



max planck institut
informatik

Variability Management

or

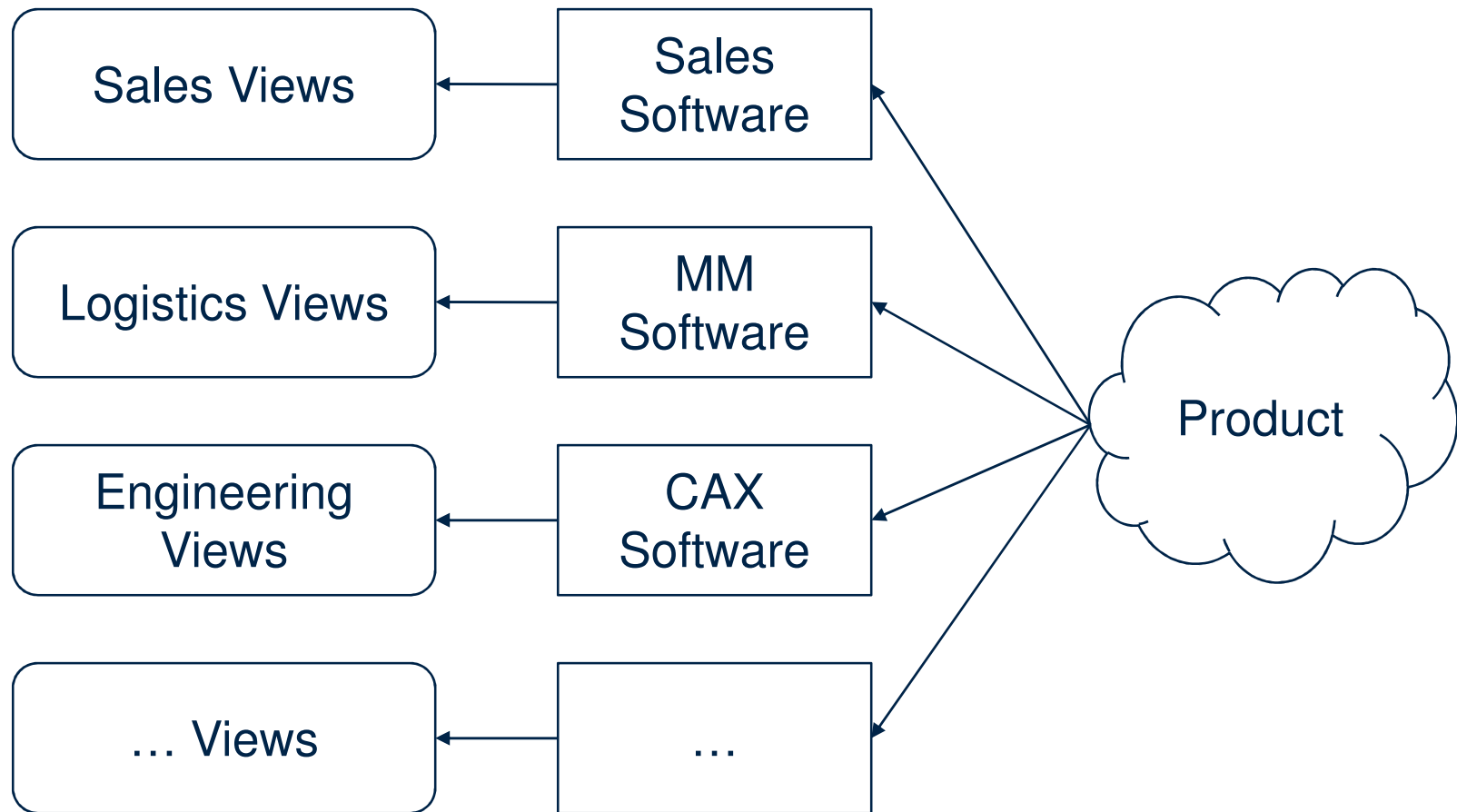
How to construct a new car in 5 min

or

How to make your specification run

Prof. Dr. Christoph Weidenbach

Today's Architecture

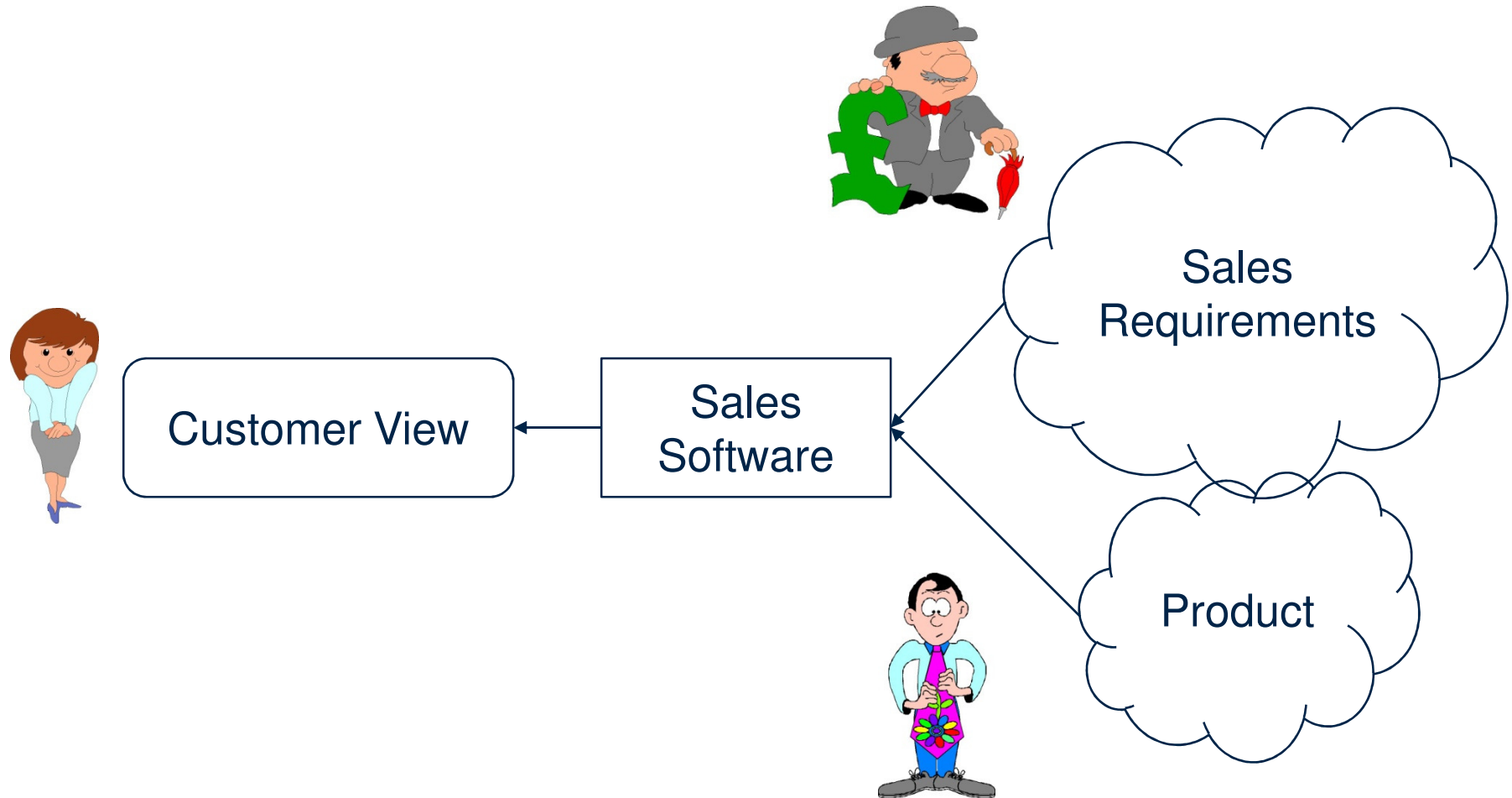


Application

Application + DB

Paper, People

Sales Software Development



Disadvantages

- changes to the product means changes to the software
- verification of the software not affordable
- architecture is not flexible
- no consistency between product and software
- no queries beyond single views

