Remainder of talk

- Why we've lost no information
- Computational Economics
- Memory requirements of v2.0 *vs* v1.0
- Version 3.0
- Using AD-Trees for
  - Bayes Net structure finding
  - Feature Selection
  - Rule Learning

Notation Reminder

ct ($a_{i(1)}$, $a_{i(2)}$, ... $a_{i(n)}$) is the contingency table for attribute-set { $a_{i(1)}$, $a_{i(2)}$, ... $a_{i(n)}$ }.

ct ($a_{i(1)}, a_{i(2)}, .. a_{i(n)} \mid a_{j(1)} = v_1, a_{j(2)} = v_2, ... a_{j(p)} = v_p$) is a conditional contigency table: It’s the contingency table for $a_{i(1)}, a_{i(2)}, .. a_{i(n)}$ restricted to those records that match

$$a_{j(1)} = v_1,\ a_{j(2)} = v_2,\ ... a_{j(p)} = v_p$$

a_1	a_2
1	2
2	1
2	1
2	2
4	1
4	2
4	2
4	2

Dataset

ct (a_1, a_2)

a_1	a_2	Count
1	1	0
1	2	1
2	1	2
2	2	1
3	1	0
3	2	0
4	1	1
4	2	3

ct (a_2)

a_2	Count
1	3
2	5

ct ($a_2 \mid a_1 = 1$)

a_2	Count
1	0
2	1

ct ($a_2 \mid a_1 = 2$)

a_2	Count
1	2
2	1

ct ($a_2 \mid a_1 = 3$)

a_2	Count
1	0
2	0

ct ($a_2 \mid a_1 = 4$)

a_2	Count
1	1
2	3

Contingency Algebra

d

a_1	a_2
1	2
2	1
2	1
2	2
4	1
4	2
4	2
4	2

Dataset

a_2	*Count*
1	*0*
2	*1*

$\mathbf{ct}(a_2 \mid a_1=1)$

+

a_2	*Count*
1	*2*
2	*1*

$\mathbf{ct}(a_2 \mid a_1=2)$

+

a_2	*Count*
1	*0*
2	*0*

$\mathbf{ct}(a_2 \mid a_1=3)$

+

a_2	*Count*
1	*1*
2	*3*

$\mathbf{ct}(a_2 \mid a_1=4)$

=

a_2	*Count*
1	*3*
2	*5*

$\mathbf{ct}(a_2)$

Contingency Algebra

a_1	a_2
1	2
2	1
2	1
2	2
4	1
4	2
4	2
4	2

Dataset

$\mathbf{ct}(a_1, a_2) =$

$\mathbf{a}_1$	
1	$\mathbf{ct}(a_2 \mid a_1 = 1)$
2	$\mathbf{ct}(a_2 \mid a_1 = 2)$
3	$\mathbf{ct}(a_2 \mid a_1 = 3)$
4	$\mathbf{ct}(a_2 \mid a_1 = 4)$

a_1	a_2	*Count*
1	1	*0*
1	2	*1*
2	1	*2*
2	2	*1*
3	1	*0*
3	2	*0*
4	1	*1*
4	2	*3*

=

$\mathbf{a}_1$ = 1:

a_2	*Count*
1	*0*
2	*1*

$\mathbf{a}_1$ = 2:

a_2	*Count*
1	*2*
2	*1*

$\mathbf{a}_1$ = 3:

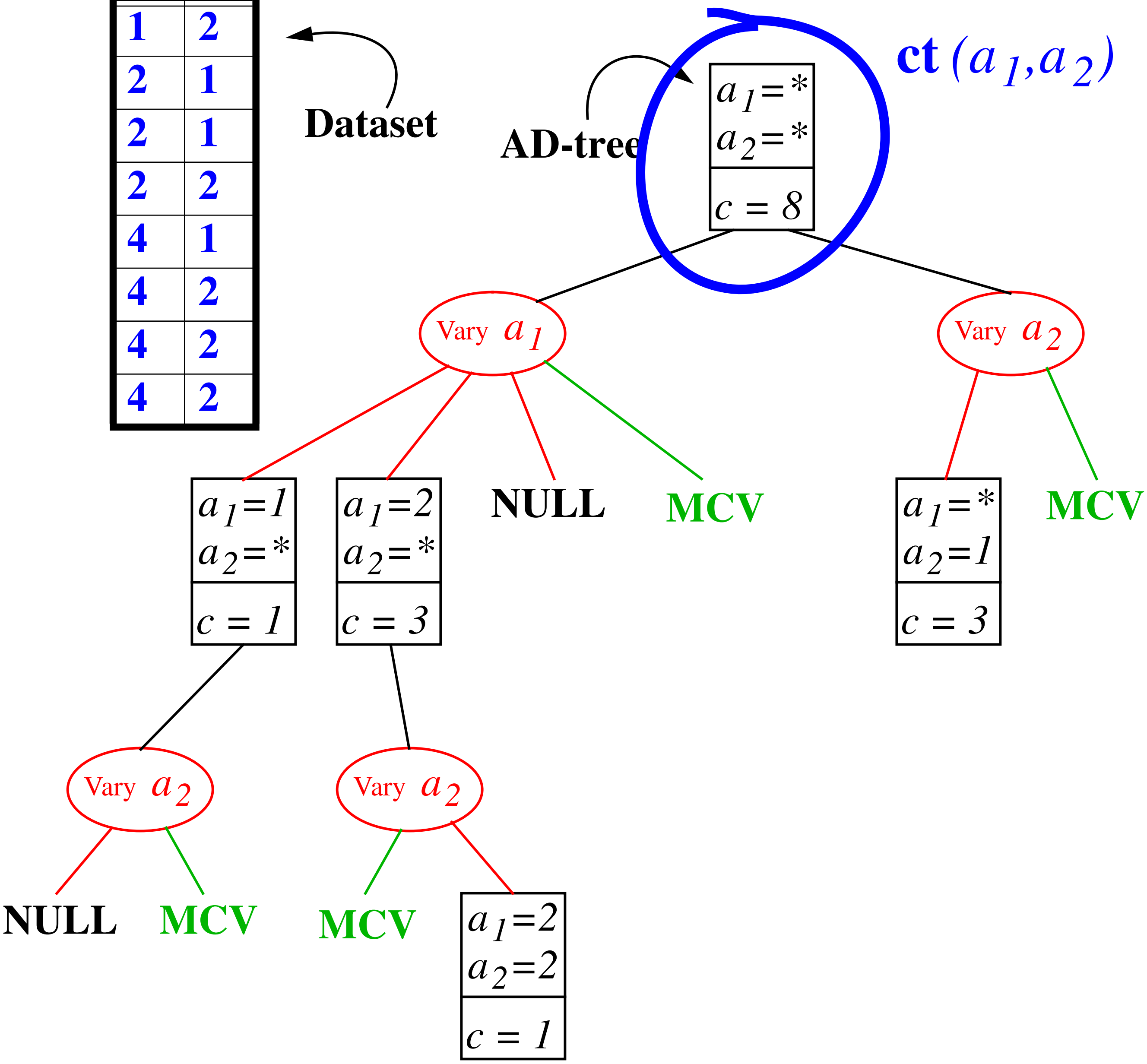
a_2	*Count*
1	*0*
2	*0*

$\mathbf{a}_1$ = 4:

