

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

Graphical Models – 10708

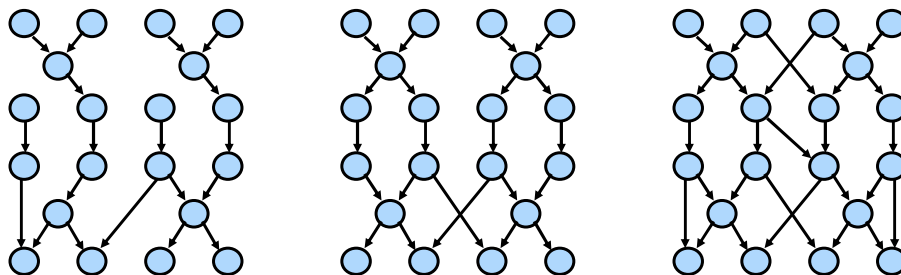
Carlos Guestrin

Carnegie Mellon University

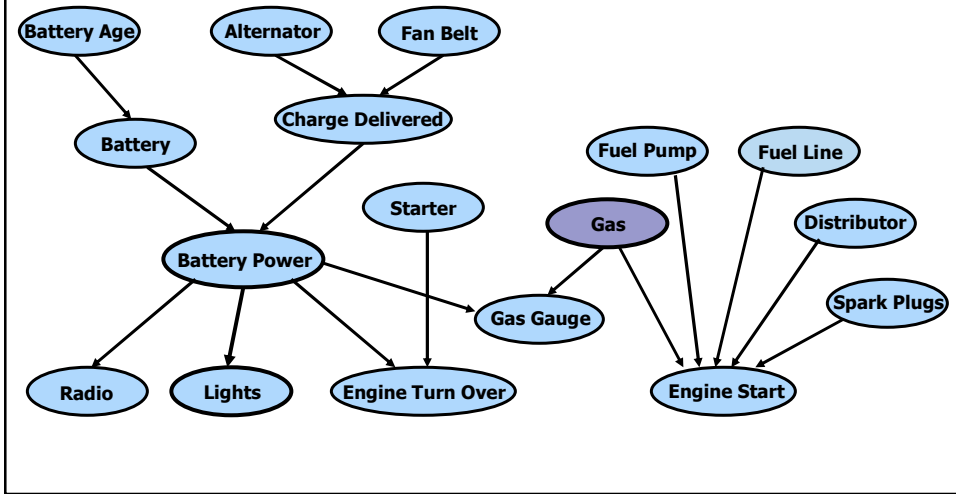
October 16th, 2006

Global Structure: Treewidth w

$O(n \exp(w))$

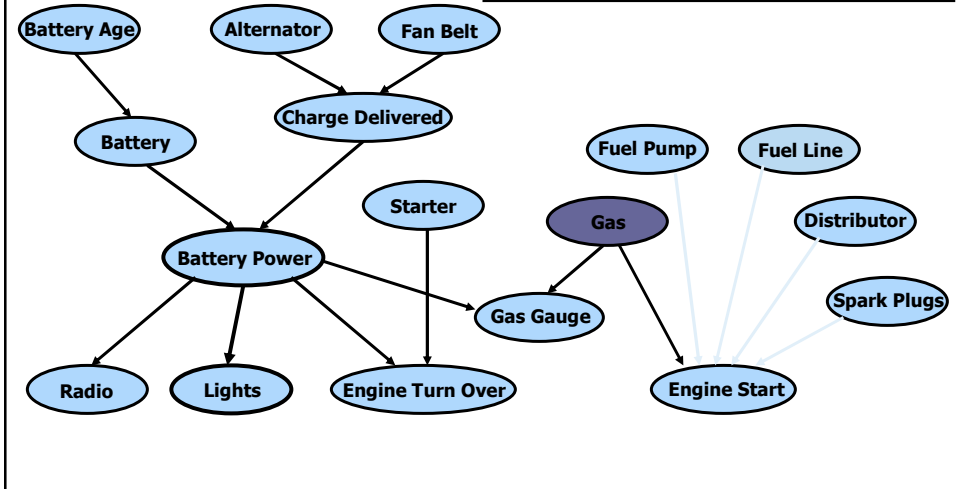


Local Structure 1: Context specific independence

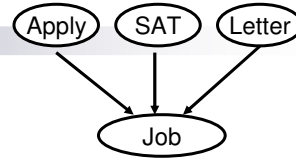


Local Structure 1: Context specific independence

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



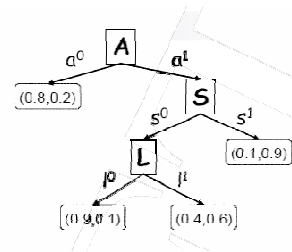
CSI example: Tree CPD



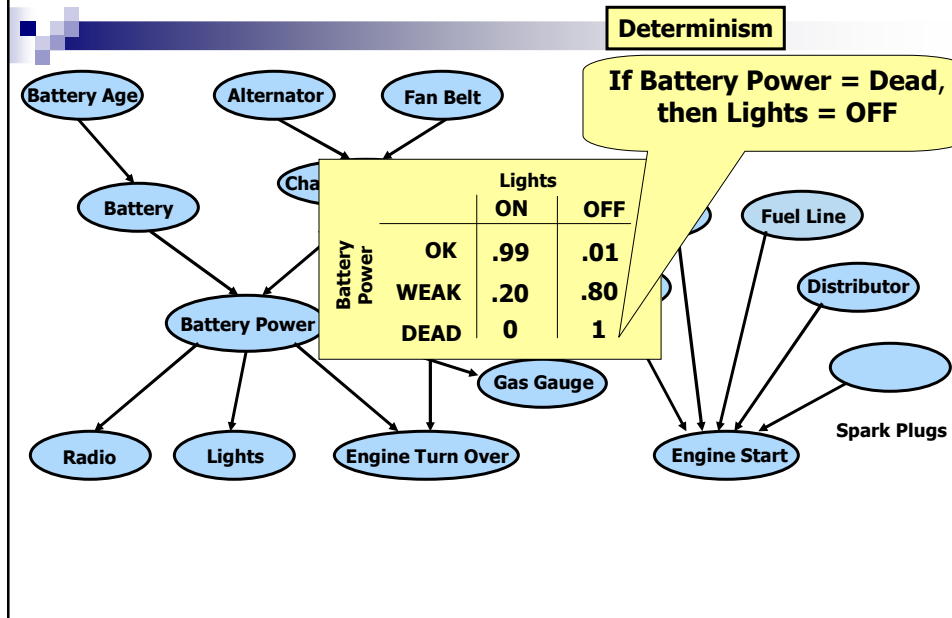
- Represent $P(X_i | \mathbf{Pa}_{X_i})$ using a decision tree
