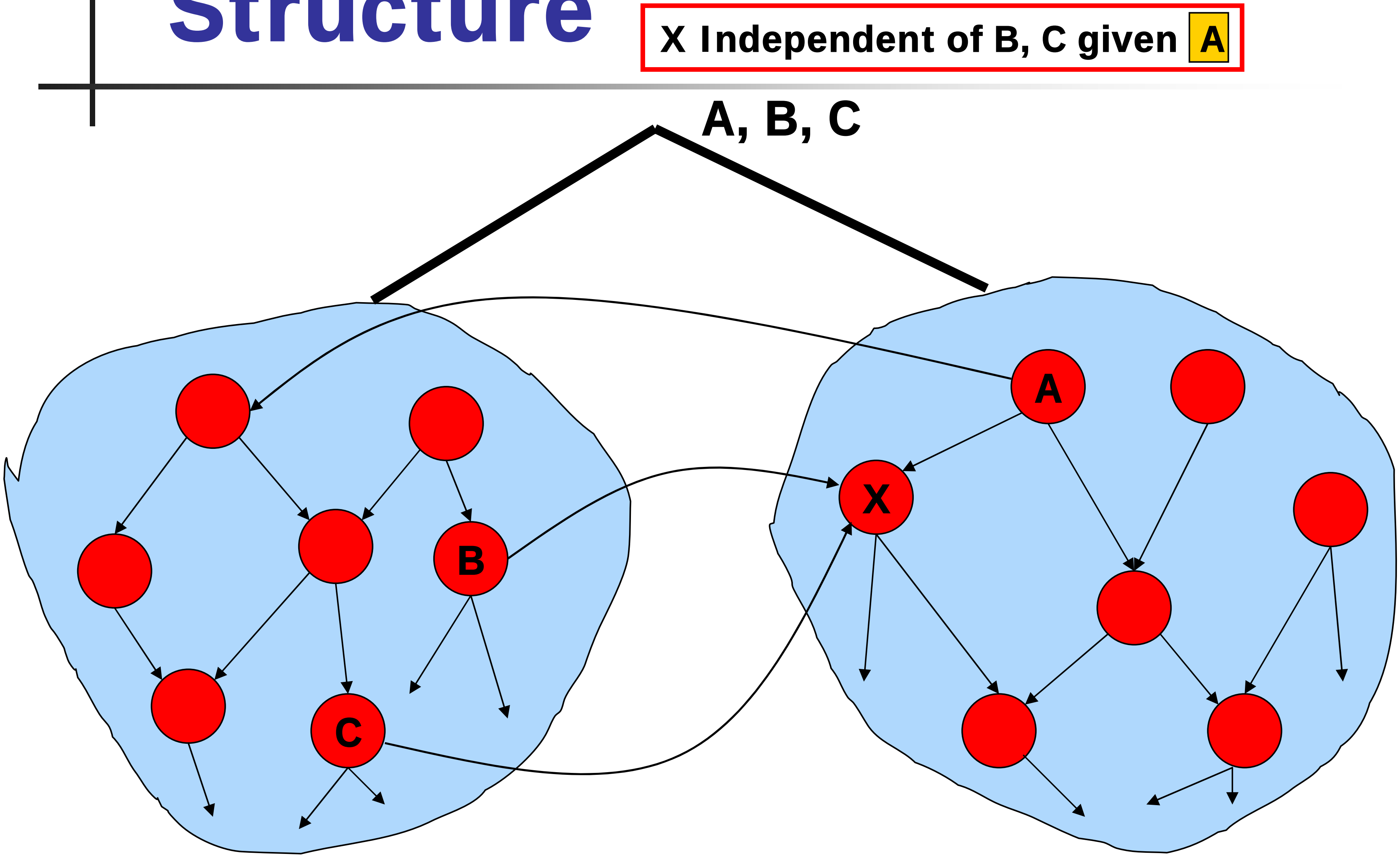
For each instantiation **c** of cutset(T)-**E** do

p := p + RC2(T^l,**ec**) RC2(T^r,**ec**)