a_2	*Count*
1	*1*
2	*3*

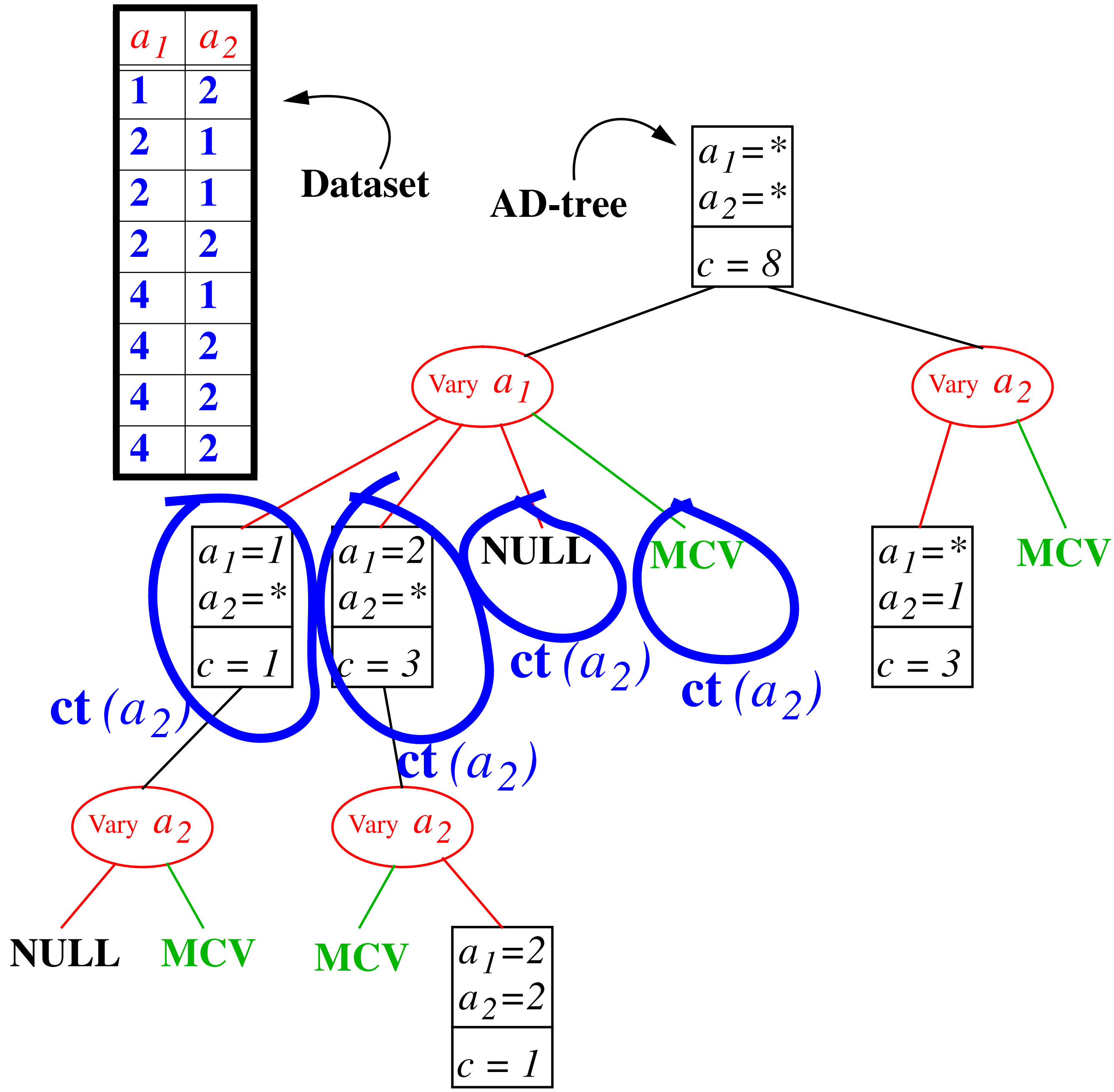
Computing ct(a_1,a_2)

a_1	a_2
1	2
2	1
2	1
2	2
4	1
4	2
4	2
4	2



To compute **ct** (a_1,a_2)....

