Keep Your Levels Straight: Separating Variation from Aggregation in Feature Models

Martin Erwig

School of EECS
Oregon State University

CCF-0917092
CCF-1219165
IIS-1314384

FA9550-09-1-0229

Keep Your Titles Short

Martin Erwig

School of EECS
Oregon State University

CCF-0917092
CCF-1219165
IIS-1314384

FA9550-09-1-0229

Fix Feature Diagrams

Martin Erwig

School of EECS
Oregon State University

CCF-0917092
CCF-1219165
IIS-1314384

FA9550-09-1-0229

Main Points

❶ *Don't use Propositional Formulas as the semantics of Feature Diagrams.*

Because they neglect the domain structure

❷ *Define compositional semantics.*

❸ *Reflect type structure in syntax.*

Semantics-Driven DSL
Formal and Practical Aspects of Domain-Specific Languages, 2012

Semantics First! Rethinking the Language Design Process
Int. Conf. on Software Language Engineering, 2011

Static Feature Modeling

Feature

Set of all features

Set of all products

Product

$$
\begin{aligned}
f, g &\in F \\
p, q &\in P = 2^F \\
L, M &\in PL = 2^P
\end{aligned}
$$

Product line

Set of all product lines

Feature Modeling Domain

$$F = \{f, g\}$$

$$P = \{\varnothing, \{f\}, \{g\}, \{f, g\}\} \xrightarrow{\textit{Abbreviation}} \{\varnothing, f, g, fg\}$$

$$PL = \{\varnothing, \{\varnothing\}, \{f\}, \{g\}, \{fg\}, \{\varnothing, f\}, \{\varnothing, g\}, \{\varnothing, fg\}, \ldots, \{\varnothing, f, g, fg\}\}$$

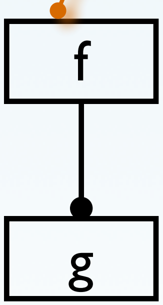
Feature Diagrams

Feature Diagrams:
A DSL for Feature Modeling

Syntax: $S = \{\textit{Trees over } F\}$

Semantics: $D = PL = 2^P = 2^{2^F}$

Feature Diagram



"Product Line Diagram"

$[\![\mathit{mand}(f, g)]\!] = \{\varnothing, fg\}$

$[\![\mathit{mand}(f, g)]\!] = f \Leftrightarrow g$

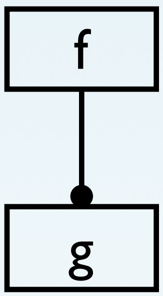
"Feature Selection Diagram"
"Feature Relationship Diagram"

f	g	$f \Leftrightarrow g$
0	0	1
0	1	0
1	0	0
1	1	1

$\varnothing$

fg

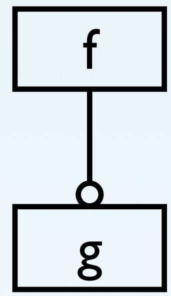
Aggregation vs. Variation



$[\![mand(\mathrm{f}, \mathrm{g})]\!] = \{\varnothing, \mathrm{fg}\}$

$[\![mand(\mathrm{f}, \mathrm{g})]\!] = \mathrm{f} \Leftrightarrow \mathrm{g}$

f	g	$\mathrm{f} \Leftrightarrow \mathrm{g}$
0	0	1
0	1	0
1	0	0
1	1	1

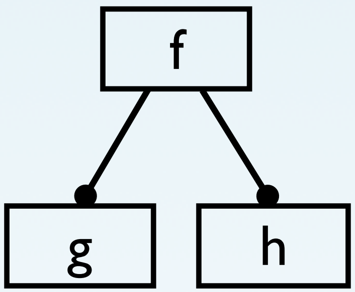


$[\![opt(\mathrm{f}, \mathrm{g})]\!] = \{\varnothing, \mathrm{f}, \mathrm{fg}\}$

$[\![opt(\mathrm{f}, \mathrm{g})]\!] = \mathrm{f} \Leftarrow \mathrm{g}$

f	g	$\mathrm{f} \Leftarrow \mathrm{g}$
0	0	1
0	1	0
1	0	1
1	1	1

Aggregation vs. Variation



$[\![\mathit{mand}(f, g)]\!] = \{\emptyset, fg\}$

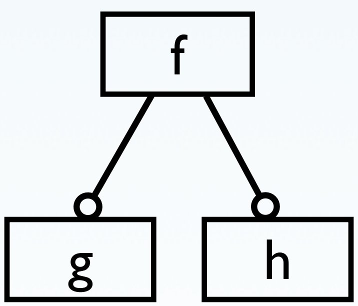
$[\![\mathit{mand}(f, g); \mathit{mand}(f, h)]\!] = \{\emptyset, fgh\}$

Building product

Adding features to one product

$[\![\mathit{mand}(f, g)); \mathit{mand}(f, h)]\!] = f \Leftrightarrow g \wedge f \Leftrightarrow h$

Aggregation operates on individual products



$[\![\mathit{opt}(f, g)]\!] = \{\emptyset, f, fg\}$

$[\![\mathit{opt}(f, g); \mathit{opt}(f, h)]\!] = \{\emptyset, f, fg, fh, fgh\}$

Creating variation

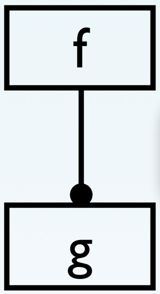
Adding products

$[\![\mathit{opt}(f, g); \mathit{opt}(f, h)]\!] = f \Leftarrow g \wedge f \Leftarrow h$