Advantages

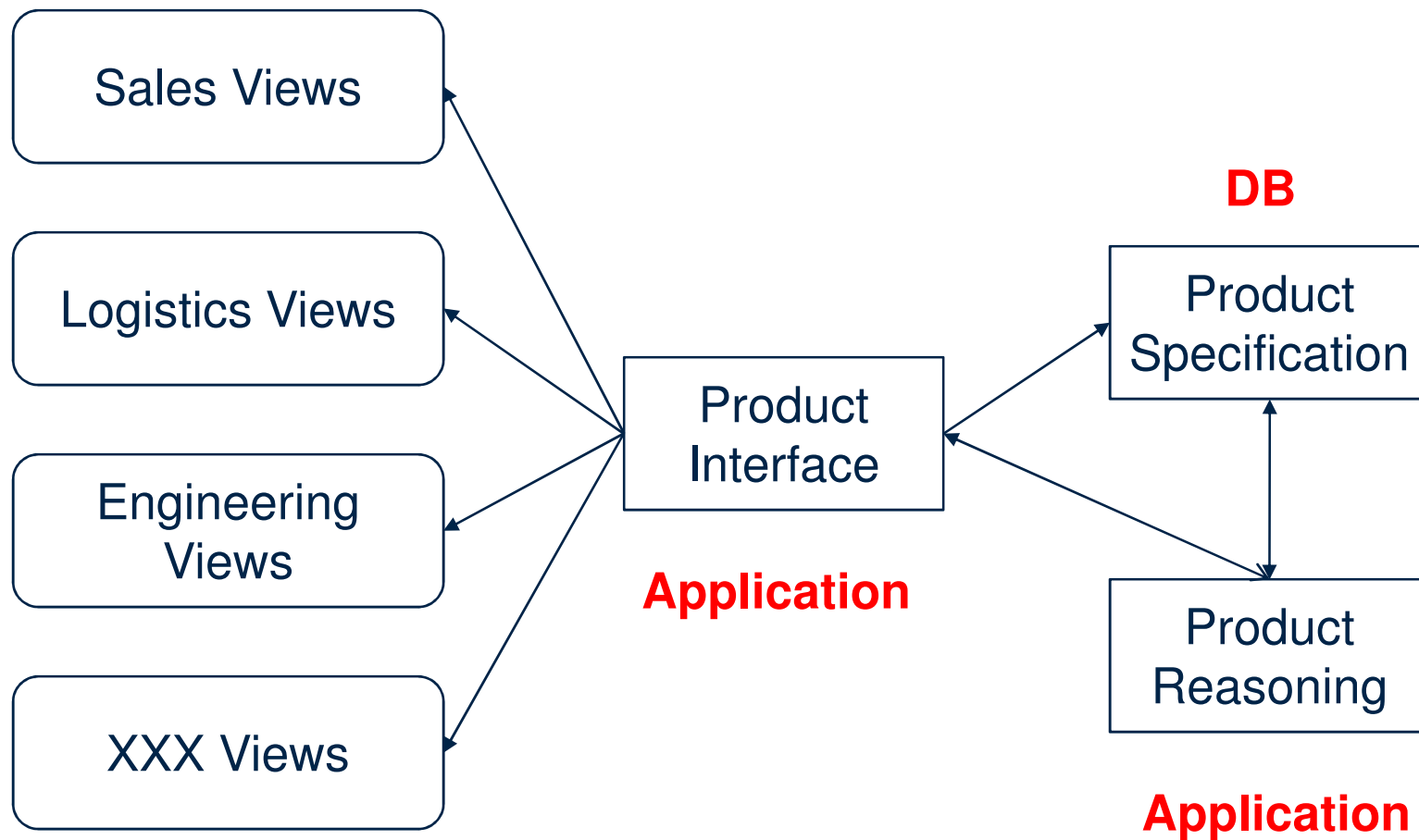
- it works



What Cannot be Answered

- Can we build a car with weight less than x ?
- Is there a reasonable substitution for part x ?
- Can we produce car x without supplier y ?
- Which parts of our portfolio are not used anymore?
- How long does it need to build a new car with properties x ?
- Is it profitable to get rid of part x ?
- What is the most profitable car we could build?
- How much does it cost to produce a real sports car?
- ...