$cache_T$[**y**] := p

Return p

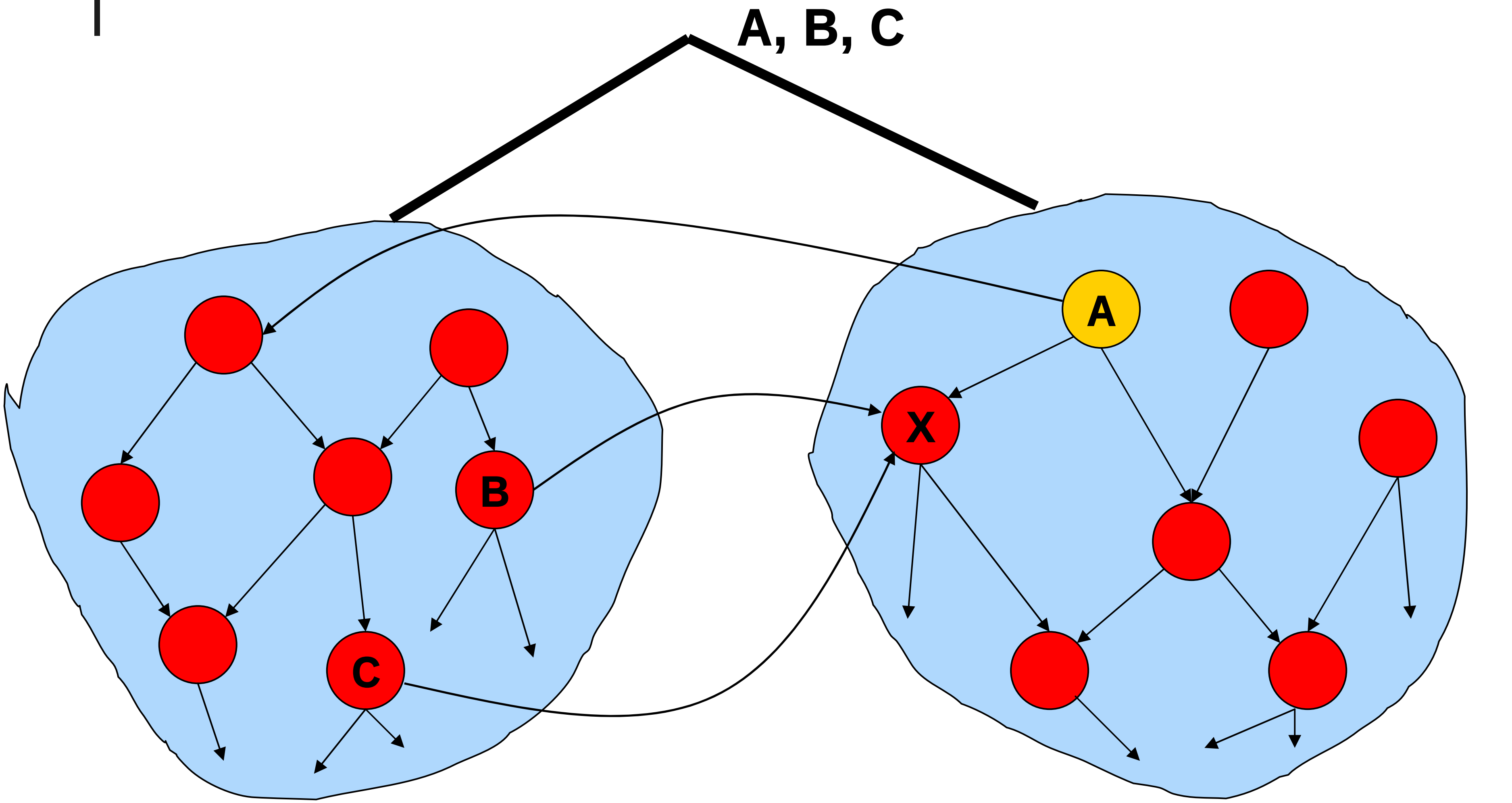
Decomposition with Local Structure



A. Darwiche