 - Path to leaf is an assignment to (a subset of) \mathbf{Pa}_{X_i}
 - Leaves are distributions over X_i given assignment of \mathbf{Pa}_{X_i} on path to leaf
- **Interpretation of leaf:**
 - For specific assignment of \mathbf{Pa}_{X_i} on 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!**



Local Structure 2: Determinism



Determinism and inference

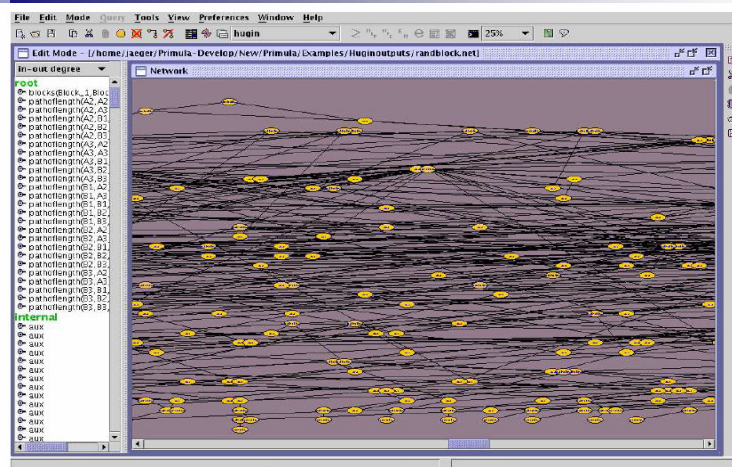
- Determinism gives a little sparsity in table, but much bigger impact on inference
- Multiplying deterministic factor with other factor introduces many new zeros
 - Operations related to theorem proving, e.g., unit resolution

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

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...

