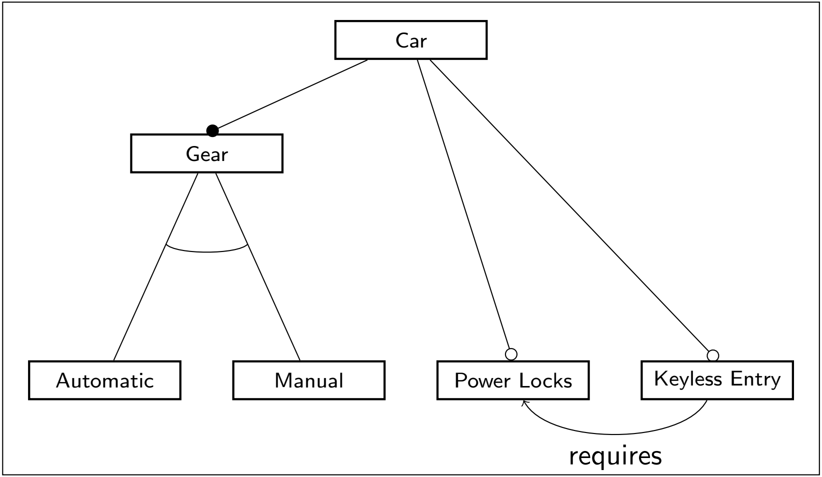
SAT Solving in Interactive Configuration