Decomposition with Local Structure

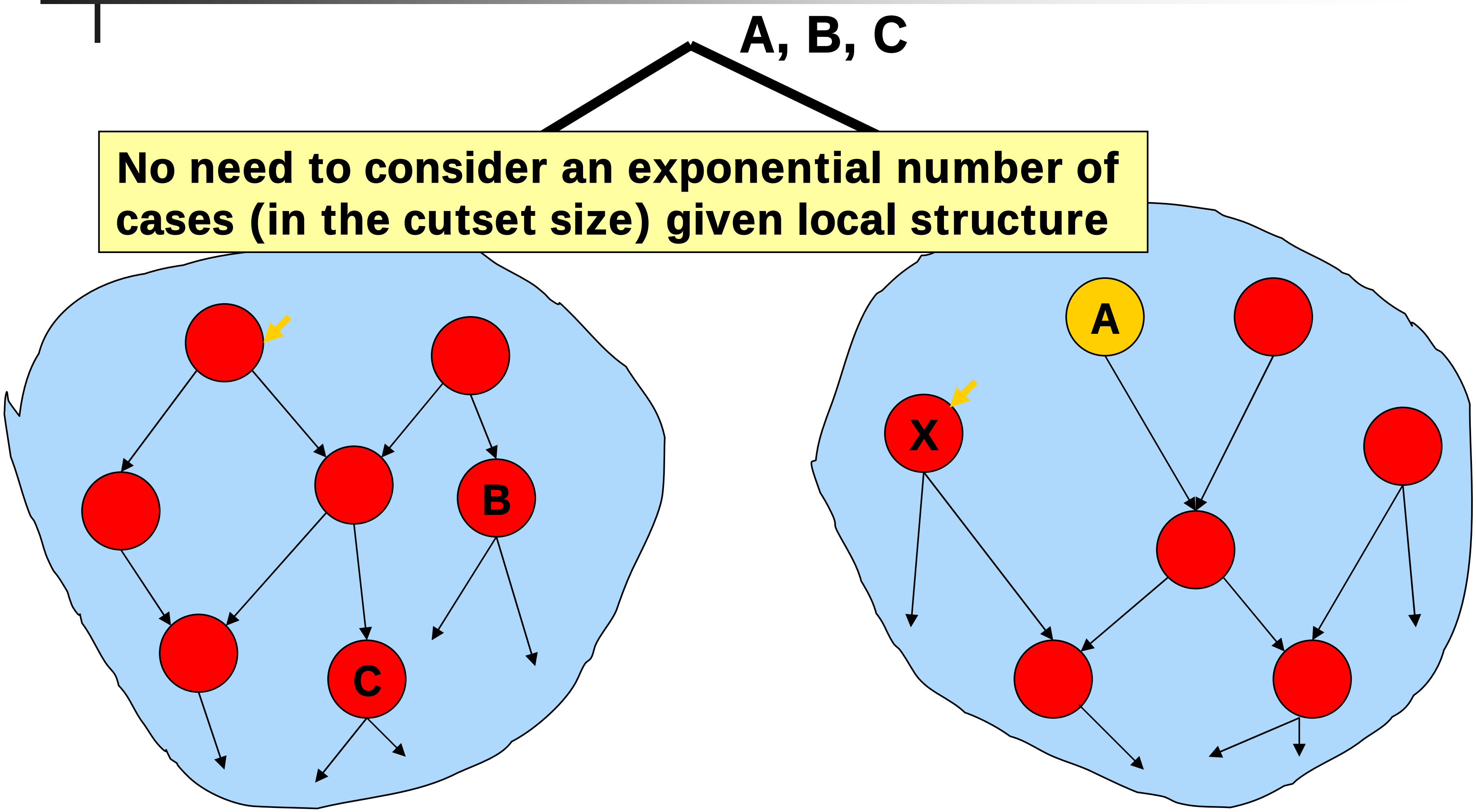
X Independent of B, C given A



A. Darwiche