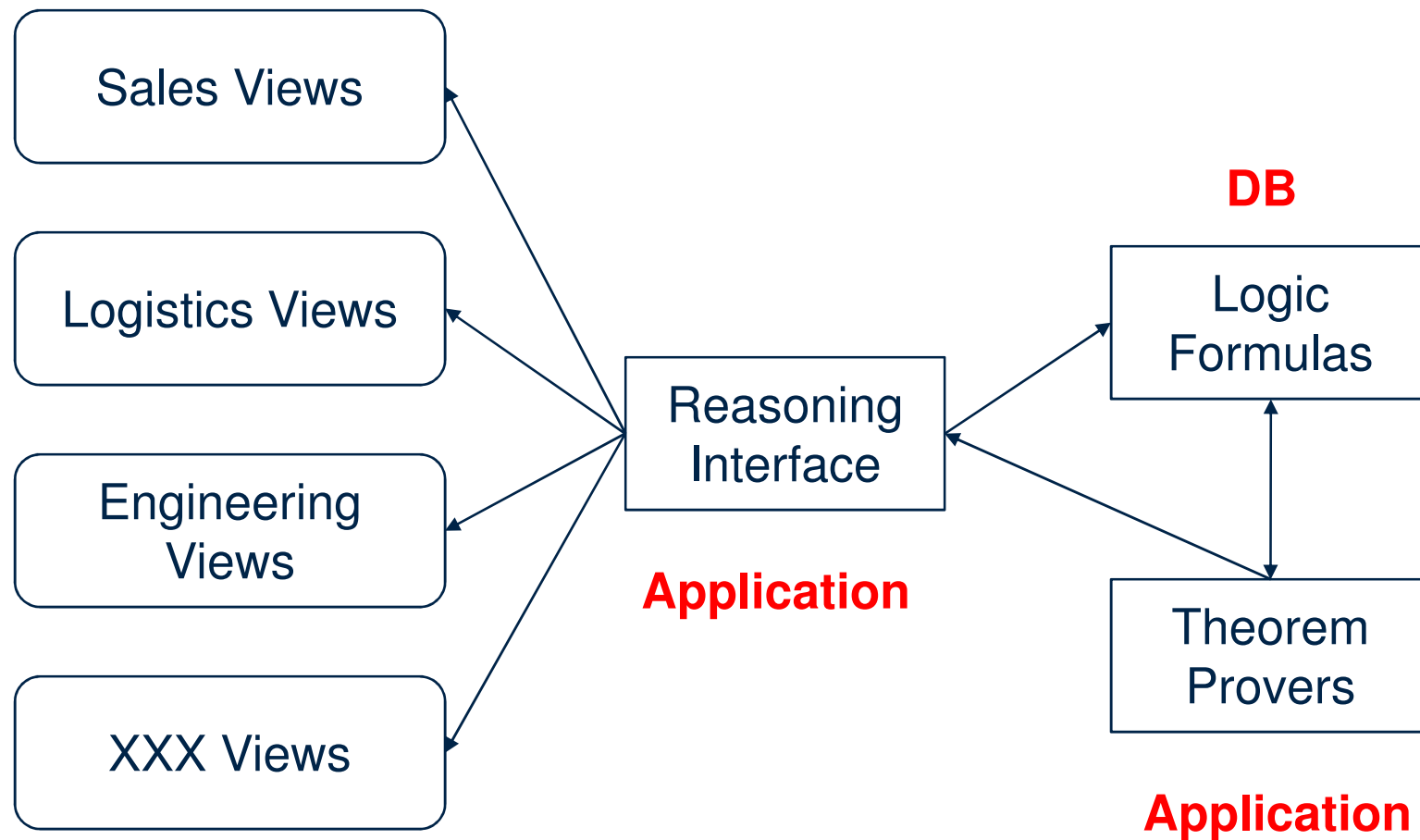
Advantages

- changes to the product automatically adjust software
- verification of the software for free
- architecture is highly flexible
- proven consistency between product and software
- support for overall product queries