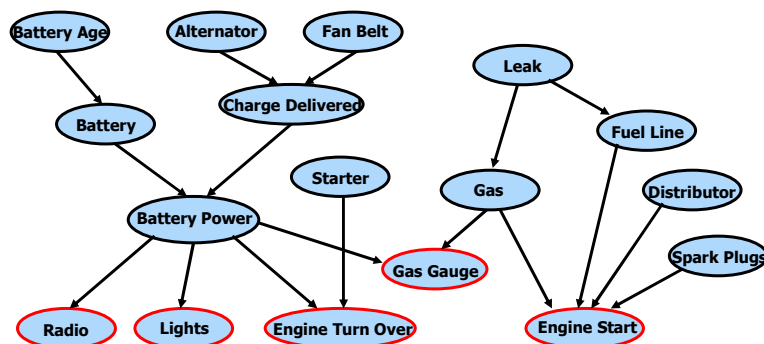


- BN from a relational model

Recursive Conditioning

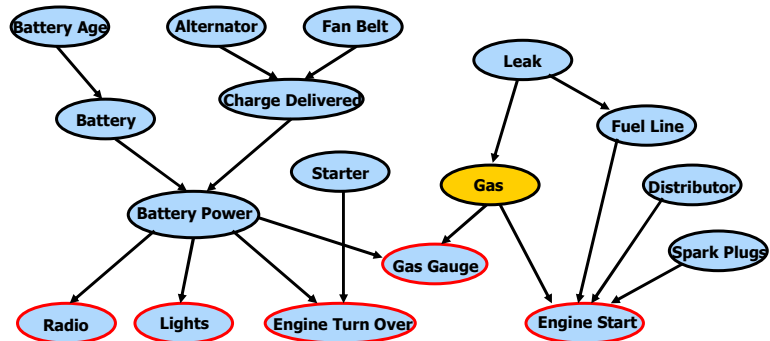
- Treewidth complexity (worst case)
- Better than treewidth complexity with local structure
- Provides a framework for time-space tradeoffs
- Only quick intuition today, details in readings