Decomposition with Local Structure

X Independent of B, C given A

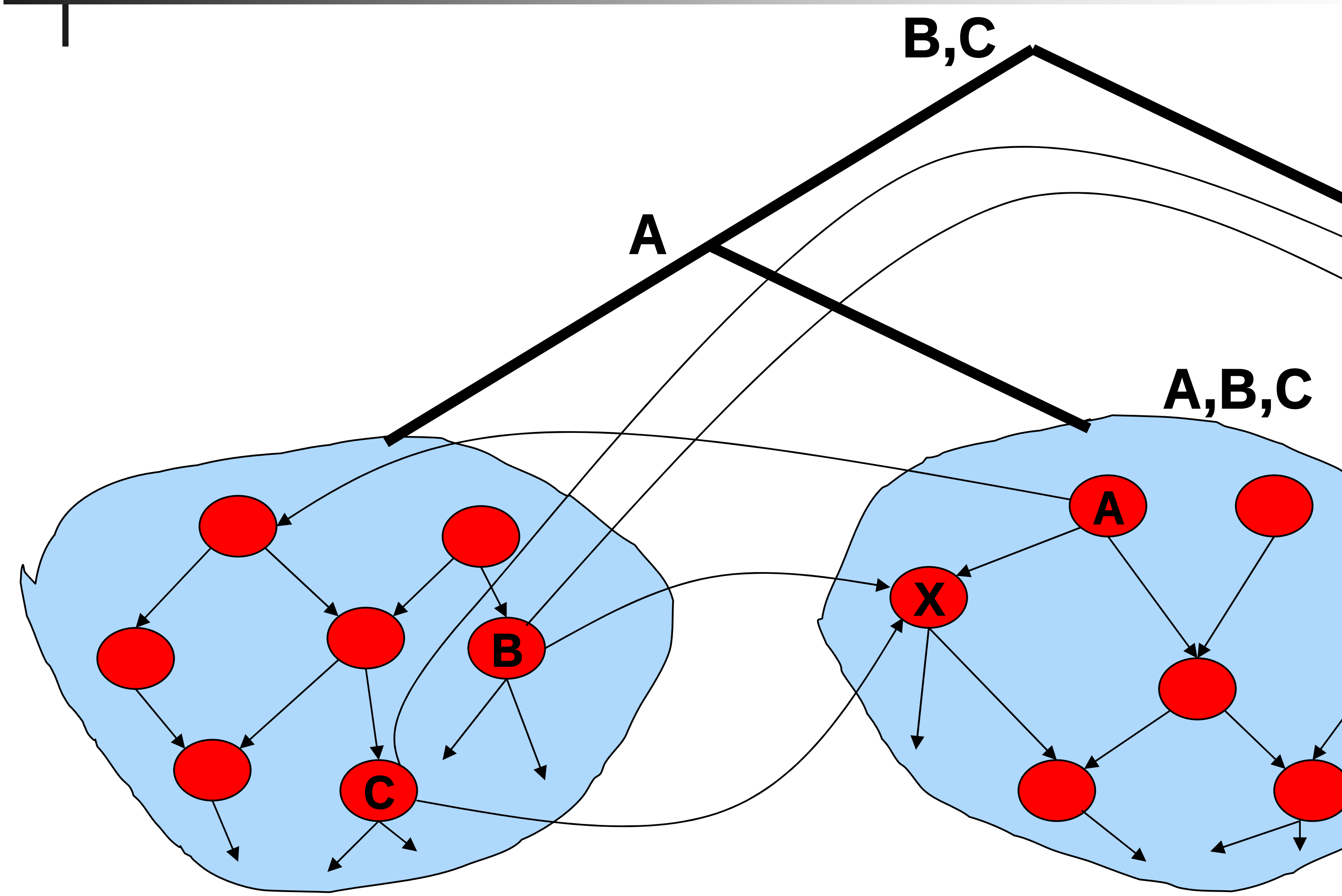


A. Darwiche

Caching with Local Structure

