Readings:

K&F: 4.1, 4.2, 4.3, 4.4, 8.4, 8.5, 8.6

"Recursive Conditioning", Adnan Darwiche. In Artificial Intelligence Journal, 125:1, pp. 5-41

Context-specific independence

CSI-BNs

Graphical Models – 10708

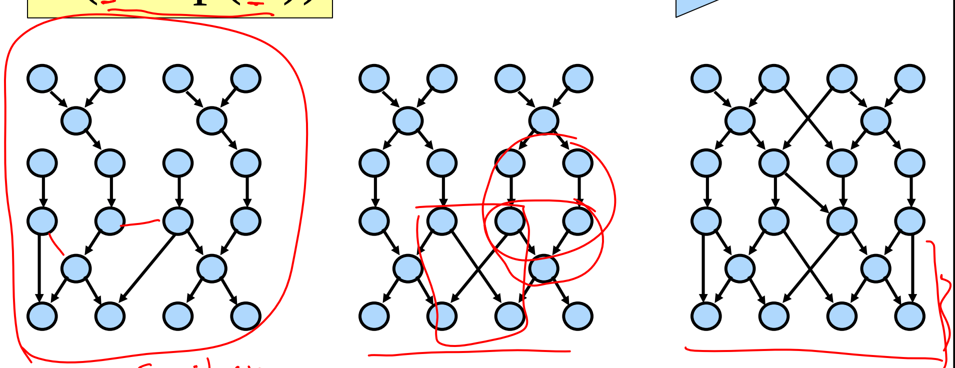
Carlos Guestrin

Carnegie Mellon University

October 16th, 2006

Global Structure: Treewidth w

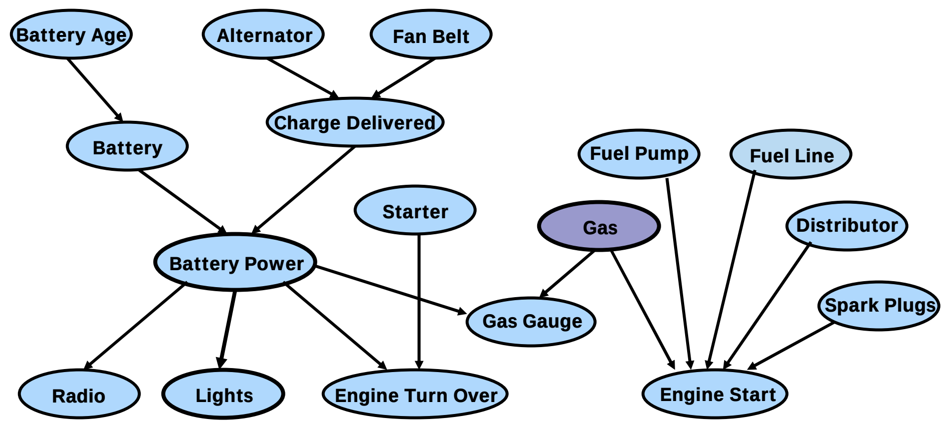
$O(n\exp(w))$



w = Family size +1

Local Structure 1: Context specific indepencence

CSI



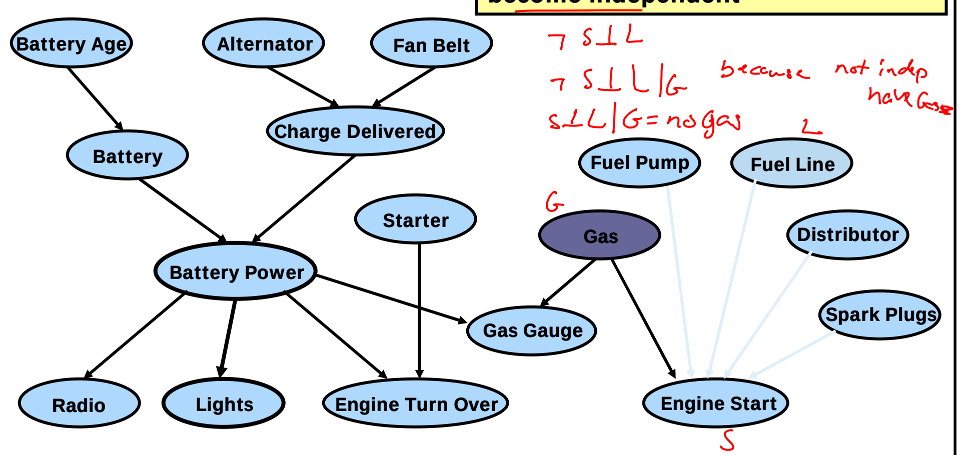
Local Structure 1: Context specific indepencence