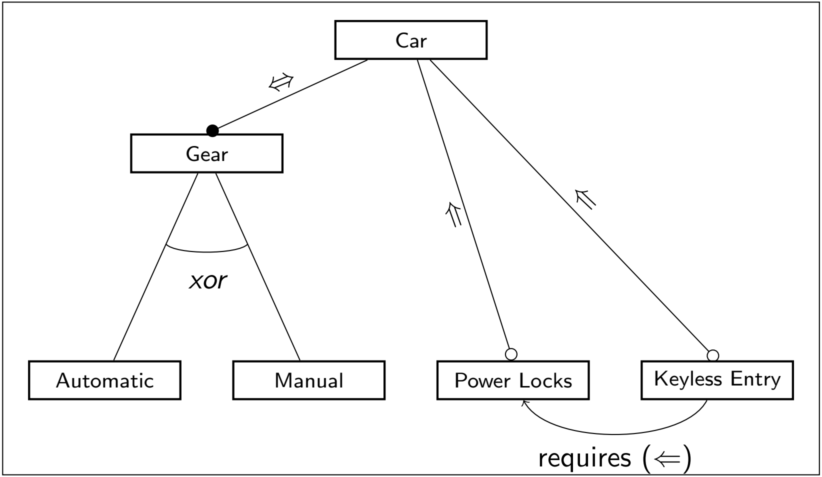
Mikoláš Janota

Lero
University College Dublin
Ireland

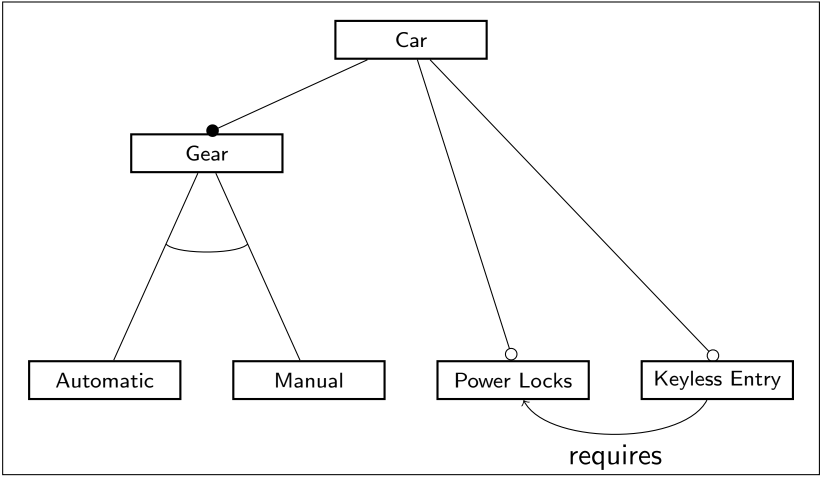
Feature Diagrams



Feature Diagrams



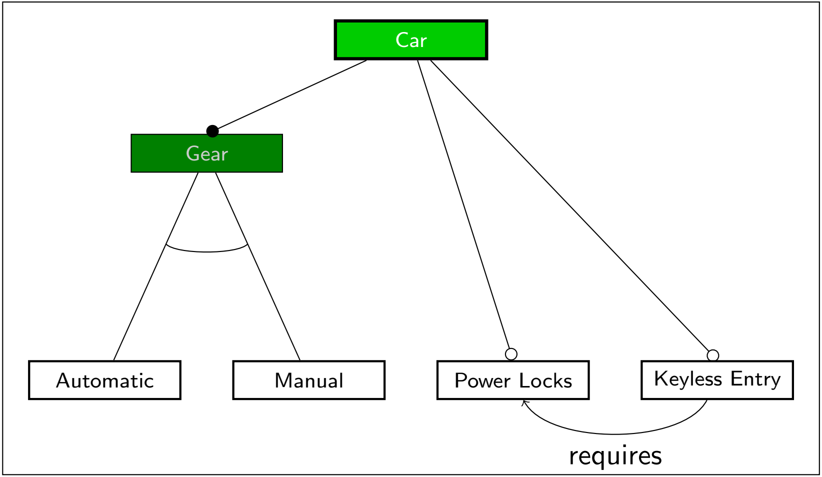
Configuration



Legend

Selection Eliminate Autoselect Autoeliminate

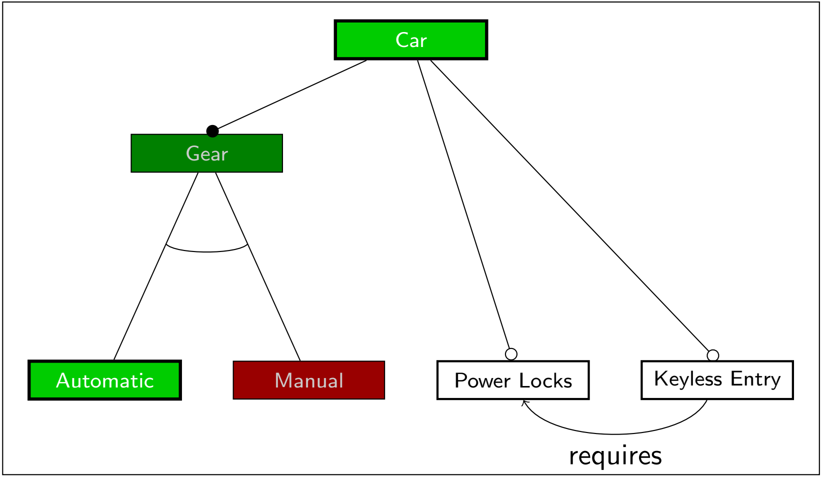
Configuration