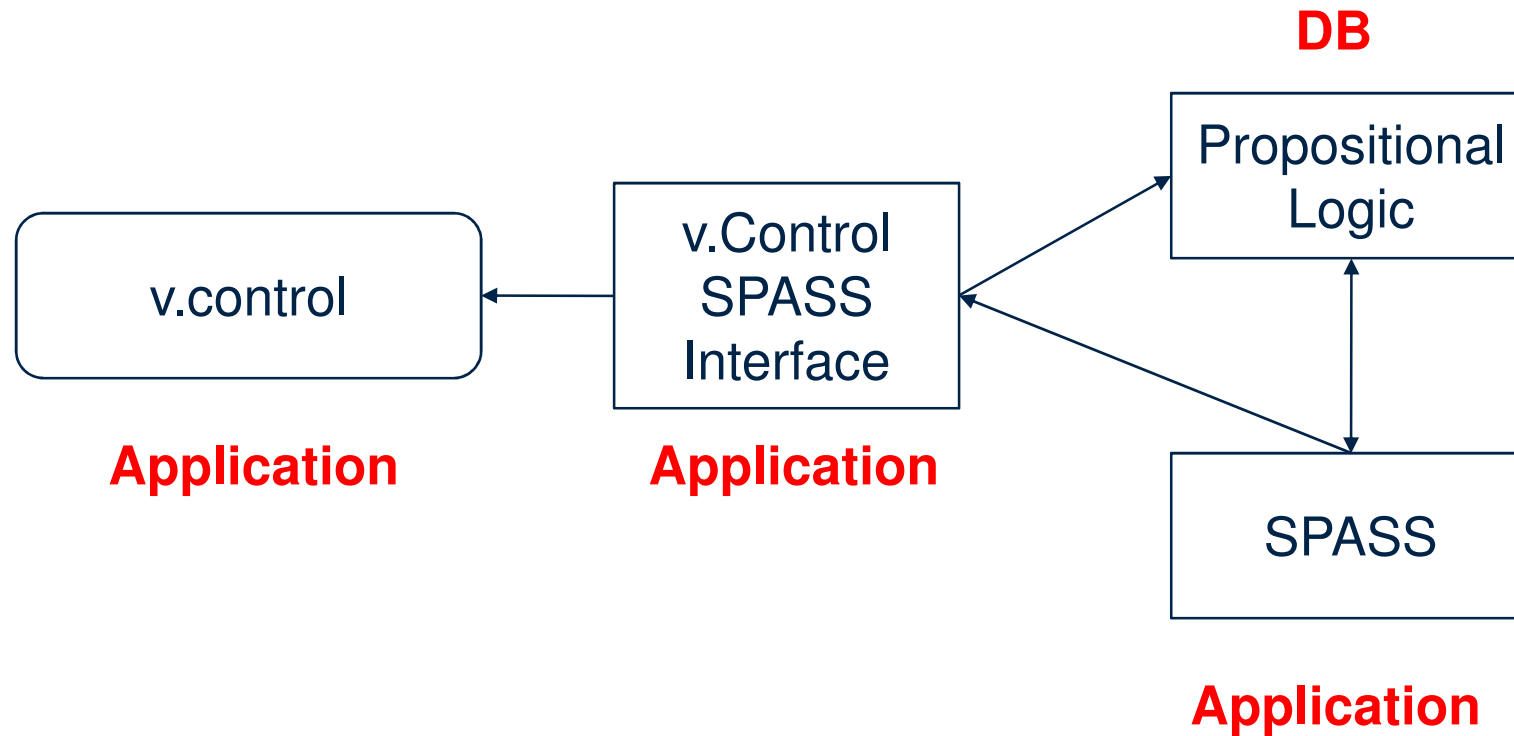
Disadvantages

- it does not work **yet**





Concrete Example



- Language: propositional variables can be true (1) or false (0)
- Connectives: $\Rightarrow$ implication, $\neg$ negation, $\vee$ disjunction, $\wedge$ conjunction
- Clause: disjunction of variables or their negations (literal)
- Validity: a formula is valid iff it is true for all possible assignments
- Assignment: setting all propositional variables 1 or 0, can also be expressed by showing the true literals
- we write $M \models C$ if the clause C is true by assignment M
- SAT: propositional satisfiability, find an assignment such that for a set of clauses all clauses are valid in the assignment



UProp(N, M)

while (there is a clause $C' \vee L \in N$ such that

$M \models \neg C'$ and $L \notin M$ and $\neg L \notin M$)

$M := M \cup \{L\};$

return M ;

UProp($\{\neg A \vee \neg B \vee C, \neg A \vee B, \neg C, D, A\}, \emptyset$)

$\rightarrow M = \emptyset$

$\rightarrow M = \{\neg C\}$

$\rightarrow M = \{\neg C, D\}$

$\rightarrow M = \{\neg C, D, A\}$

$\rightarrow M = \{\neg C, D, A, B\}$



DPLL(N, M)

if for all $C \in N$ we have $M \models C$ return true;
 if there is some $C \in N$ with $M \models \neg C$ return false;
 select a variable P occurring in N but not in M ;