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

$\neg\, S \perp L$

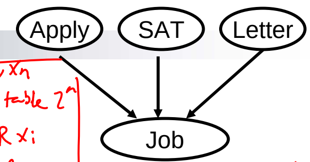
$\neg\, S \perp L \mid G$ because not indep have Gas

$S \perp L \mid G = \text{no gas}$



Representation

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

$X_1 \ldots X_n$, Y, table 2^n

$Y = OR\ X_i$

$O(n)$

$P(J \mid A, S, L)$ ← table

binary: 8 or 16 table

$P(J=t \mid A=f, S, L) = 0$

SL irrelevant

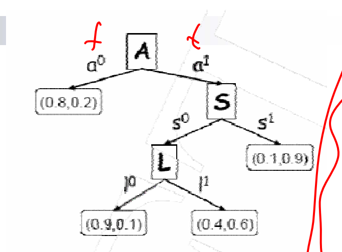
$P(J=t \mid A=t, S=low, L) = \epsilon = 0.1$

$J \perp \{S, L\} \mid A=f$

4 parents

Tabular VE with Tree CPDs

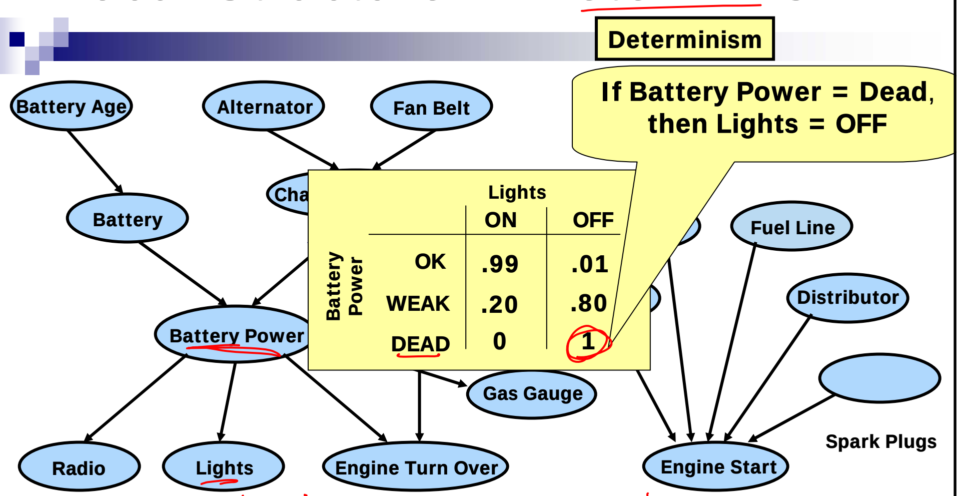
- If we turn a **tree CPD into table**
  - **“Sparsity” lost**!
- Need inference approach that **deals with tree CPD directly**!



$P(J \mid A, S, L)$

ASL	ttt	ttf	tft	tff	...
J=t	.9	.9			
J=f					

Local Structure 2: Determinism



$P(L|B)$ $P(L=t|B=dead)=0$

Determinism and inference

$P(x_i | Pa\,x_i)$ exp in # parents

- 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

$\prod_i f_i$

$g(y)$

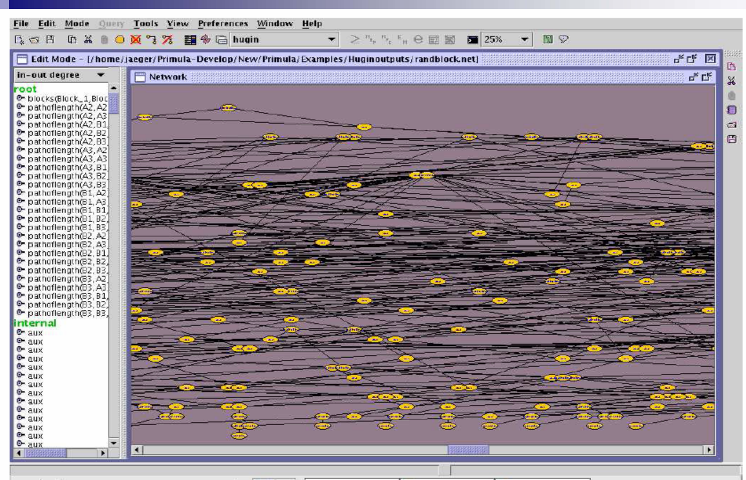
$f_i(x_i)=0$

all values of the $g(y)$ table where x_i will be zero

Today's Models …

- **Often characterized by:**
  - Richness in local structure (determinism, CSI)
  - Massiveness in size (10,000's variables)
  - High connectivity (treewidth) high
- **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…