Legend

🟩 Selection 🟥 Eliminate 🟩 Autoselect 🟥 Autoeliminate

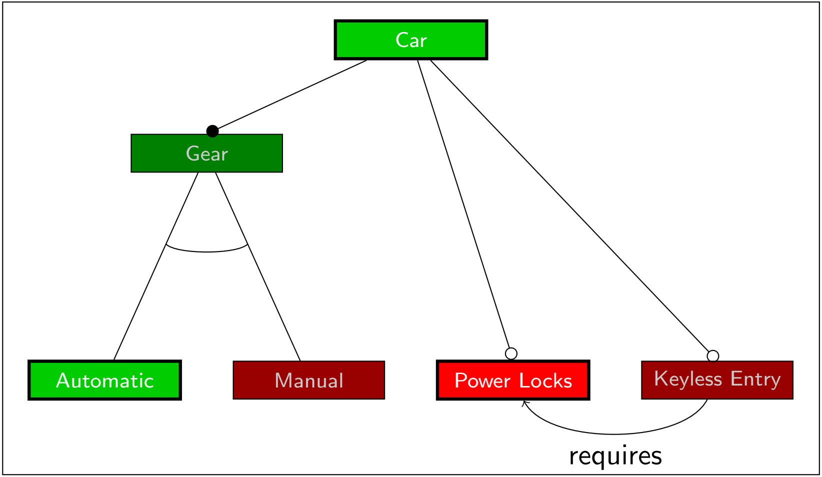
Configuration



Legend

- Selection
- Eliminate
- Autoselect
- Autoeliminate

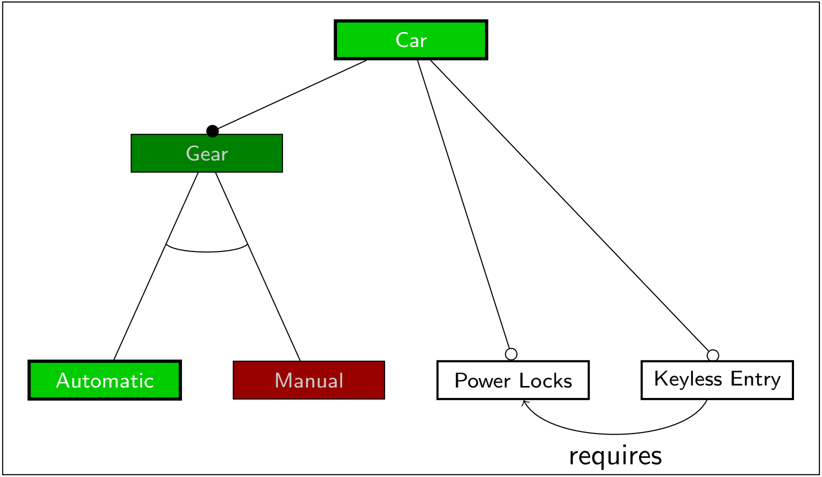
Configuration



Legend

Selection | Eliminate | Autoselect | Autoeliminate

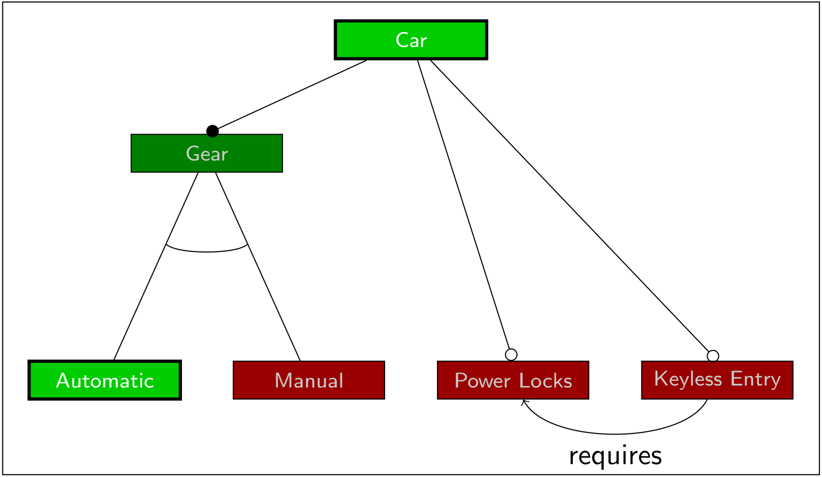
“I’m Done”