The Computational Power of Assumptions



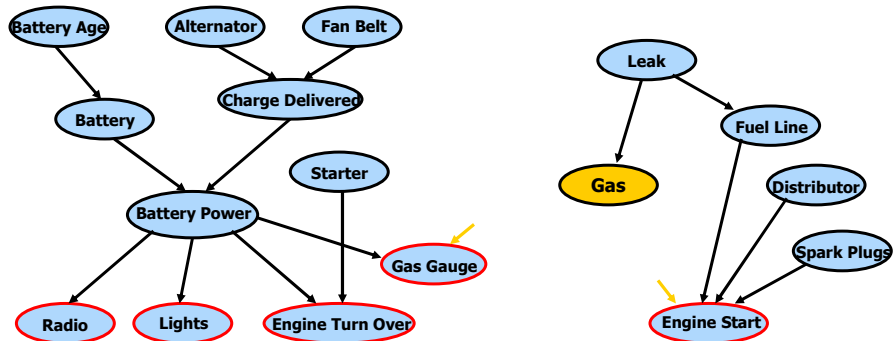
A. Darwiche

The Computational Power of Assumptions



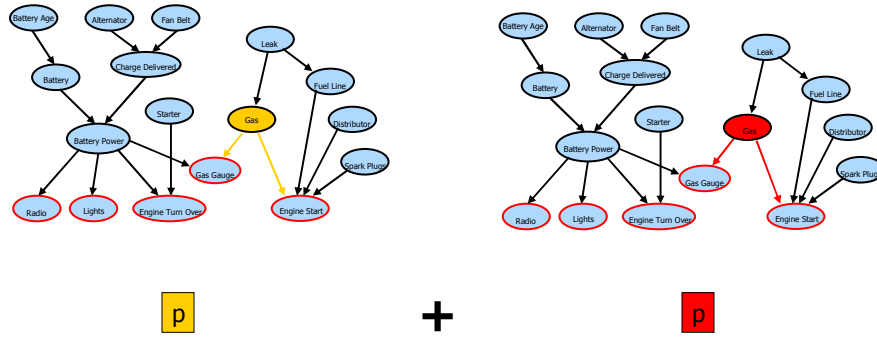
A. Darwiche

Decomposition



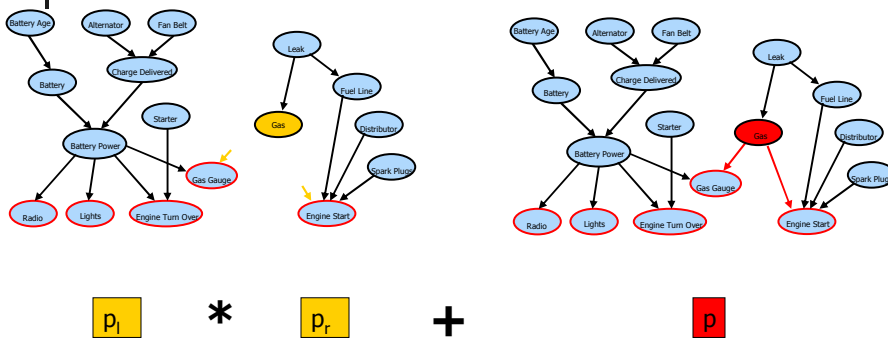
A. Darwiche

Case Analysis



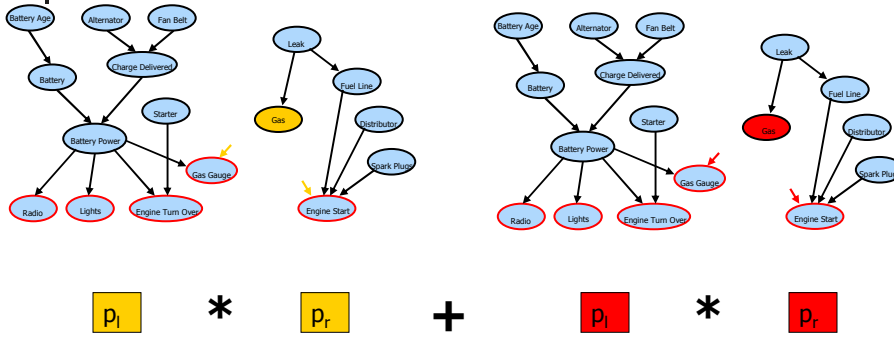
A. Darwiche

Case Analysis



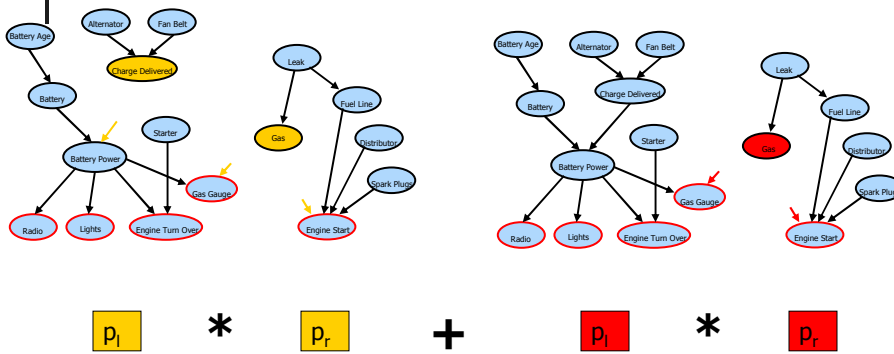
A. Darwiche

Case Analysis



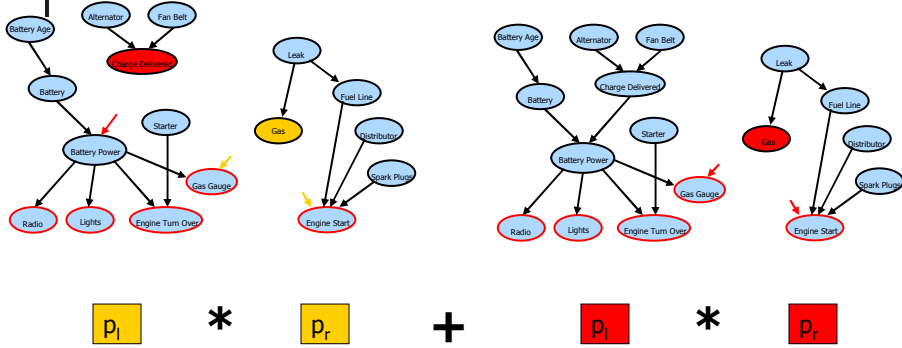
A. Darwiche