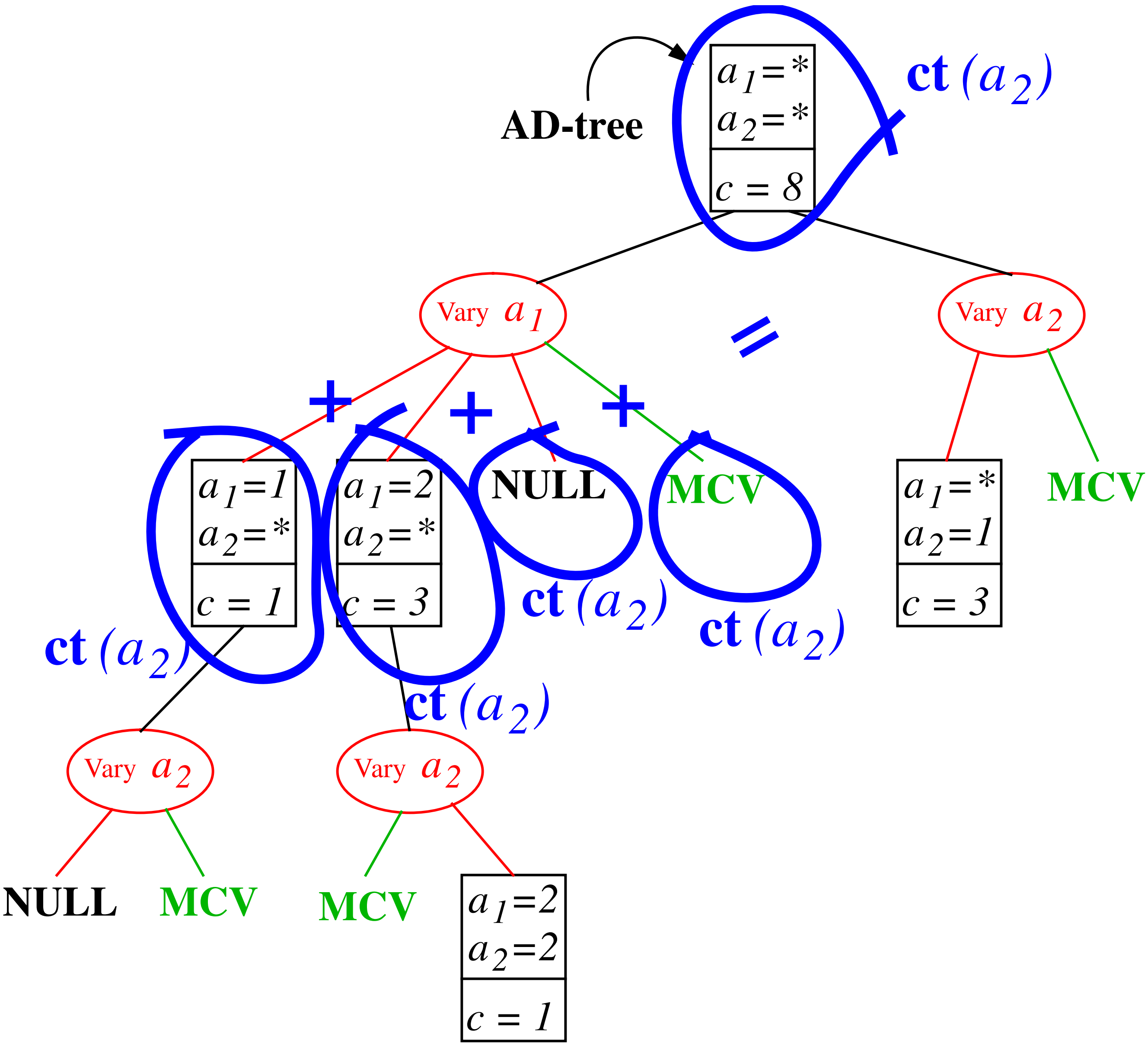
Computing ct(a_1,a_2)



To compute **ct** (a_1,a_2)....

Compute **ct** (a_2|a_1=1), **ct** (a_2|a_1=2), **ct** (a_2|a_1=3) and **ct** (a_2|a_1=4) and concatenate them...

Computing ct(a_1,a_2)



To compute **ct** (a_1,a_2)....

- Compute **ct** (a_2)
- Compute **ct** $(a_2|a_1=1)$, **ct** $(a_2|a_1=2)$, **ct** $(a_2|a_1=3)$
- **ct** $(a_2|a_1=4)$ = **ct** (a_2)-**ct** $(a_2|a_1=1)$-**ct** $(a_2|a_1=2)$-**ct** $(a_2|a_1=3)$
- and concatenate them...

General Case...

MakeCT $(\{ a_{i(1)}\ a_{i(2)} .. a_{i(n)} \} , ADN)$

Let VN := The *Vary* $a_{i(1)}$ subnode of ADN.

Let $MCV := VN.MCV$

For $k := 1 , 2 , \ldots , n_{i(1)}$

If $k\ != MCV$

ADN_k := the $a_{i(1)}=k$ subnode of VN

CT_k := **MakeCT** $(\{a_{i(2)} .. a_{i(n)}\} ,ADN_k)$

Let CT_{base} = **MakeCT** $(\{a_{i(2)} .. a_{i(n)}\} , ADN)$

$CT_{MCV} := CT_{base} - \Sigma_{(k!=MCV)}\ CT_k$

Return concatenation of $CT_1\ CT_2 \ldots CT_{ni(1)}$

Computational Economics

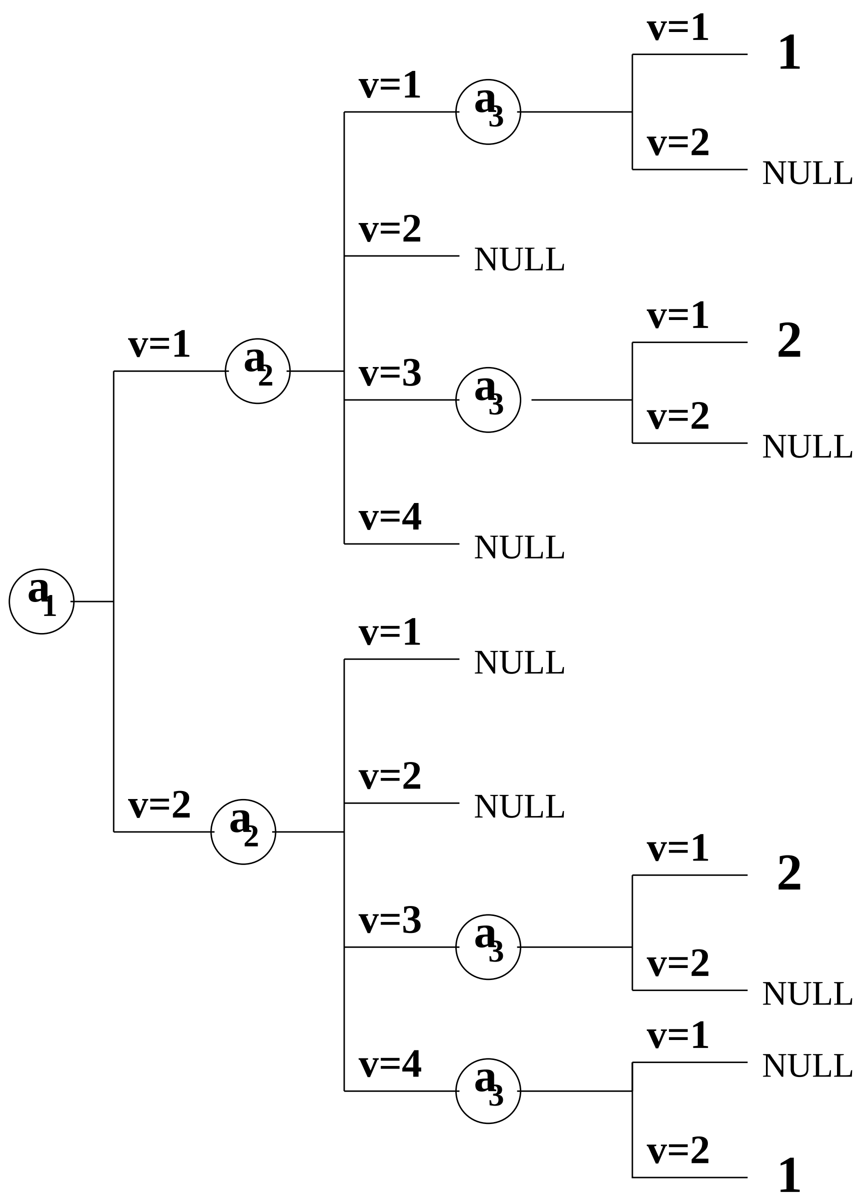
Cost of building a Contingency Table for n attributes given R records. (Worst Case) (Assuming Binary attributes) (sorry about all these parentheses)