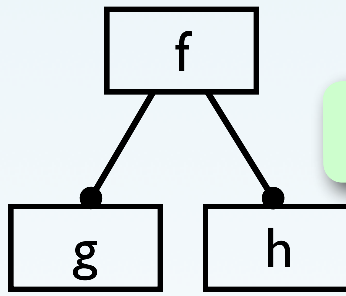
Variation operates on sets of products

Simplifying Assumption

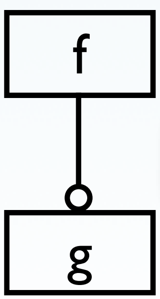
Ignore empty products



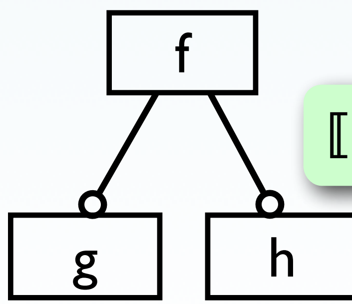
$\llbracket \mathit{mand}(\mathrm{f}, \mathrm{g}) \rrbracket = \{\mathrm{fg}\}$



$\llbracket \mathit{mand}(\mathrm{f}, \mathrm{g}); \mathit{mand}(\mathrm{f}, \mathrm{h}) \rrbracket = \{\mathrm{fgh}\}$

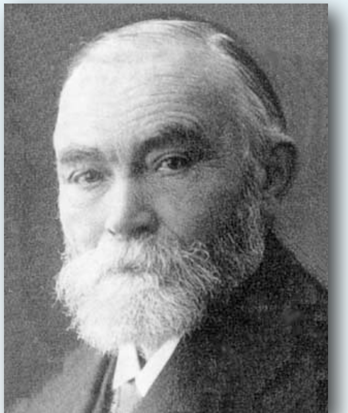


$\llbracket \mathit{opt}(\mathrm{f}, \mathrm{g}) \rrbracket = \{\mathrm{f}, \mathrm{fg}\}$



$\llbracket \mathit{opt}(\mathrm{f}, \mathrm{g}); \mathit{opt}(\mathrm{f}, \mathrm{h}) \rrbracket = \{\mathrm{f}, \mathrm{fg}, \mathrm{fh}, \mathrm{fgh}\}$