Case Analysis



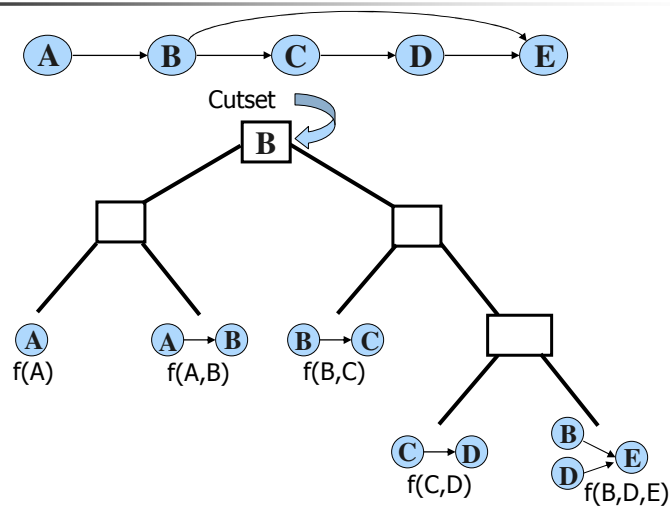
A. Darwiche

Case Analysis



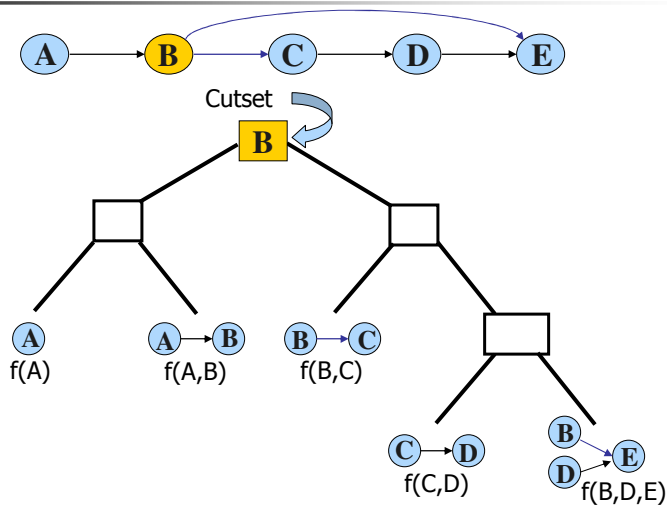
A. Darwiche

Decomposition Tree



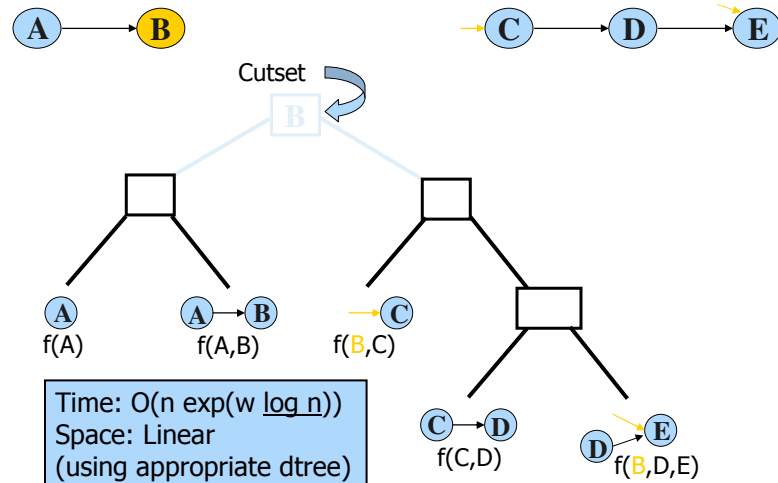
A. Darwiche

Decomposition Tree



A. Darwiche

Decomposition Tree



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

A. Darwiche

RC1

```
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
```

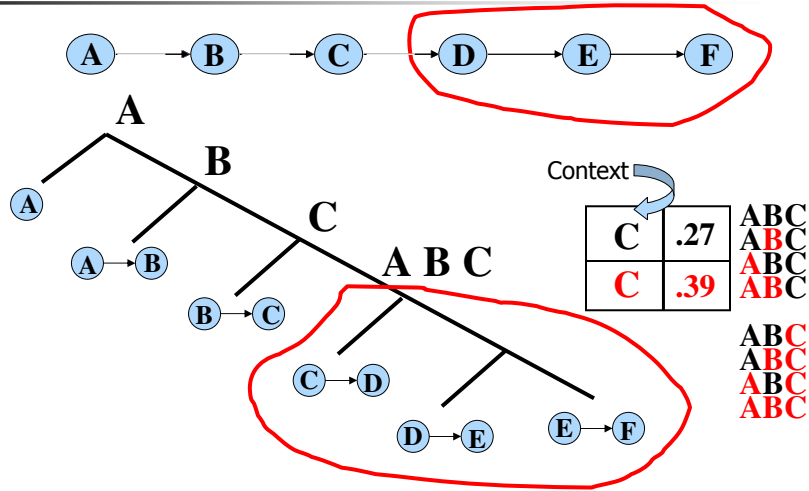
A. Darwiche

Lookup(T,e)

```