Method	Cost
Direct Counting Approach	$O(2^n + R)$
AD-Tree Version 1.1 *(MCV nodes are non-NULL)*	$O(2^n)$
AD-Tree Version 2.0 *(MCV nodes are NULL)*	$O(n2^n)$

Sparse Contingency T ables

$ct(a_1,a_2,a_3)$

a_1	a_2	a_3	#
1	1	1	1
1	1	2	0
1	2	1	0
1	2	2	0
1	3	1	2
1	3	2	0
1	4	1	0
1	4	2	0
2	1	1	0
2	1	2	0
2	2	1	0
2	2	2	0
2	3	1	2
2	3	2	0
2	4	1	0
2	4	2	1



If n is large , the worst case bounds are usually muc h too pessimistic , because contingency tables can be implemented as trees instead of arra ys….

Counting General Case...

count $(\{a_{i(1)}=v_1\ a_{i(2)}=v_2\ ..a_{i(n)}=v_n\}, ADN)$

Let VN := The *Vary* $a_{i(1)}$ subnode of ADN.

if $v_1 = VN.MCV$

Let c_{base} := **count** $(\{a_{i(2)}=v_2.\ a_{i(n)}=v_n\},ADN)$

For $k := 1, 2, \ldots, n_{i(1)}$

If $k \mathrel{!}= MCV$

ADN_k := the $a_{i(1)}=k$ subnode of VN

c_k:=**count** $(\{a_{i(2)}=v_2.\ a_{i(n)}=v_n\},ADN_k)$

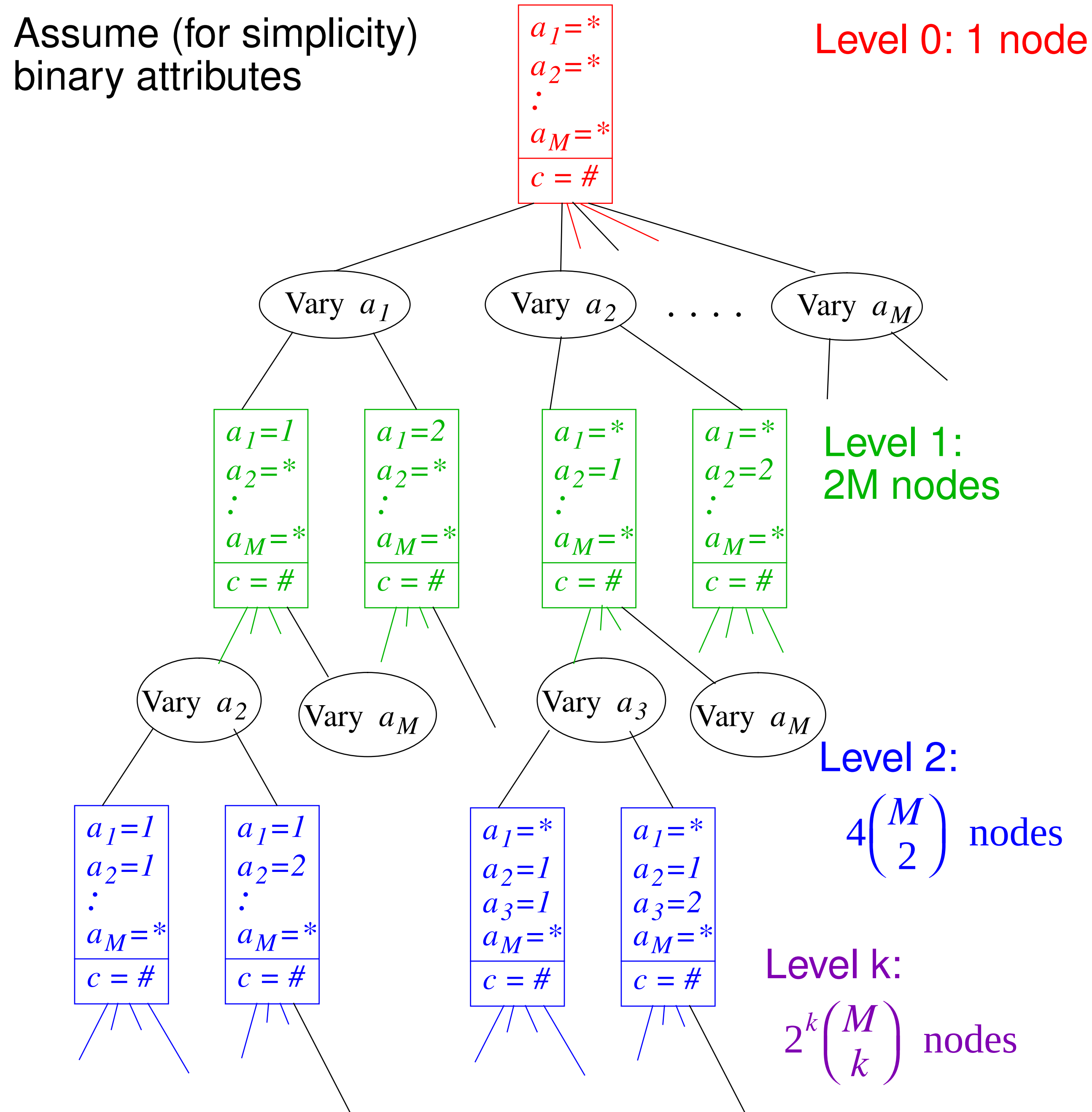
return $c_{base} - \Sigma_{(k!=MCV)}\ c_k$