Structural cache

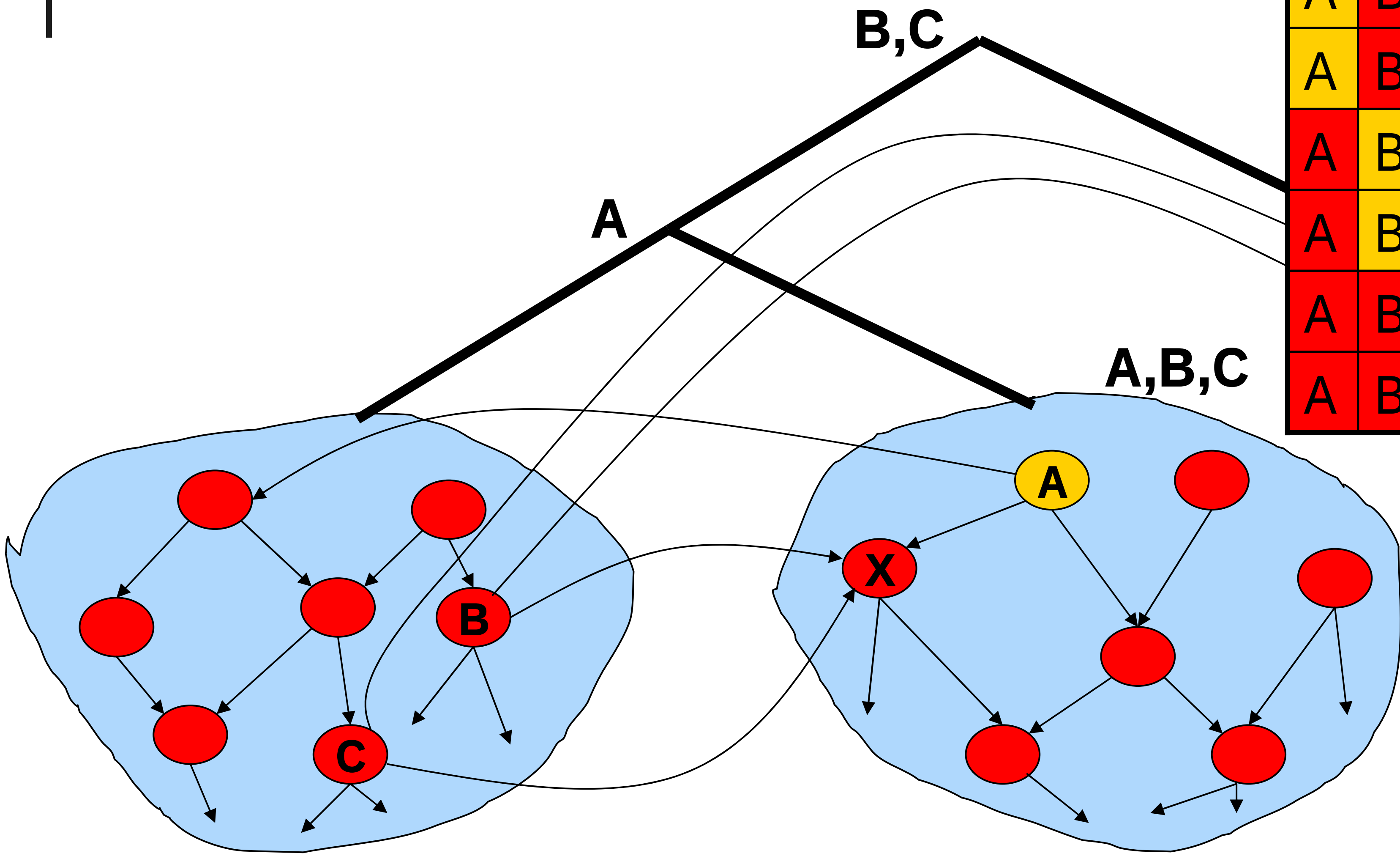
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C