 $\Theta_{X|U}$  : 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  $\theta_{x|u}$ 
Else return  $1 = \sum_x \theta_{x|u}$ 
```

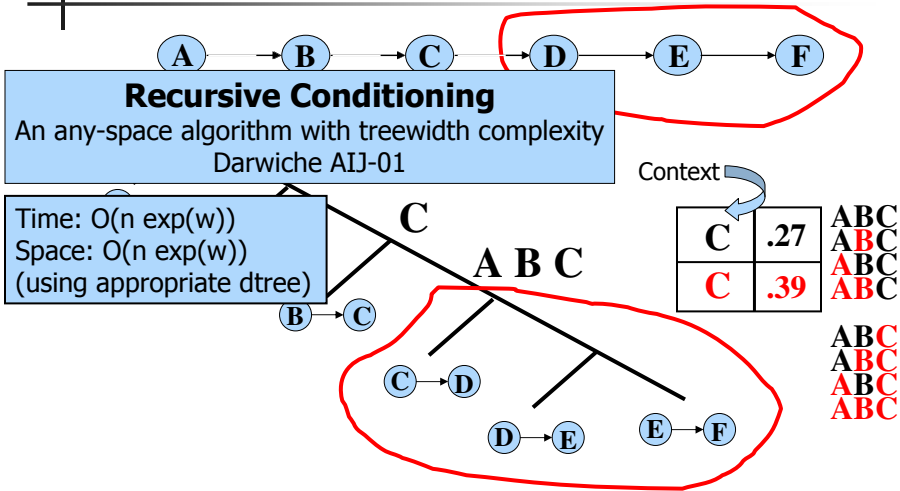
A. Darwiche

Caching



A. Darwiche

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)

A. Darwiche

RC2

RC2(T,e)

If T is a leaf node, return $\text{Lookup}(T,e)$

$\mathbf{y} :=$ instantiation of $\text{context}(T)$

If $\text{cache}_T[\mathbf{y}] \neq \text{nil}$, return $\text{cache}_T[\mathbf{y}]$