else

Let ADN_{v1} = the $a_{i(1)}=v_1$ subnode of VN

return **count** $(\{a_{i(2)}=v_2.\ a_{i(n)}=v_n\},ADN_{v1})$

Reminder: Ad-Tree v1.1 Size

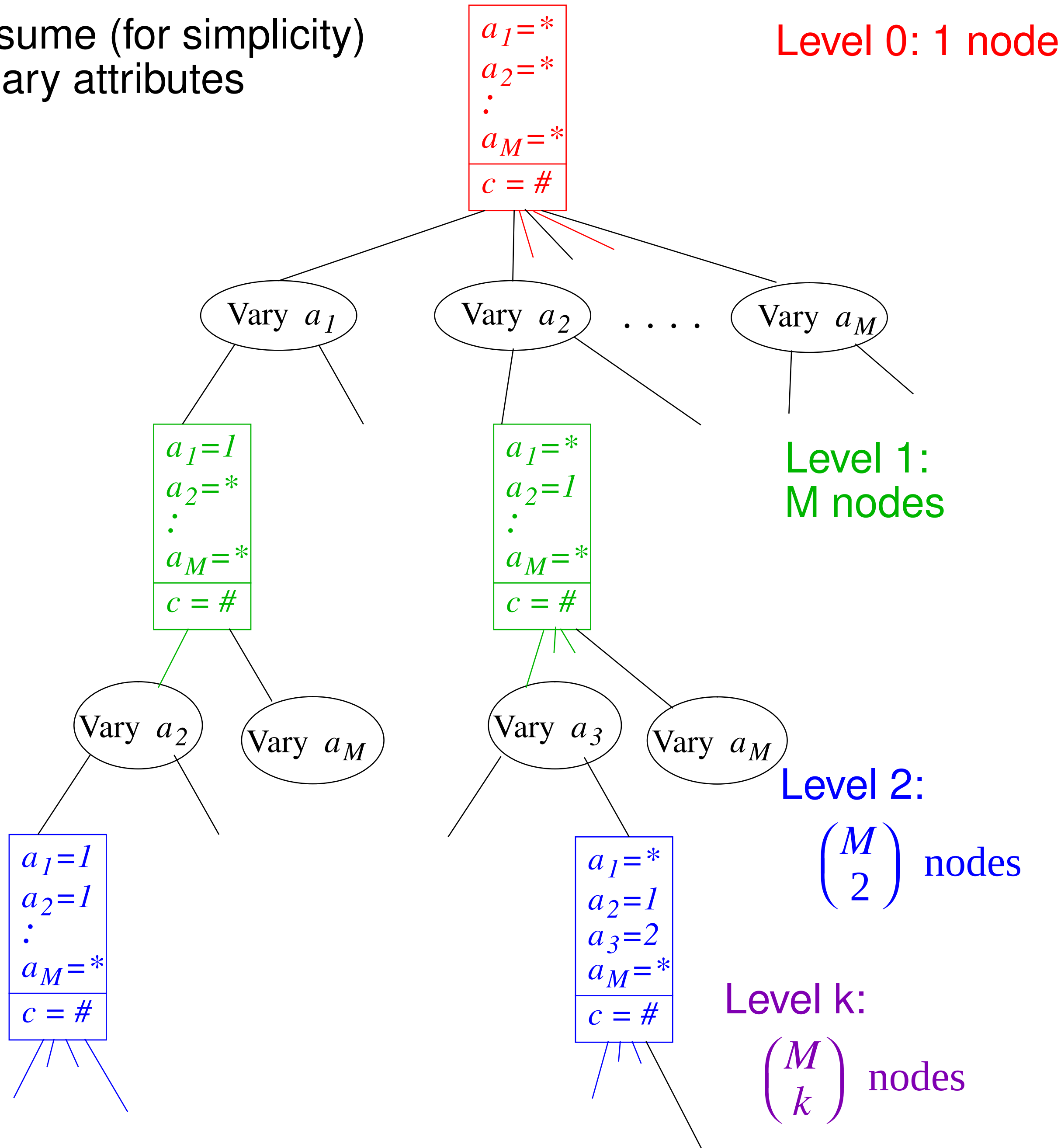


Number nodes is 3^M in the worst case. But if $R < 2^M$ then (in best case) number of nodes is about