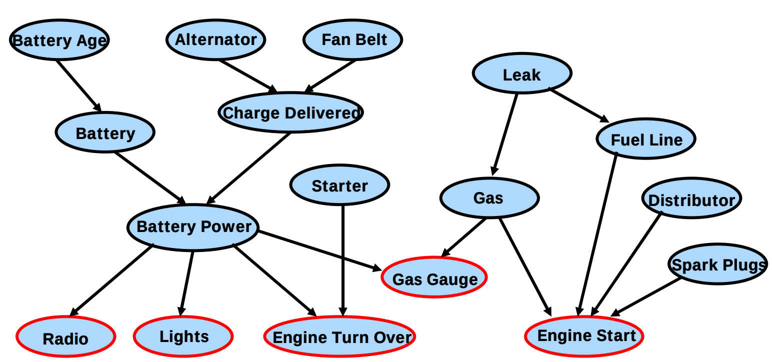


- BN from a relational model

Recursive Conditioning

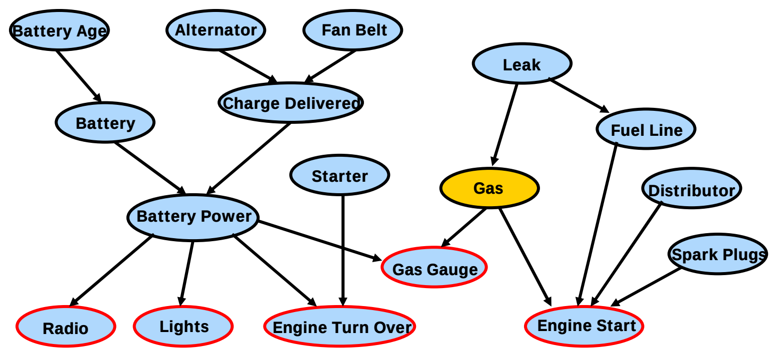
- Treewidth complexity (worst case)
- Better than treewidth complexity with local structure
- Provides a framework for time-space tradeoffs
- Only quick intuition today, details in readings