$p := 0$

For each instantiation \mathbf{c} of $\text{cutset}(T)-\mathbf{E}$ do

$p := p + \text{RC2}(T',\mathbf{ec})$

$\text{cache}_T[\mathbf{y}] := p$

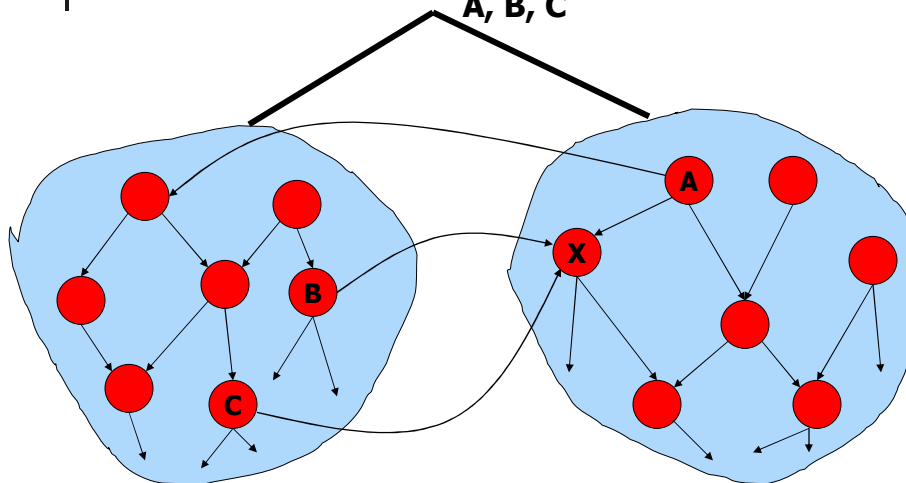
Return p

A. Darwiche

Decomposition with Local Structure

X Independent of B, C given A

A, B, C

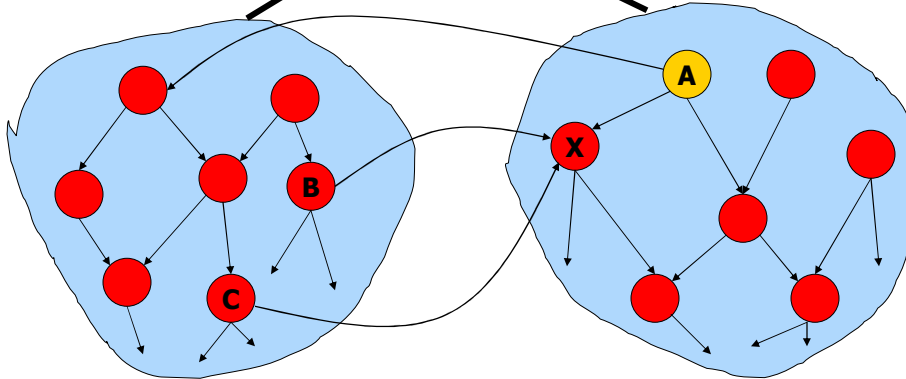


A. Darwiche

Decomposition with Local Structure

X Independent of B, C given A

A, B, C



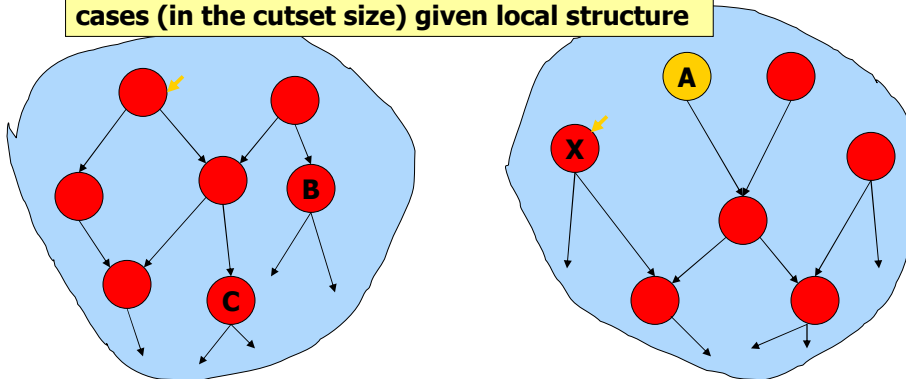
A. Darwiche

Decomposition with Local Structure

X Independent of B, C given A

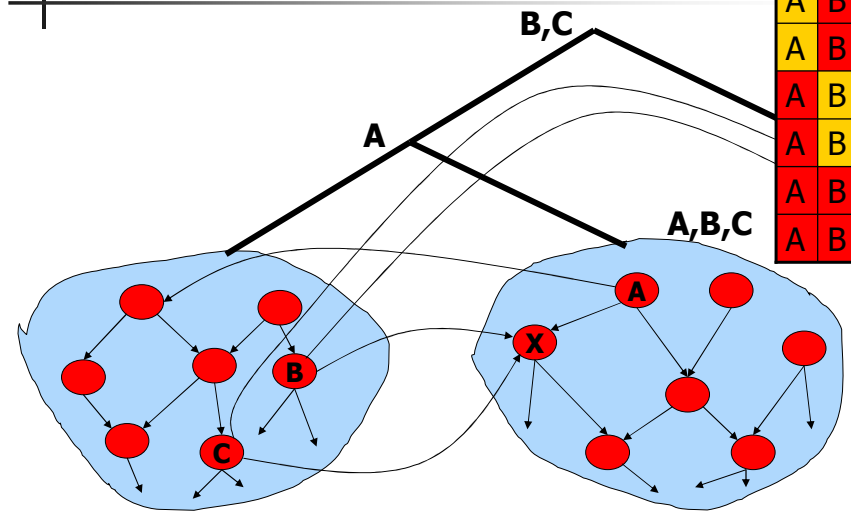
A, B, C

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