$$\sum_{k=0}^{B} 2^k \binom{M}{k} \quad \text{where} \quad B = \log_2 R$$

Ad-Tree v2.0 Size

Assume (for simplicity) binary attributes

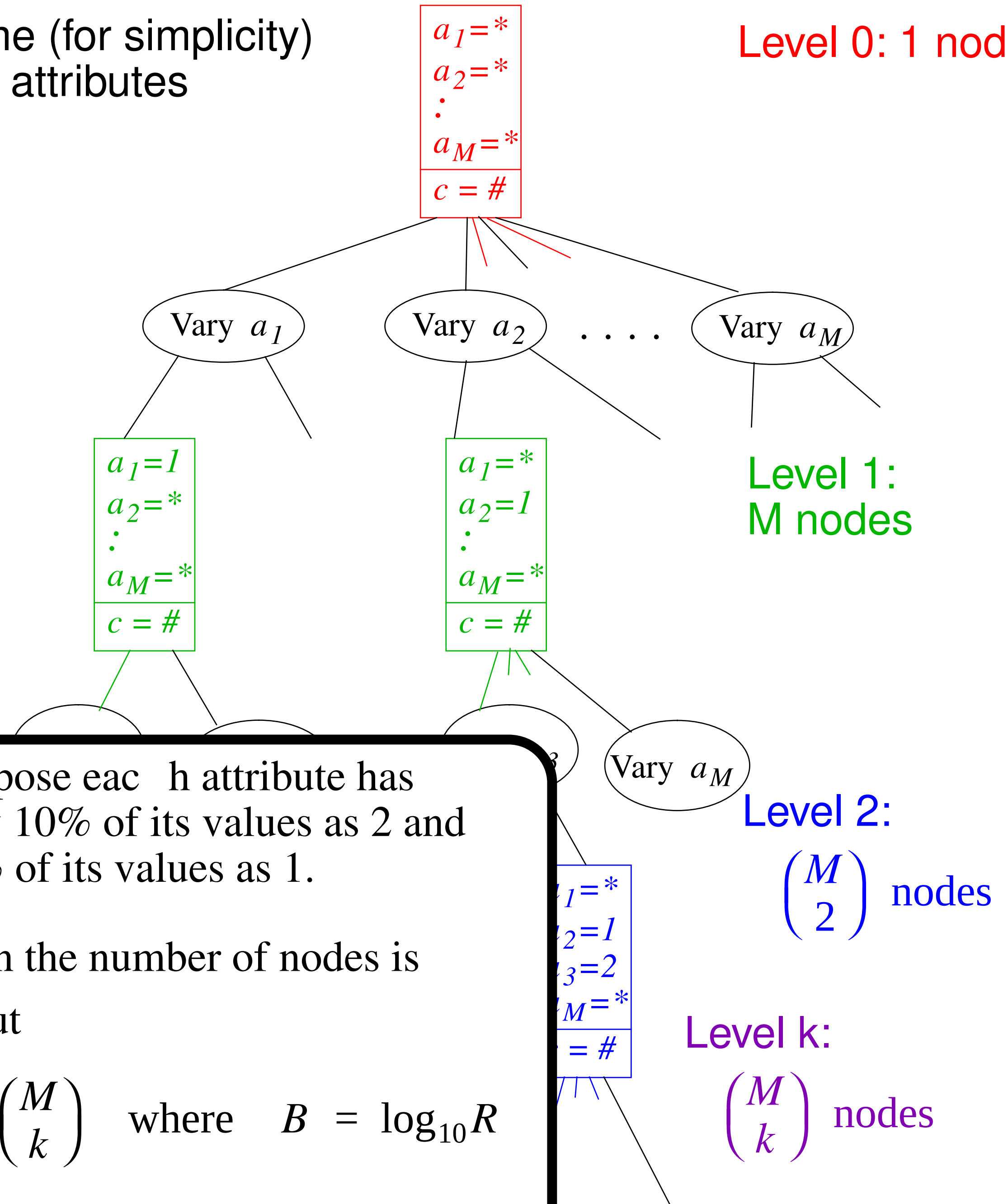


Number nodes is 2^M in the worst case. But if $R < 2^M$ then (in worst case) number of nodes is about

$$\sum_{k=0}^{B} \binom{M}{k} \quad \text{where} \quad B = \log_2 R$$

AD-Tree size: Sparse Atts

Assume (for simplicity) binary attributes



Suppose each attribute has only 10% of its values as 2 and 90% of its values as 1.

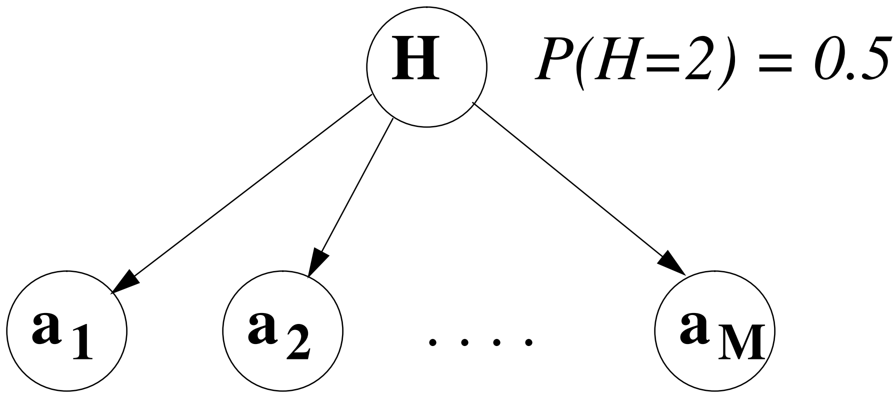
Then the number of nodes is about

$$\sum_{k=0}^{B} \binom{M}{k} \quad \text{where} \quad B = \log_{10} R$$

AD-Tree size: Correlated Atts

A
b

evel 0: 1 node



$P(a_i = H) = 0.95$

evel 1:
1 nodes

Vary a_M

Level 2:

$\binom{M}{2}$ nodes

Level k:

$\binom{M}{k}$ nodes

Then the number of nodes is about

$$\sum_{k=0}^{B} \binom{M}{k} \quad \text{where} \quad B = \log_4 R$$

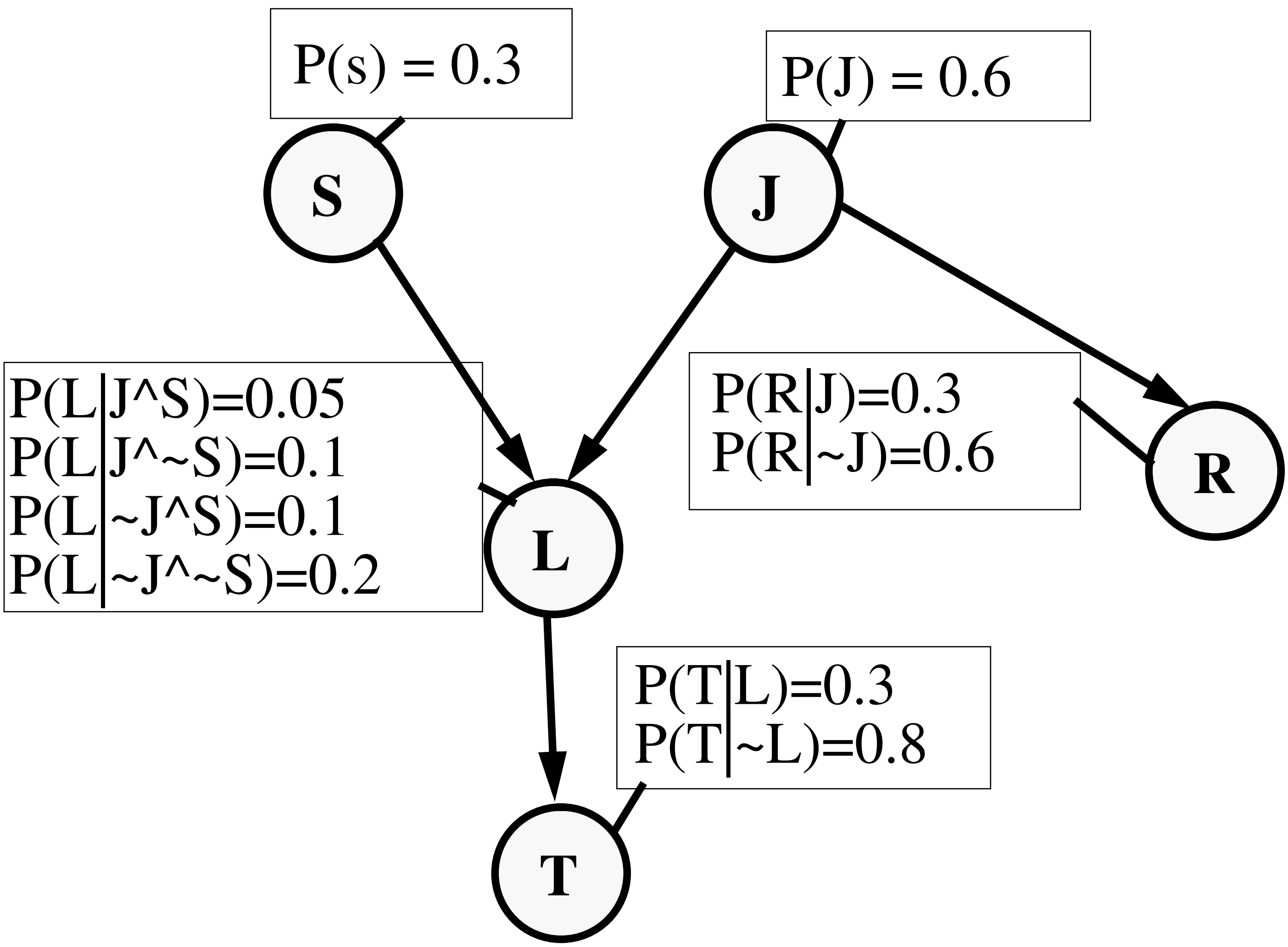
This is achieved with attributes that have uniform marginals.