Caching with Local Structure

Structural cache

A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C



Caching with Local

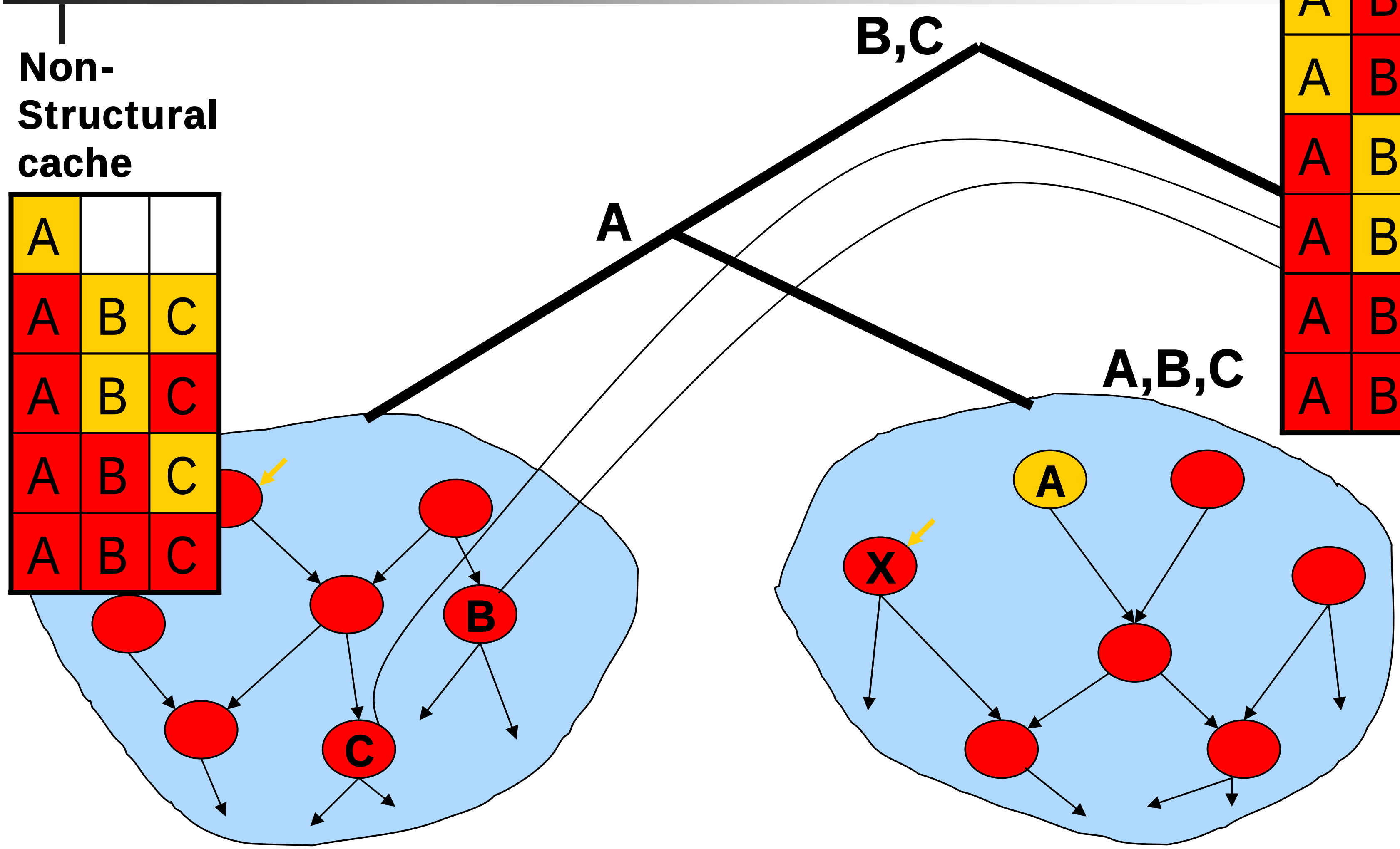
No need to cache an exponential number of results (in the context size) given local structure