The Computational Power of Assumptions



A. Darwiche

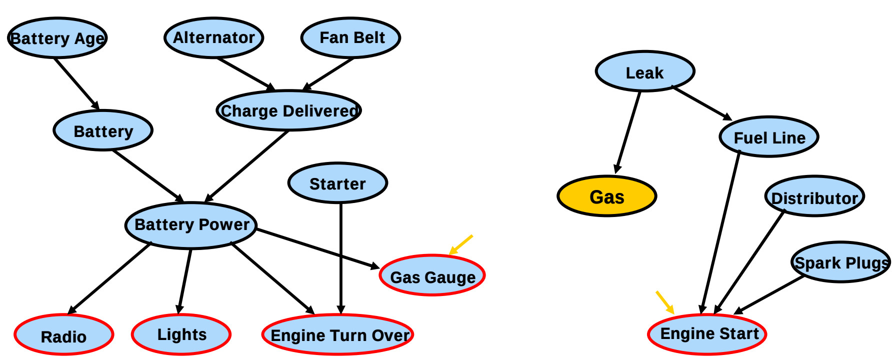
The Computational Power of Assumptions



A. Darwiche

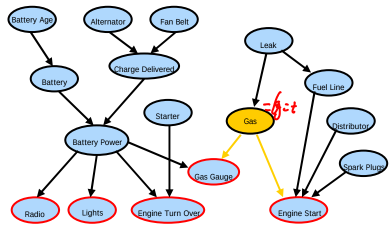
Decomposition

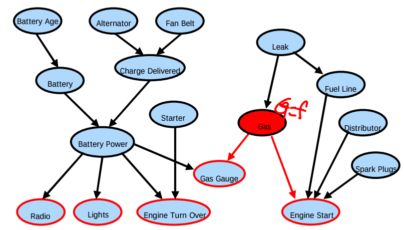
VE exploits this



A. Darwiche

Case Analysis

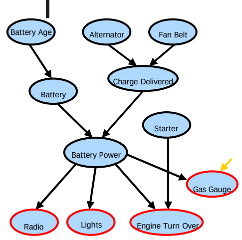


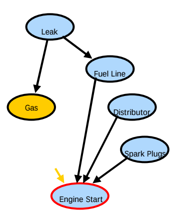


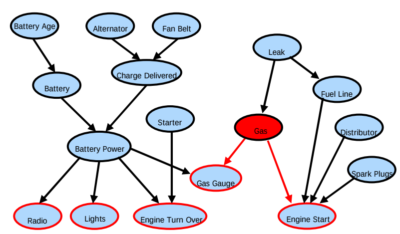
p + p

A. Darwiche

Case Analysis



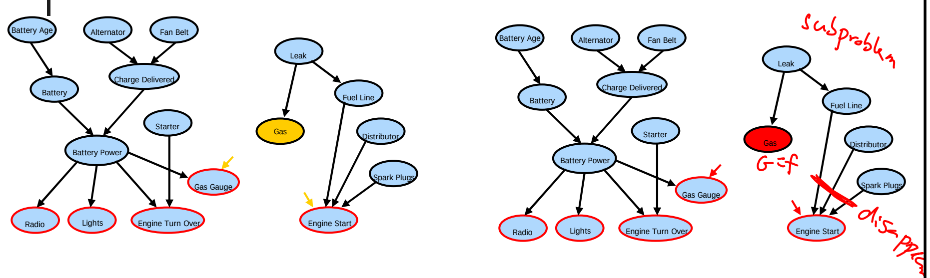




p_l * p_r + p

A. Darwiche

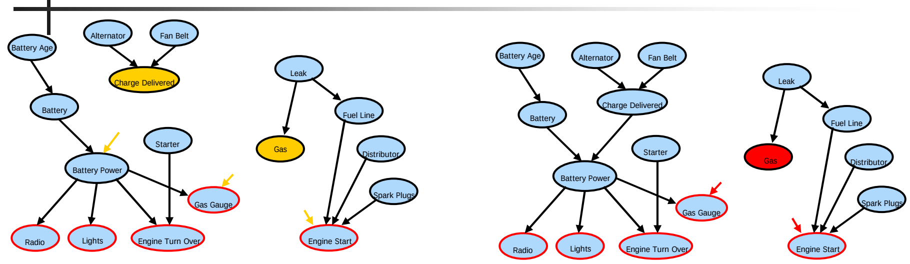
Case Analysis



p_l * p_r + p_l * p_r

A. Darwiche

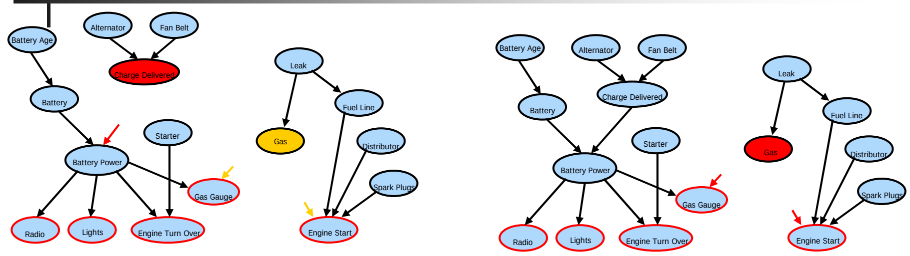
Case Analysis



p_l * p_r + p_l * p_r

A. Darwiche