Compositionality



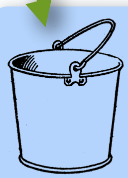
Gottlob Frege

Compositionality

$$[\![f(e_1, \ldots, e_k)]\!] = [\![f]\!]([\![e_1]\!], \ldots, [\![e_k]\!])$$

Inductive Definition

bucket



kick

Napoleon kicked the bucket

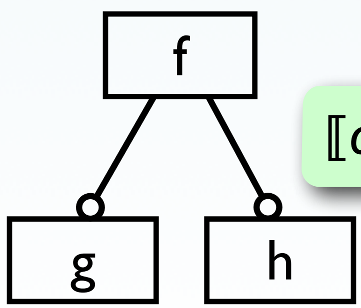
Napoleon booted the bucket

† May 5, 1821

Napoleon Bonaparte

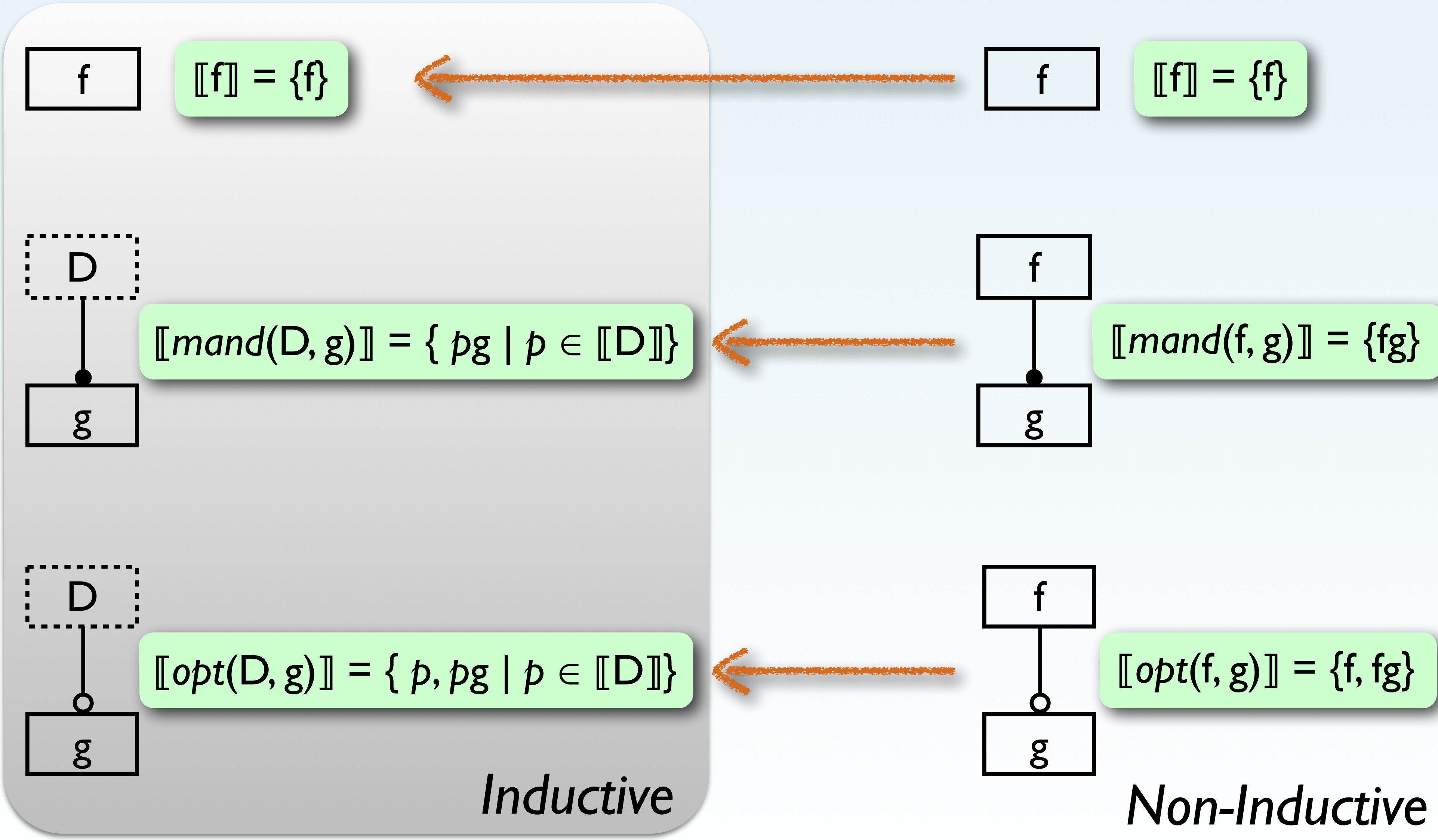
FD Semantics is Not Inductive

"We have to process all edges in one big step"

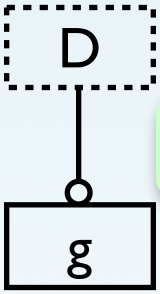


$[\![\mathit{opt}(\mathrm{f}, \mathrm{g}); \mathit{opt}(\mathrm{f}, \mathrm{h})]\!] = \{\mathrm{f}, \mathrm{fg}, \mathrm{fh}, \mathrm{fgh}\}$

Inductive Definition

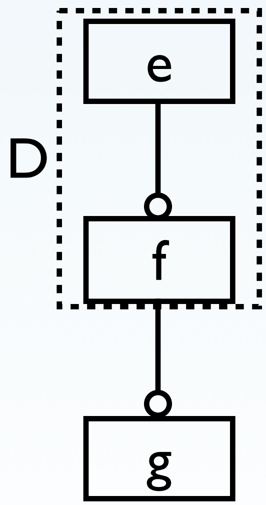


Loss of Compositionality



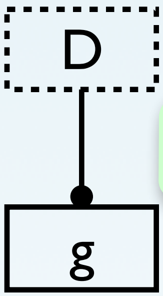
$[\![\mathit{opt}(\mathrm{D}, \mathrm{g})]\!] = \{\, p, p\mathrm{g} \mid p \in [\![\mathrm{D}]\!] \}$

Inductive Definition



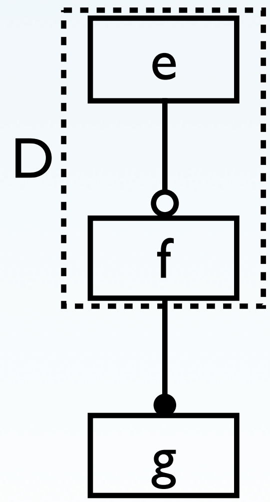
$$
\begin{aligned}
& [\![\mathit{opt}(\mathit{opt}(\mathrm{e}, \mathrm{f}), \mathrm{g})]\!] \\
&= \{\, p, p\mathrm{g} \mid p \in [\![\mathit{opt}(\mathrm{e}, \mathrm{f})]\!] \} \\
&= \{\, p, p\mathrm{g} \mid p \in \{\mathrm{e}, \mathrm{ef}\}]\!] \} \\
&= \{\mathrm{e}, \cancel{\mathrm{eg}}, \mathrm{ef}, \mathrm{efg}\}
\end{aligned}
$$

Loss of Compositionality



$[\![\mathit{mand}(\mathrm{D}, \mathrm{g})]\!] = \{\, p\mathrm{g} \mid p \in [\![\mathrm{D}]\!] \}$

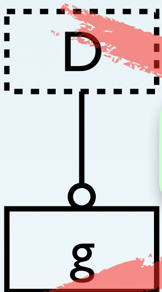
Inductive Definition



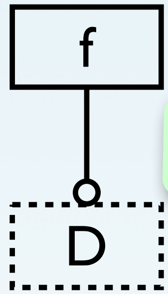
$[\![\mathit{mand}(\mathit{opt}(\mathrm{e}, \mathrm{f}), \mathrm{g})]\!]$
$= \{\, p\mathrm{g} \mid p \in [\![\mathit{opt}(\mathrm{e}, \mathrm{f})]\!] \}$
$= \{\, p\mathrm{g} \mid p \in \{\mathrm{e}, \mathrm{ef}\}]\!] \}$
$= \{\cancel{\mathrm{eg}}, \mathrm{efg}\}$