 if (DPLL(N , UProp(N , $M \cup \{P\}$))) then
 return true;
 else
 return DPLL(N , UProp(N , $M \cup \{\neg P\}$));

$\neg A \vee \neg B \vee C$

$\neg A \vee B$

$\neg C$

$A \vee D$

DPLL($N, \emptyset$)

DPLL(N , UProp($N, \{A\}$)) DPLL(N , UProp($N, \{\neg A\}$))

DPLL($N, \{A, B, \neg C\}$) DPLL($N, \{\neg A, D, \neg C\}$)

DPLL is sound and complete and terminating for SAT.



$\text{Corsa} \Rightarrow \text{Wheels} \wedge \text{Engines}$

$4\text{-Holes} \Rightarrow \text{Wheels}$

$5\text{-Holes} \Rightarrow \text{Wheels}$

$4\text{-Holes} \Rightarrow \neg 5\text{-Holes}$

$5\text{-Holes} \Rightarrow \neg 4\text{-Holes}$

$\text{Diesel} \Rightarrow \text{Engines}$

$\text{Gasoline} \Rightarrow \text{Engines}$

$\text{Diesel} \Rightarrow \neg \text{Gasoline}$

$\text{Gasoline} \Rightarrow \neg \text{Diesel}$

$\text{Diesel} \Rightarrow \neg 4\text{-Holes}$

Reasoning: $\text{Corsa} \rightarrow \text{Wheels, Engines}$

$4\text{-Holes} \rightarrow \neg 5\text{-Holes, } \neg \text{Diesel, Gasoline}$

$\text{Gasoline} \rightarrow \neg \text{Diesel}$



Challenge: Scalability

- before 2009: approx. 1500 nodes
- in 2009: v.control + SPASS approx. 3000 nodes
- in x years: for a reasonable product approx. 60000 nodes



Thank you for your attention