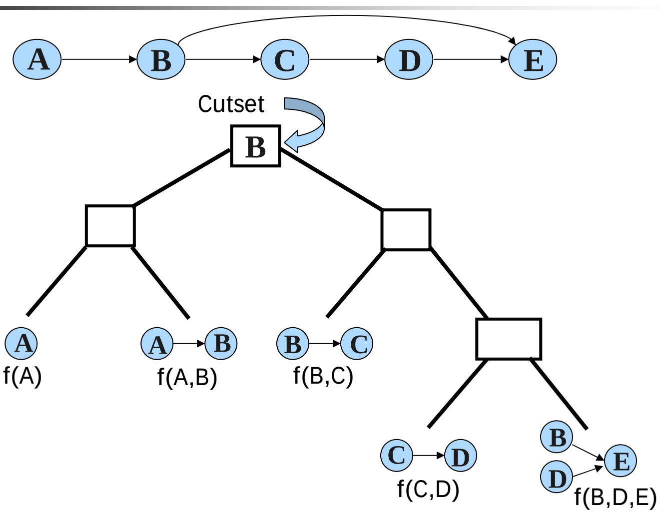
Case Analysis



p_l * p_r + p_l * p_r

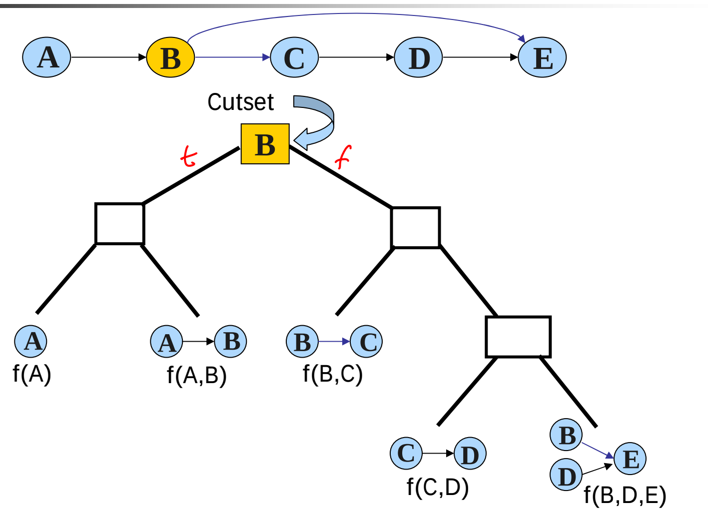
A. Darwiche

Decomposition Tree



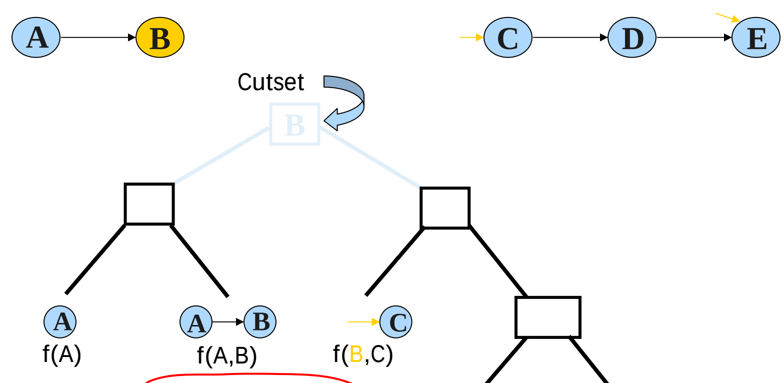
A. Darwiche

Decomposition Tree



A. Darwiche

Decomposition Tree



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

A. Darwiche

RC1

choosing T related to choosing VE order

```
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
```

look up prob. from CPTs

A. Darwiche

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**

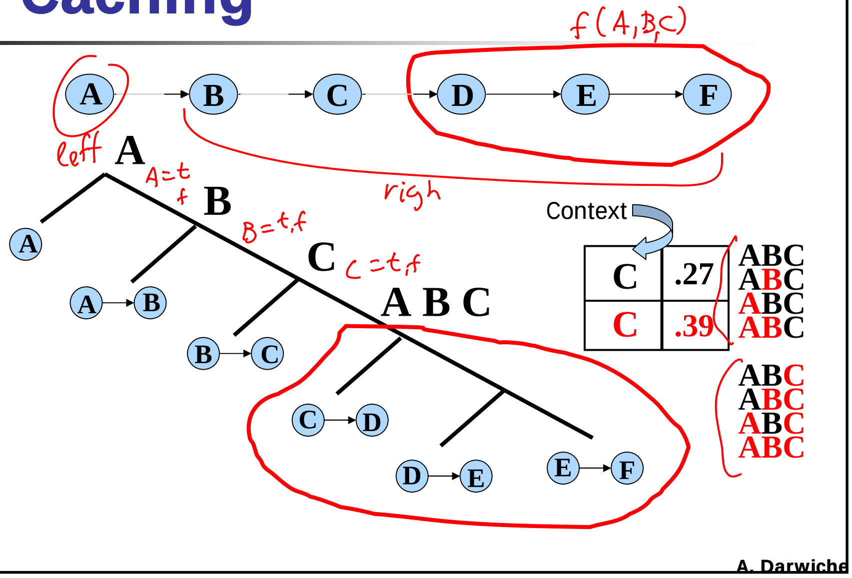
u: value of **U** in **e**

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

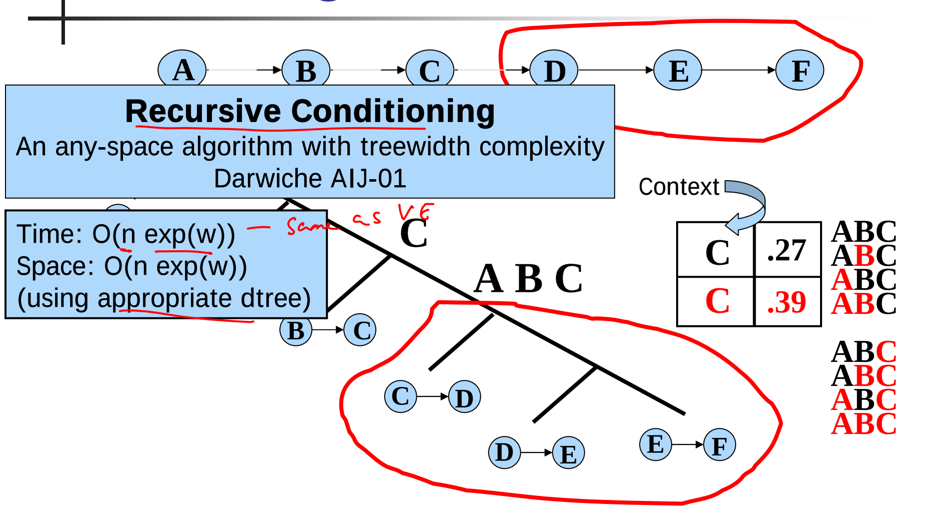
Else return 1 = $\Sigma_x\ \theta_{x|\mathbf{u}}$