$\{\mathrm{e}, \mathrm{efg}\}$

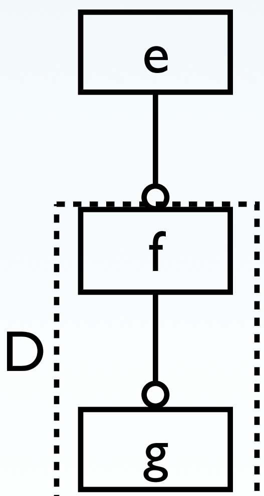
Loss of Compositionality



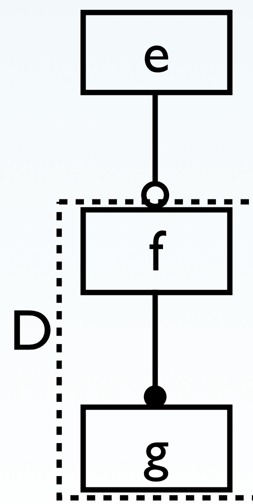
$\cancel{\llbracket \mathit{opt}(\mathrm{D}, \mathrm{g}) \rrbracket = \{\ p, p\mathrm{g} \mid p \in \llbracket \mathrm{D} \rrbracket \}}$



$\llbracket \mathit{opt}(\mathrm{f}, \mathrm{D}) \rrbracket = \{\ \mathrm{f}, \mathrm{f}p \mid p \in \llbracket \mathrm{D} \rrbracket \}$



$\llbracket \mathit{opt}(\mathrm{e}, \mathit{opt}(\mathrm{f}, \mathrm{g})) \rrbracket$
$= \{\ \mathrm{e}, \mathrm{e}p \mid p \in \llbracket \mathit{opt}(\mathrm{f}, \mathrm{g}) \rrbracket \}$
$= \{\ \mathrm{e}, \mathrm{e}p \mid p \in \{\mathrm{f}, \mathrm{fg}\} \rrbracket \}$
$= \{\mathrm{e}, \mathrm{ef}, \mathrm{efg}\}$



$\llbracket \mathit{opt}(\mathrm{e}, \mathit{mand}(\mathrm{f}, \mathrm{g})) \rrbracket$
$= \{\ \mathrm{e}, \mathrm{e}p \mid p \in \llbracket \mathit{mand}(\mathrm{f}, \mathrm{g}) \rrbracket \}$
$= \{\ \mathrm{e}, \mathrm{e}p \mid p \in \{\mathrm{fg}\} \rrbracket \}$
$= \{\mathrm{e}, \mathrm{efg}\}$

Observations

binds stronger than

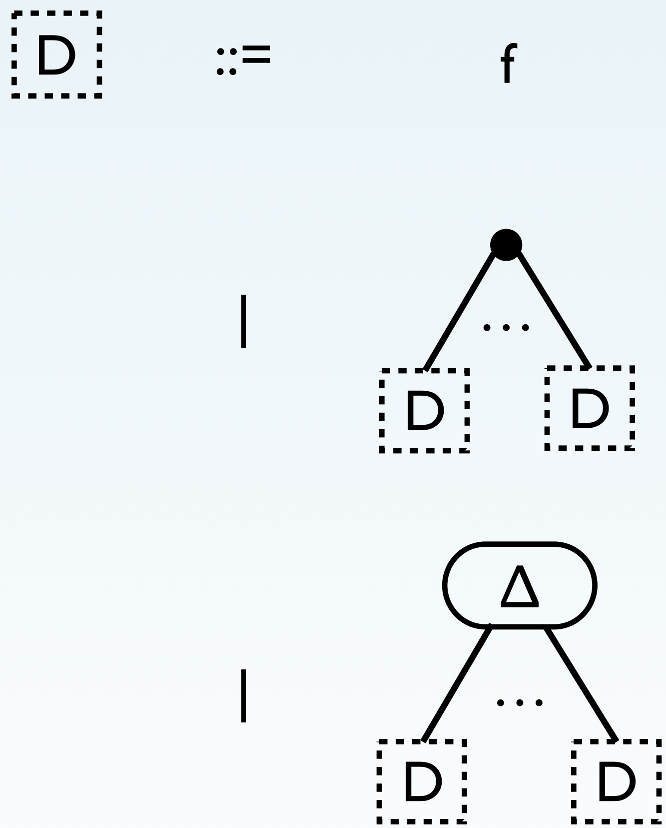
associates to the "bottom"