Structural cache

A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C
A	B	C

Non-Structural cache

A		
A	B	C
A	B	C
A	B	C
A	B	C



A. Darwiche

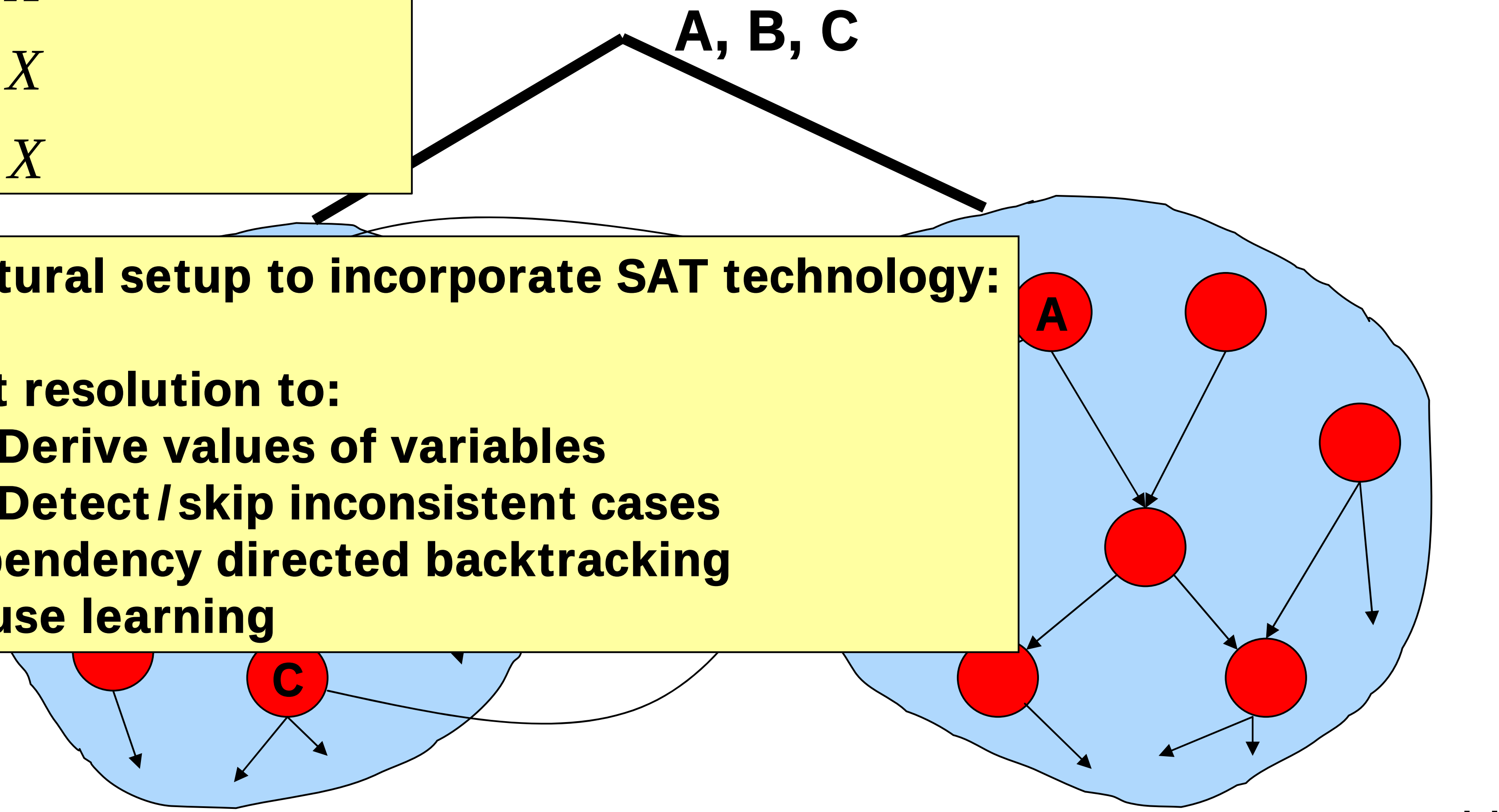
Determinism…

$$\neg A \wedge \neg B \wedge \neg C \Rightarrow \neg X$$

$$A \Rightarrow X$$

$$B \Rightarrow X$$

$$C \Rightarrow X$$



A natural setup to incorporate SAT technology:

- **Unit resolution to:**
  - **Derive values of variables**
  - **Detect / skip inconsistent cases**
- **Dependency directed backtracking**
- **Clause learning**

A. Darwiche

CSI Summary

- Exploit local structure
  - Context-specific independence
  - Determinism
- Significantly speed-up inference
  - Tackle problems with tree-width in the thousands

- Acknowledgements
  - Recursive conditioning slides courtesy of Adnan Darwiche
  - Implementation available:
    - http://reasoning.cs.ucla.edu/ace

Where are we?

- Bayesian networks
  - Represent exponentially-large probability distributions compactly
- Inference in BNs
  - Exact inference very fast for problems with low tree-width
  - Exploit local structure for fast inference
- **Now: Learning BNs**
  - Given structure, estimate parameters

Thumbtack – Binomial Distribution

- P(Heads) = θ, P(Tails) = 1-θ

$\theta = \frac{3}{5}$

$P(HHTHT) = \theta\theta(1-\theta)\theta(1-\theta) = \theta^3(1-\theta)^2$

- Flips are i.i.d.:
  - Independent events
  - Identically distributed according to Binomial distribution
- Sequence D of α_H Heads and α_T Tails

$$P(\mathcal{D} \mid \theta) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

Maximum Likelihood Estimation

- **Data:** Observed set *D* of α_H Heads and α_T Tails   i.i.d.
- **Hypothesis:** Binomial distribution
- Learning θ is an optimization problem
  - What's the objective function?

$$\hat{\theta} = \arg\max_{\theta} P(HHTHT \mid \theta)$$

- MLE: Choose θ that maximizes the probability of observed data:

$$\begin{aligned}\hat{\theta} &= \arg\max_{\theta} \; P(\mathcal{D} \mid \theta) \\ &= \arg\max_{\theta} \; \ln P(\mathcal{D} \mid \theta)\end{aligned}$$