A. Darwiche

Caching



Caching



Recursive Conditioning
An any-space algorithm with treewidth complexity
Darwiche AIJ-01

Time: O(n exp(w)) — same as VE
Space: O(n exp(w))
(using appropriate dtree)

RC2

RC2(T,**e**)

at leaf look at CPT

- If T is a leaf node, return Lookup(T,e)
- **y** := instantiation of context(T)
- If cache$_T$[**y**] <> nil, return cache$_T$[**y**]

or check cache

compute

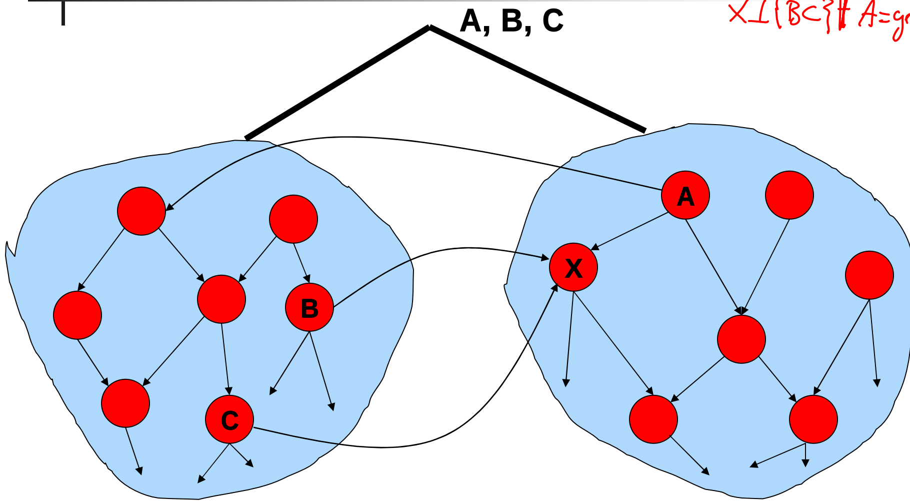
- p := 0
- For each instantiation **c** of cutset(T)-**E** do
  - p := p + RC2(T^l,**ec**) RC2(T^r,**ec**)
- cache$_T$[**y**] := p
- Return p