And Then …

… I ran out of time

Product Line Diagrams

Alternative Notation for Feature Diagram



Feature Nodes

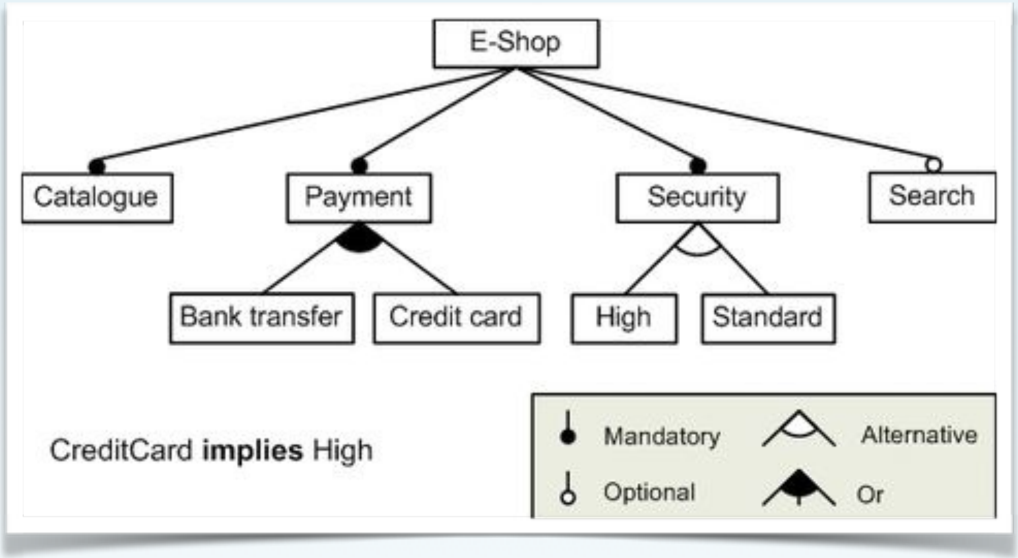
Only in leaves

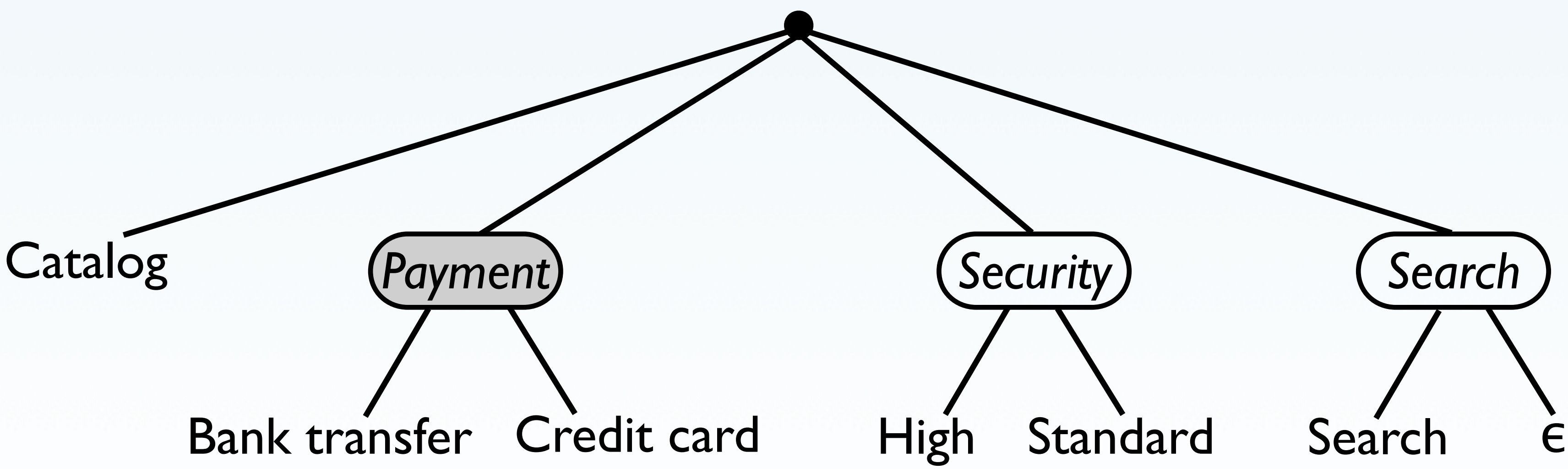
Product / Aggregation Nodes

Only internal nodes

Family / Variation Nodes

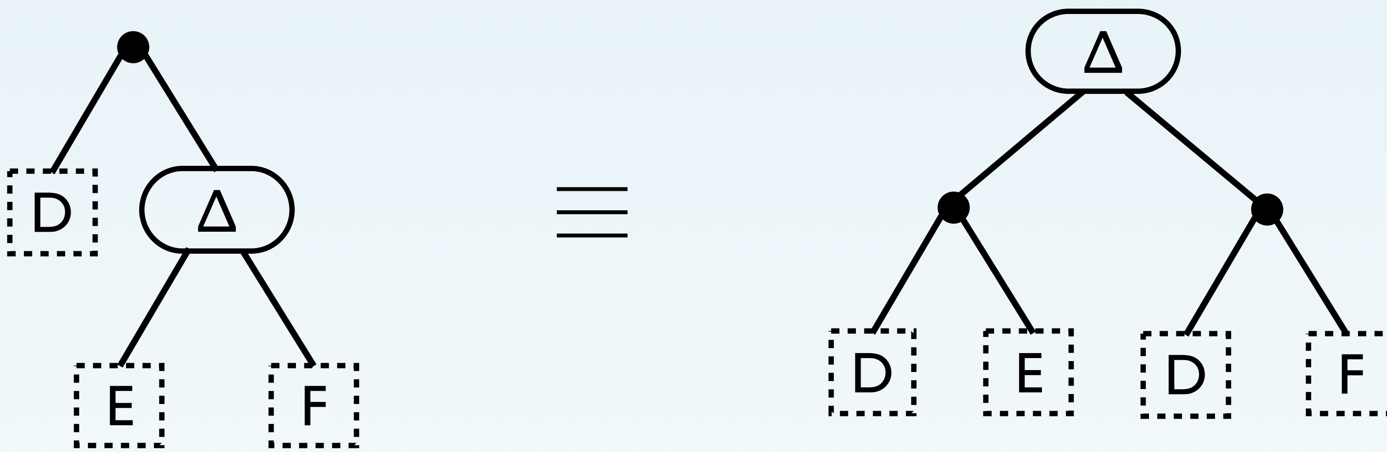