A. Darwiche

Caching with Local Structure

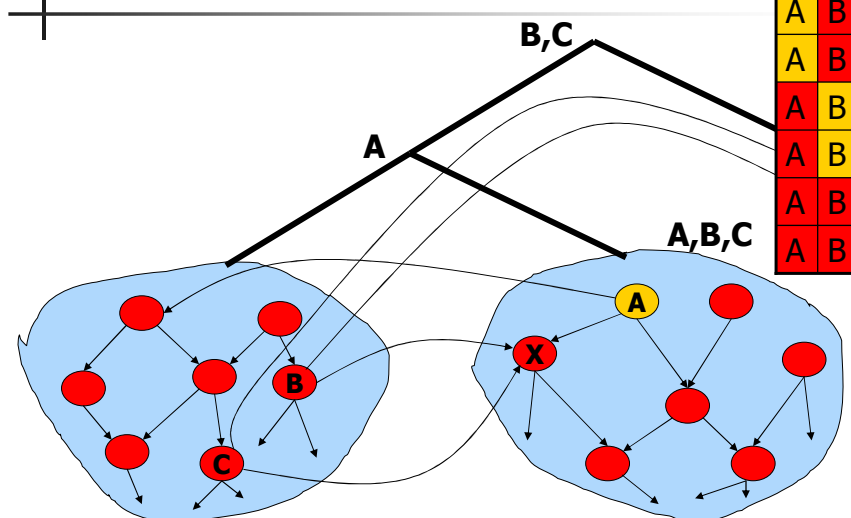


Structural cache

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

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

A. Darwiche

Caching with Local

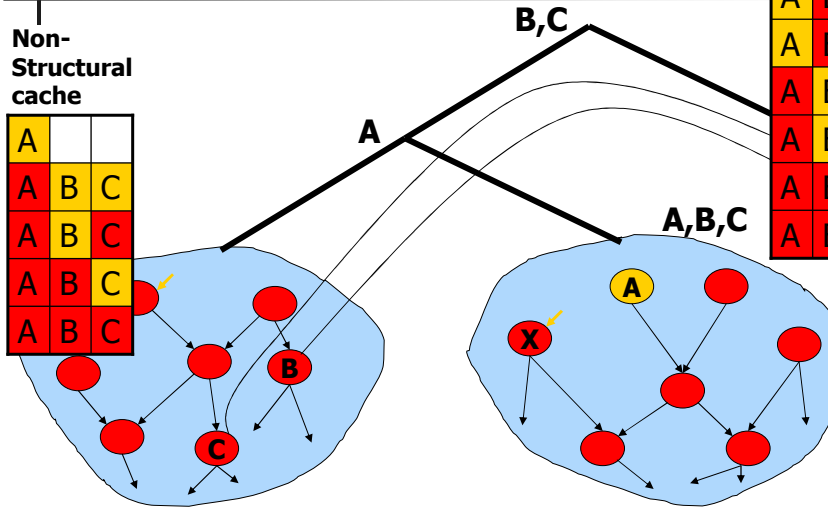
No need to cache an exponential number of results (in the context size) given local structure

Non-Structural cache

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

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
A	B	C
A	B	C



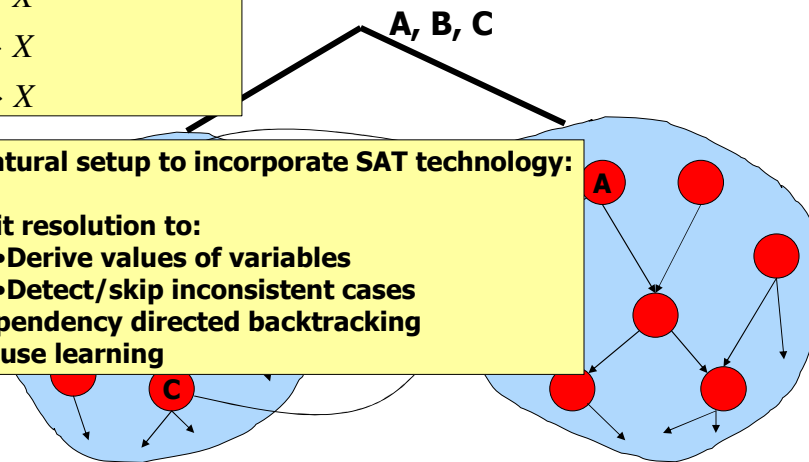
A. Darwiche

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>