A. Darwiche

Decomposition with Local Structure

X Independent of B, C given $\boxed{A}$

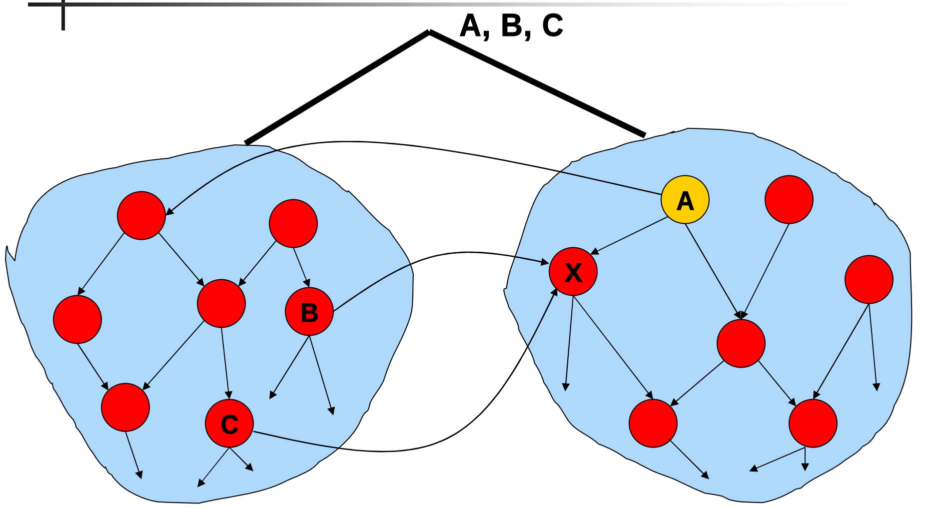
$X \perp \{BC\} \mid A$ = gelb



A. Darwiche

Decomposition with Local Structure

X Independent of B, C given A

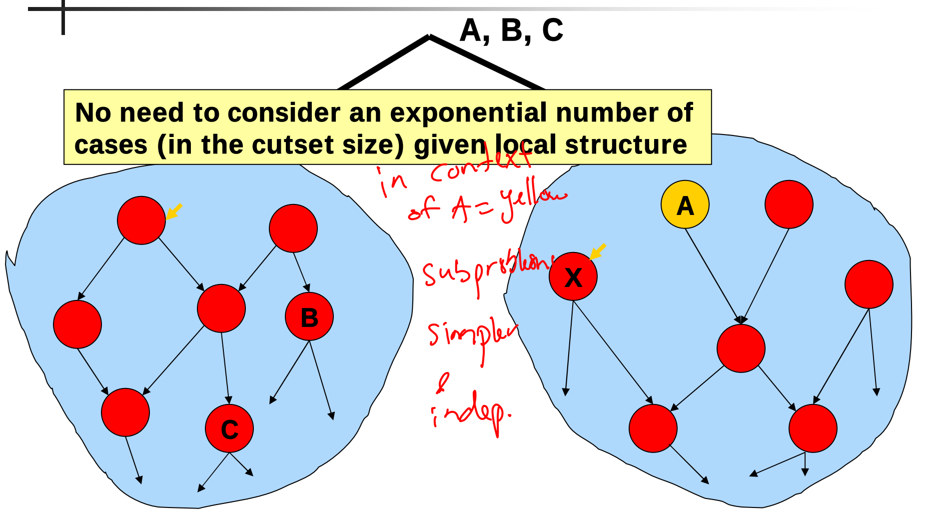


A. Darwiche

Decomposition with Local Structure

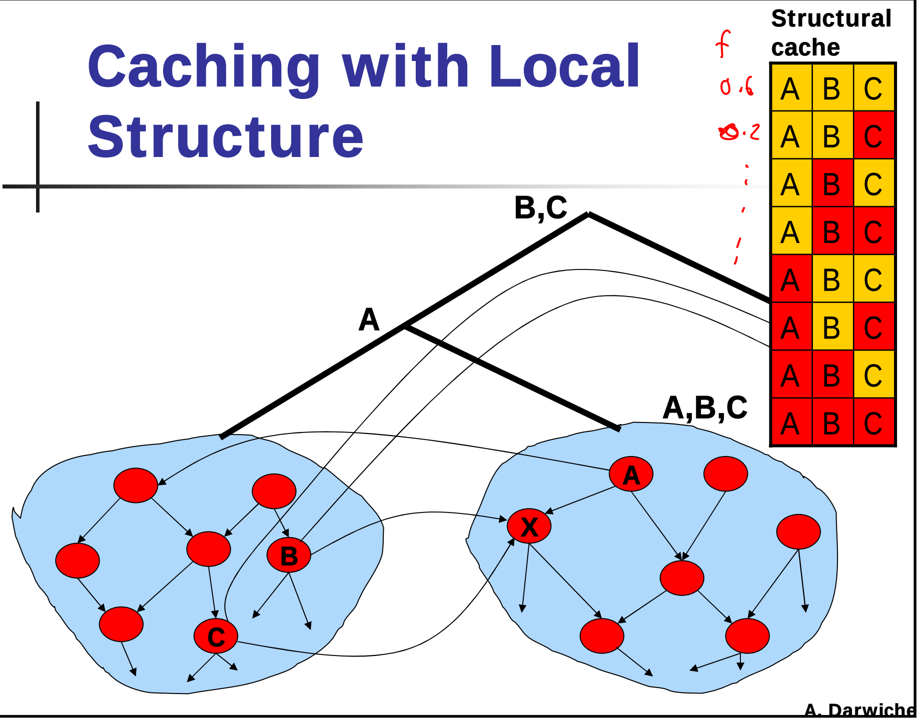
X Independent of B, C given A

No need to consider an exponential number of cases (in the cutset size) given local structure