Examples

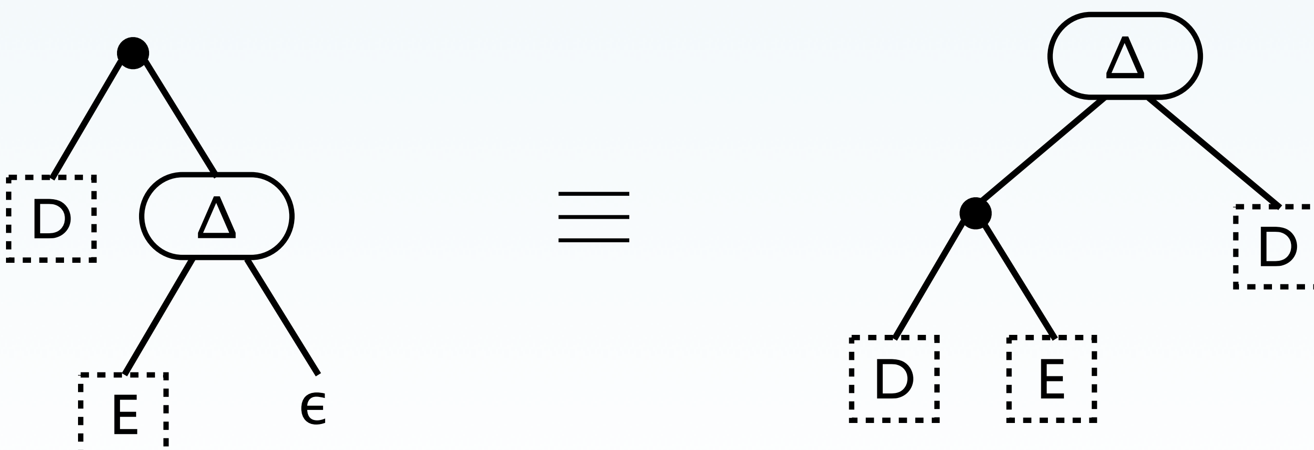




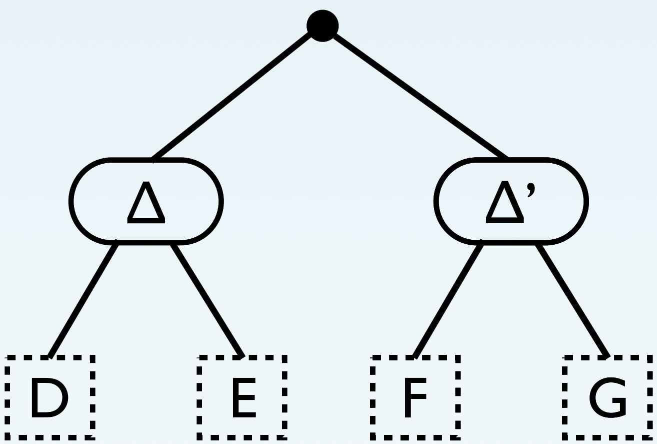
Credit card ⇒ High

Diagram Laws





More Diagram Laws



$\Delta(D, E) \bullet \Delta'(F, G)$
$= \Delta(D \bullet \Delta'(F, G), E \bullet \Delta'(F, G))$
$= \Delta(\Delta'(D\bullet F, D\bullet G), \Delta'(E\bullet F, E\bullet G))$