Your first learning algorithm

$$\widehat{\theta} = \arg\max_{\theta} \ \ln P(\mathcal{D} \mid \theta)$$
$$= \arg\max_{\theta} \ \ln \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

- Set derivative to zero: $\frac{d}{d\theta} \ \ln P(\mathcal{D} \mid \theta) = 0$

$$\frac{d}{d\theta} \ln \theta^{\alpha_H}(1-\theta)^{\alpha_T} = \frac{d}{d\theta} \alpha_H \ln \theta + \frac{d}{d\theta} \alpha_T \ln(1-\theta)$$

$$= \frac{\alpha_H}{\theta} - \frac{\alpha_T}{1-\theta} = 0 \quad \Rightarrow \quad \theta = \frac{\alpha_H}{\alpha_H+\alpha_T}$$

Multinomial dist, $|X|>2$

MLE for conditional probabilities

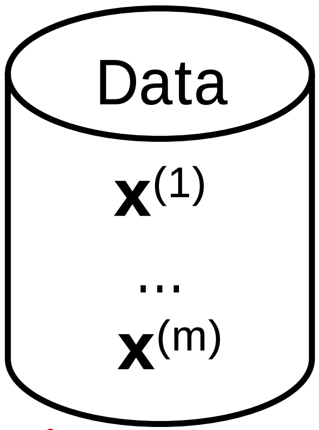
- MLE estimate of $\hat{P}(X=x) = \dfrac{\text{Count}(X=x)}{\text{Count}(\emptyset)}$

- MLE estimate of P(X=x|Y=y)
  - Only consider subset of data where Y=y

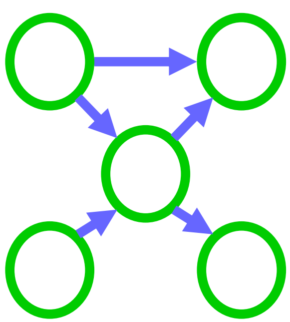
x, y
x, y
$x, \bar{y}$
$\bar{x}, y$
$\bar{x}, \bar{y}$

$$\hat{P}(X=x \mid Y=y) = \frac{\text{Count}(X=x, Y=y)}{\text{Count}(Y=y)}$$

Learning the CPTs



$x^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$



MLE: $\hat{P}(X_i = x_i \mid X_{i-1} = x_{i-1}) = \dfrac{\text{Count}(X_i = x_i, X_{i-1} = x_{i-1})}{\text{Count}(X_{i-1} = x_{i-1})}$

for this GM: $X_1 \rightarrow X_2 \rightarrow X_3$ ...

$$\ln P(D \mid \theta) = \ln \prod_i P(x^{(i)} \mid \theta) = \ln \prod_{i \in D} \prod_j P(x_j^{(i)} \mid x_{j-1}^{(i)}, \theta)$$

$$= \sum_{i \in D} \sum_j \ln P(x_j^{(i)} \mid x_{j-1}^{(i)}, \theta_{j|j-1})$$

$$= \sum_j \sum_{i \in D} \ln P(x_j^{(i)} \mid x_{j-1}^{(i)}, \theta_{j|j-1})$$

Decomposing score according to structure of BN

MLE learning CPTs for general BN

- Vars $X_1,\ldots,X_n$ and BN structure given $P(X_1,\ldots,X_n) = \prod_{i=1}^{n} P\left(X_i \mid \mathbf{Pa}_{X_i}\right)$
- Each i.i.d. data point assigns a value all vars no missing values
- Likelihood of the data:

$$\ln P(D \mid \theta) = \sum_i \sum_{j \in D} \ln P\left(x_i^{(j)} \mid Pa_{x_i}^{(j)}, \theta_{i \mid Pa_{x_i}}\right)$$

- MLE for CPT $P(X_i \mid \mathbf{Pa}_{X_i})$: $\widehat{P}(X_i = x_i \mid \mathbf{Pa}_{\mathbf{X_i}} = \mathbf{u}) = \frac{\text{Count}(X_i = x_i, \mathbf{Pa}_{\mathbf{X_i}} = \mathbf{u})}{\text{Count}(\mathbf{Pa}_{\mathbf{X_i}} = \mathbf{u})}$