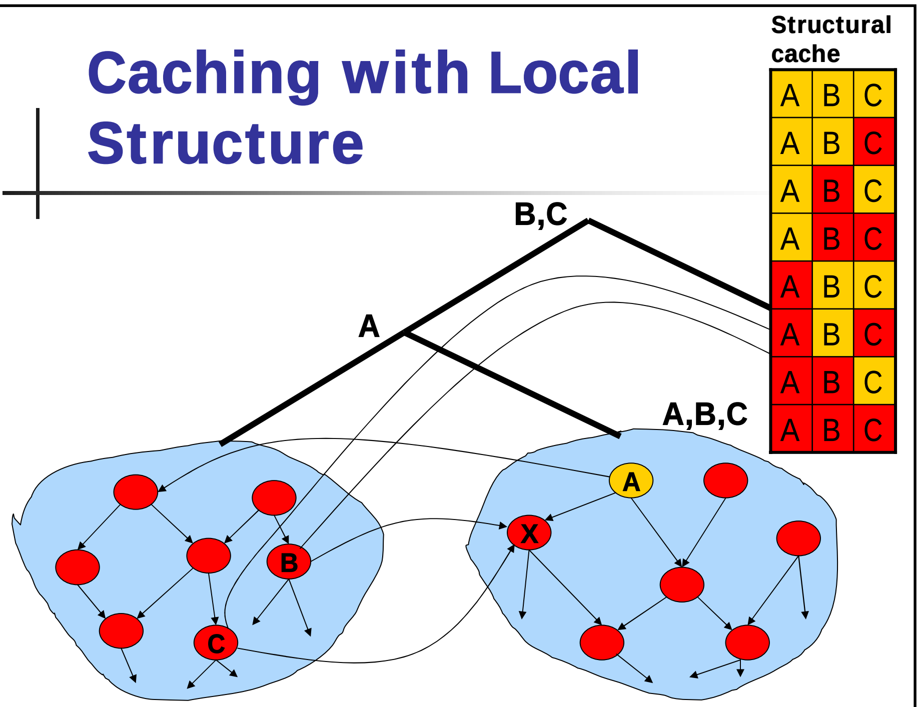


A. Darwiche

Caching with Local Structure

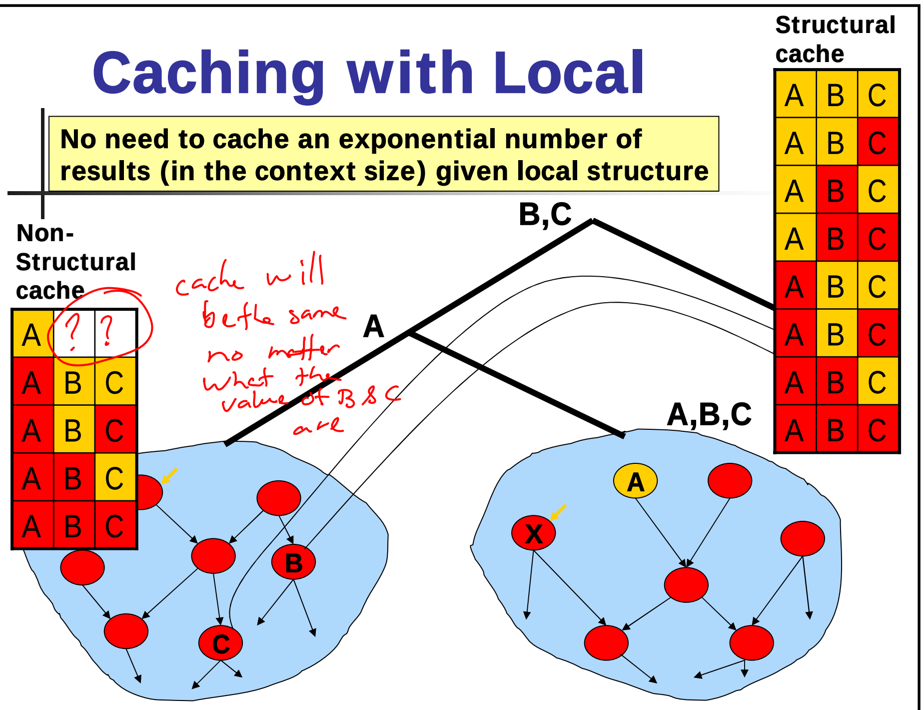


Caching with Local Structure



Caching with Local

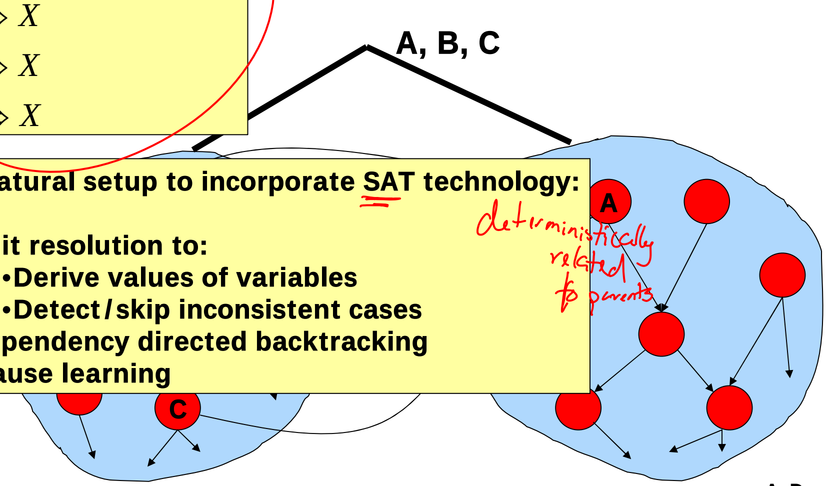
No need to cache an exponential number of results (in the context size) given local structure



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 inside CPT
  - 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