$\equiv$

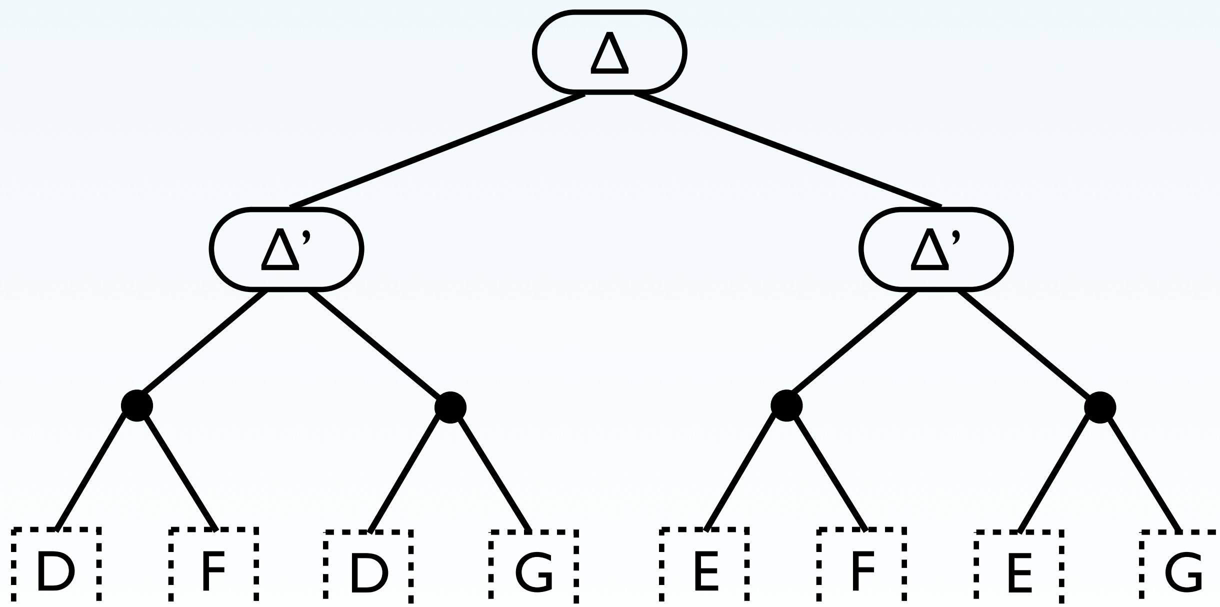
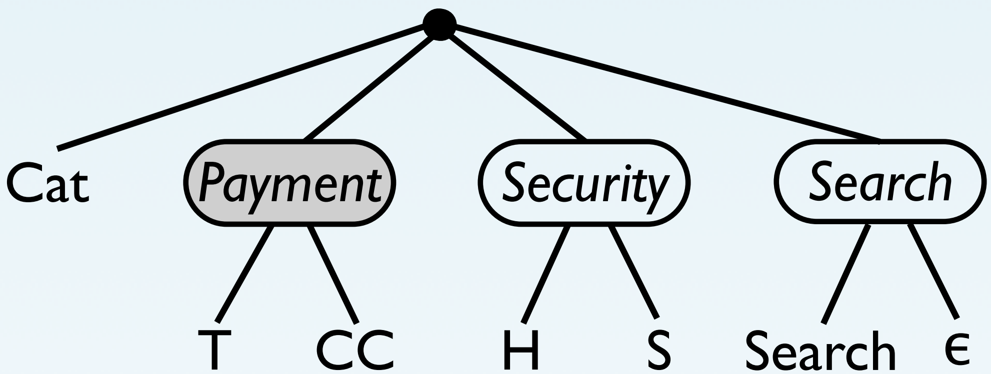
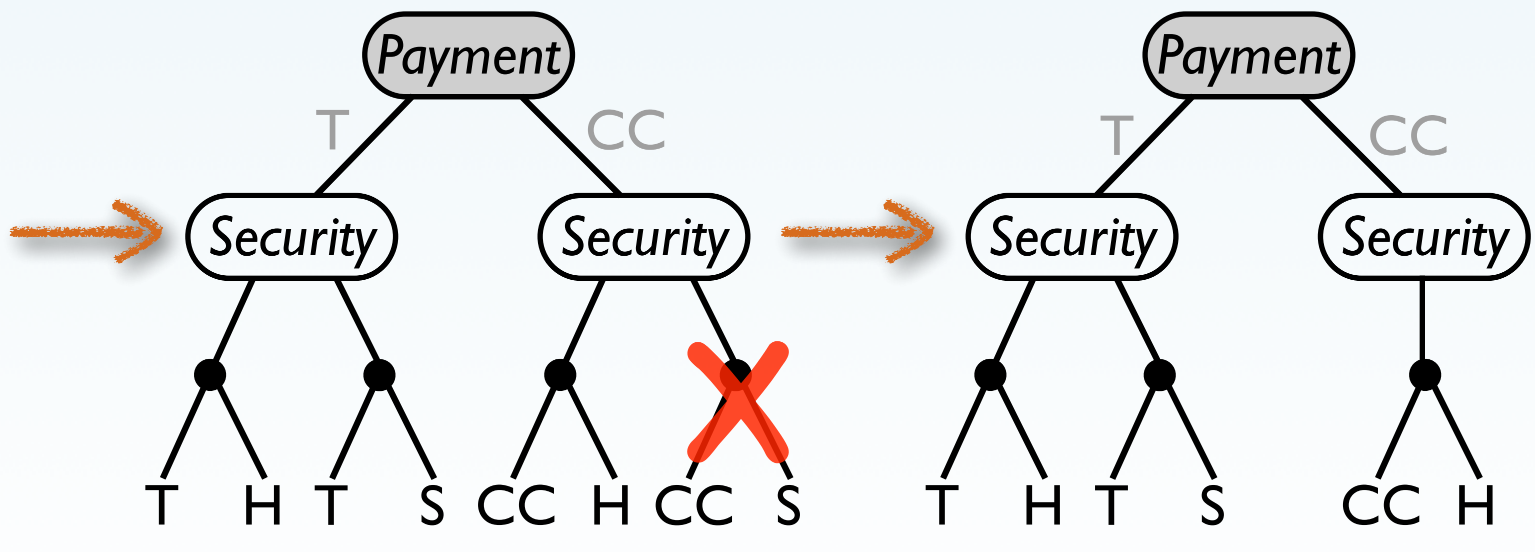