Correlated Attributes: an extreme case

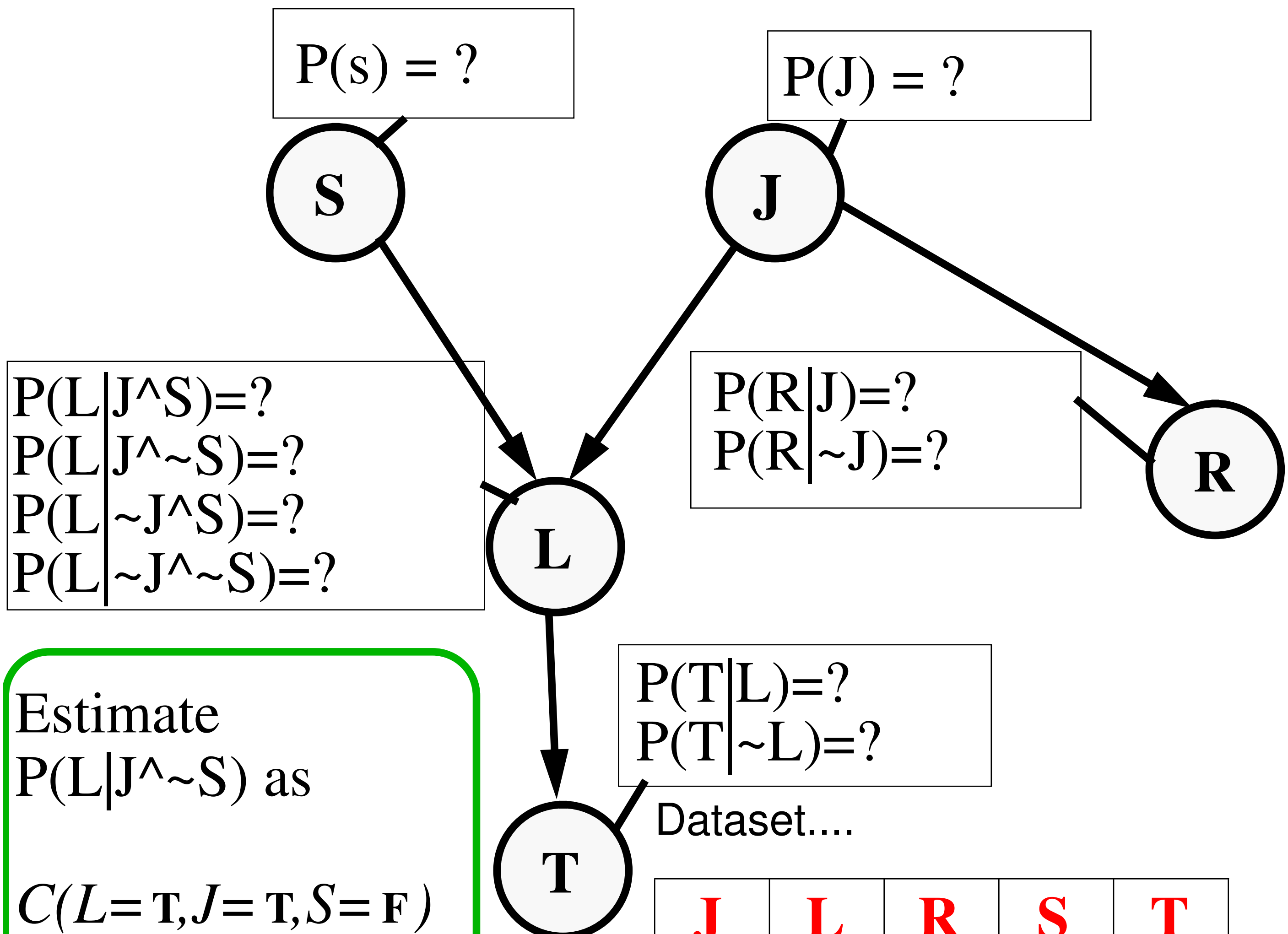
a_1	a_2	a_3	. . . .	a_{100}
1	1	1	....	1
1	1	1		1
1	1	1		1
1	1	1		1
1	1	1		1
1	1	1	....	1
1	1	1		1
2	2	2		2
2	2	2		2
2	2	2		2
2	2	2	....	2