Diagram Reasoning





Credit card ⇒ High

Diagram Reasoning

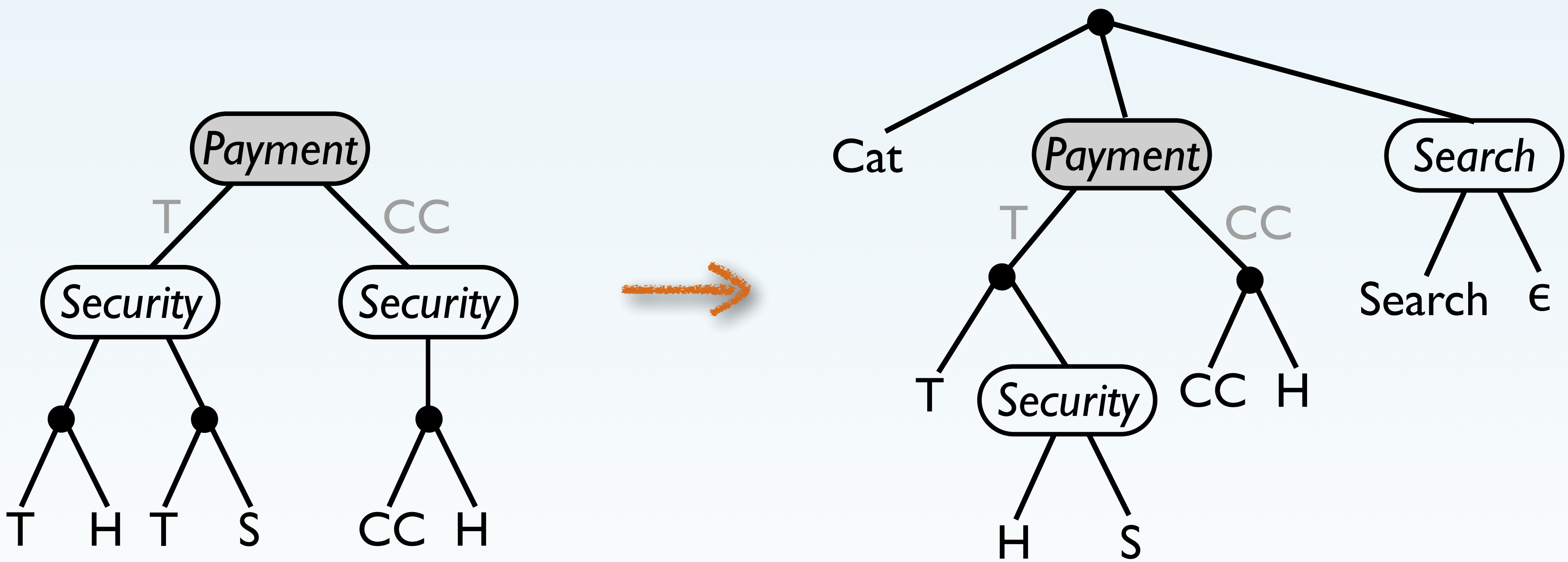
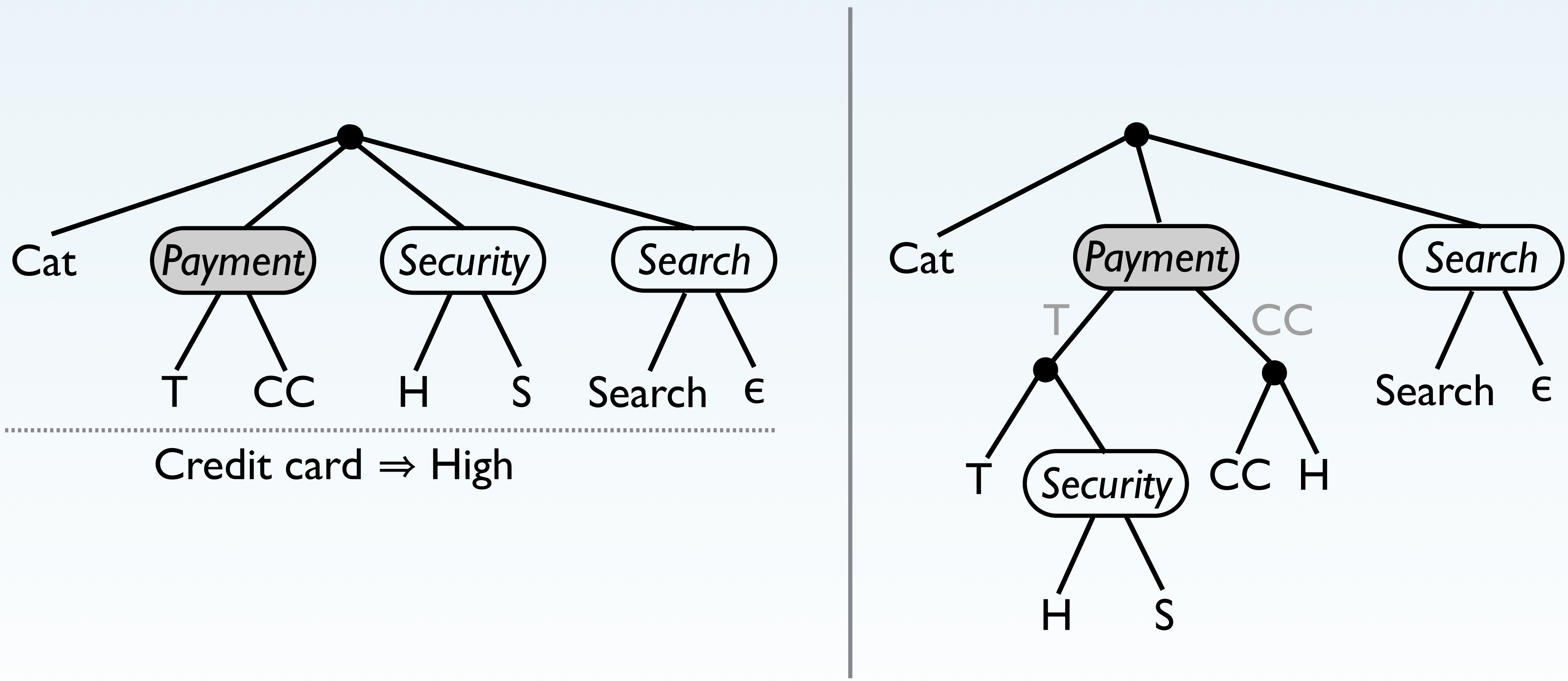


Diagram Reasoning



And Finally …

GOD – *Greatest of Diagrams*

Inbred – *Inductive Product Line Reasoning Diagrams*

Splendid – *Software Product Line Enriching Reasoning Diagrams Do it*

In-Law – *Inductive, Lawful Notation for Product Families*