- ADTree V ersion 1.0 has 2^{201} nodes .
- ADTree V ersion 2.0 has 101 nodes

Applying AD-Trees to learning Bayes Networks



Estimating Bayes Network Probabilities



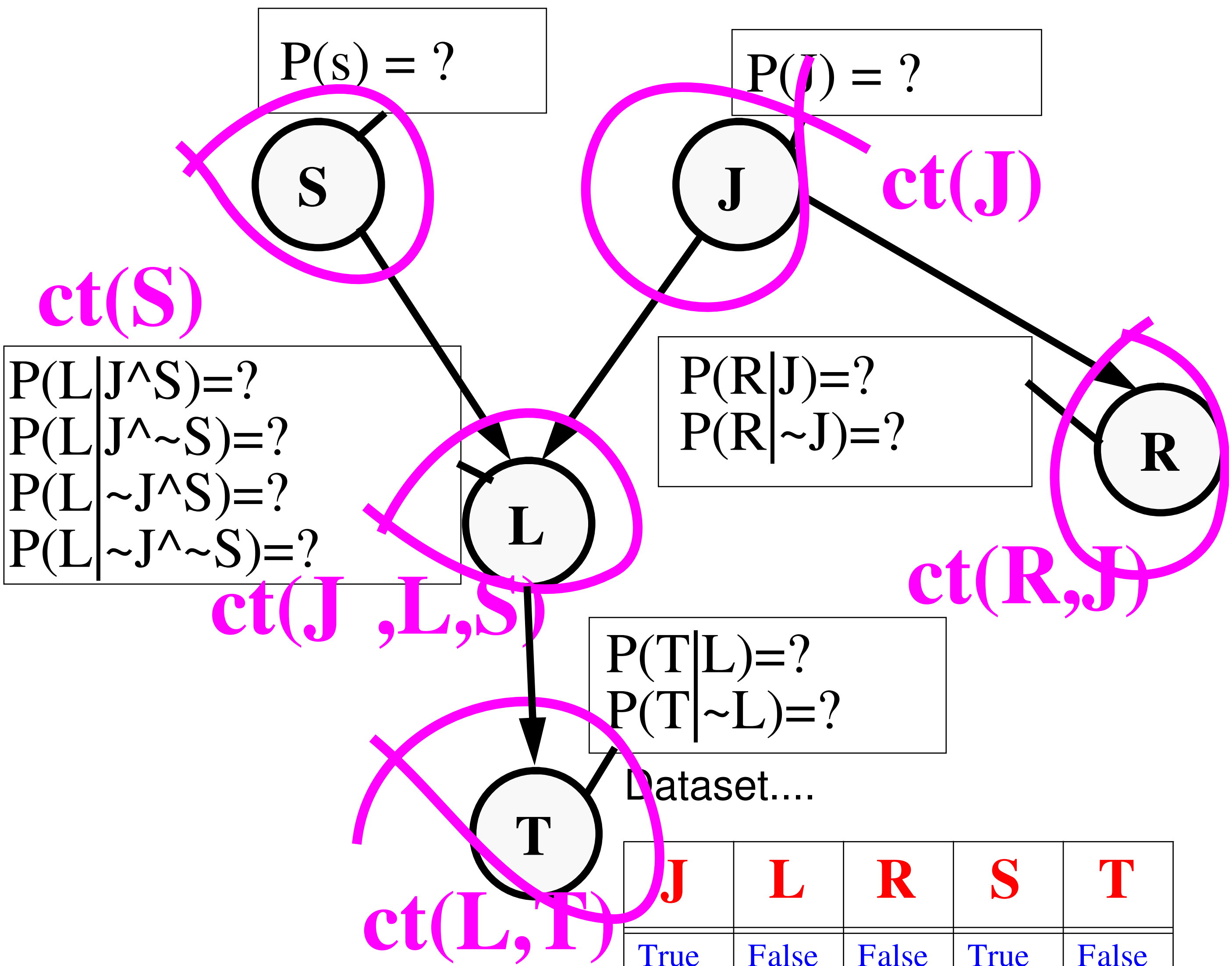
Estimate
P(L|J^~S) as

$$\frac{C(L=\mathrm{T}, J=\mathrm{T}, S=\mathrm{F})}{C(J=\mathrm{T}, S=\mathrm{F})}$$

Dataset....

J	L	R	S	T
True	False	False	True	False
True	True	False	True	True
True	False	False	True	False
False	True	True	False	False
True	True	False	True	True
:	:	:	:	:

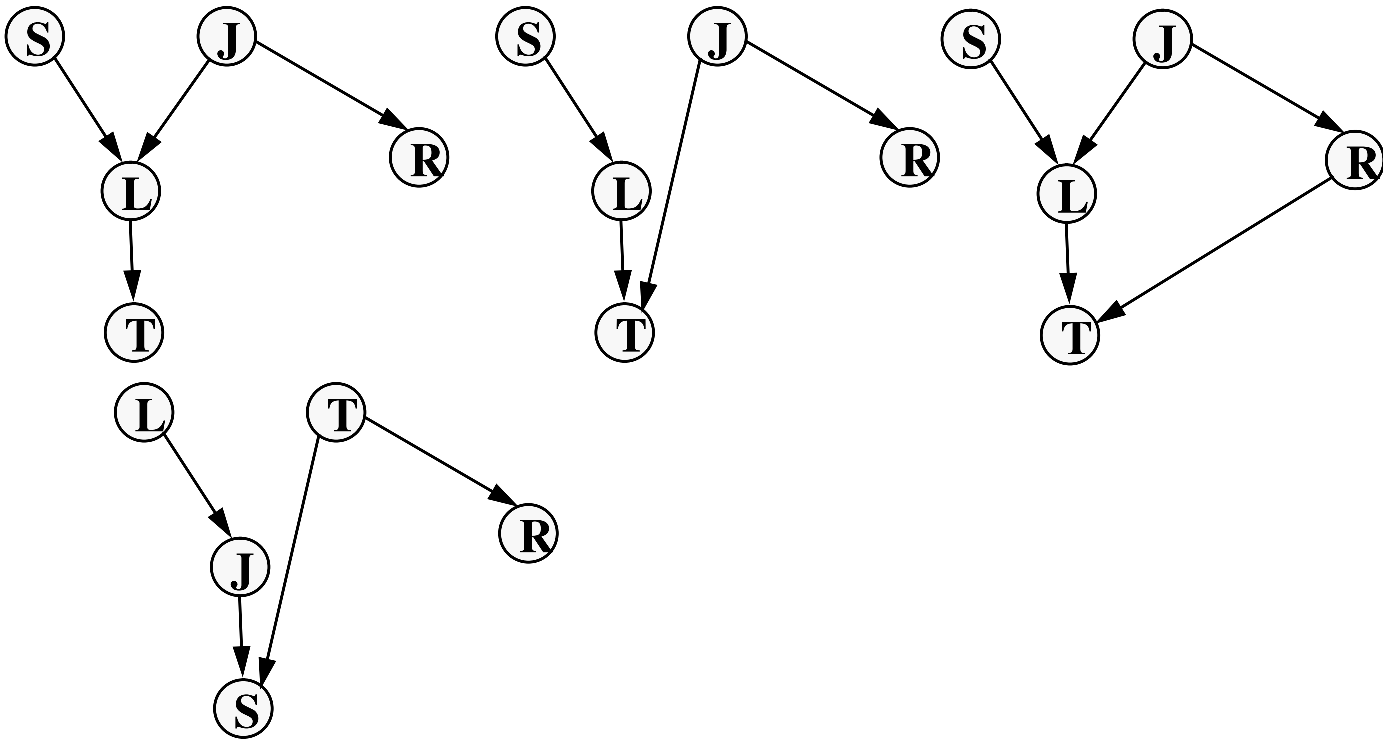
Estimating Bayes Net Probabilities



Dataset....

J	L	R	S	T
True	False	False	True	False
True	True	False	True	True
True	False	False	True	False
False	True	True	False	False
True	True	False	True	True
:	:	:	:	:

Scoring a Bayes Net structure



Score =

$-N_{params}\ \log(R)/2 +$

$R\ \Sigma_{j=1}^{M}$

$\Sigma_{Vals\ in\ Pj}$

$\Sigma_{v=1}^{n(j)}\ P(aj=v \wedge Vals)\ \log\ P(aj=v|vals)$