Legend

Selection Eliminate Autoselect Autoeliminate

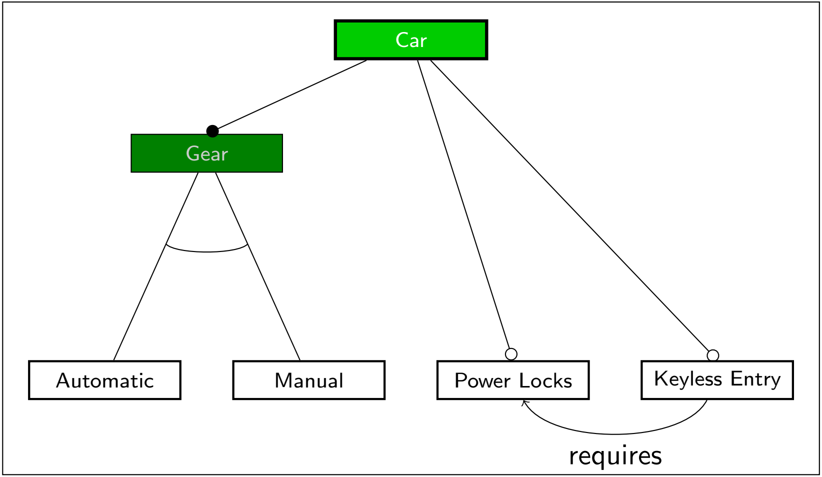
“I’m Done”



Legend

🟩 Selection 🟥 Eliminate 🟩 Autoselect 🟥 Autoeliminate

“I’m Done” sometimes doesn’t work



Legend

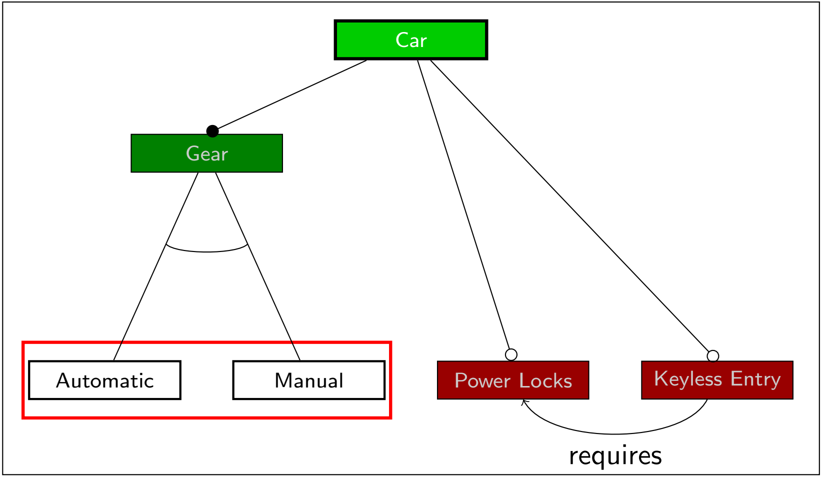
Selection

Eliminate

Autoselect

Autoeliminate

“I’m Done” sometimes doesn’t work



Legend

Selection Eliminate Autoselect Autoeliminate

The General Problem

What is the general problem?

Provide assistance during interactive configuration.

The General Problem

What is the general problem?

Provide assistance during interactive configuration.

Why is it important?

- **applicability** – a means to personalization of products and systematic reuse of software components and services

The General Problem

What is the general problem?

Provide assistance during interactive configuration.

Why is it important?

- **applicability** – a means to personalization of products and systematic reuse of software components and services
- **constraints** reduce the risk of incorrect usage

The General Problem

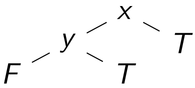
What is the general problem?

Provide assistance during interactive configuration.

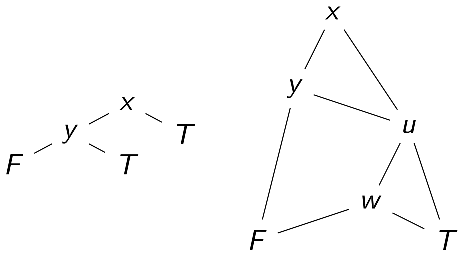
Why is it important?

- **applicability** – a means to personalization of products and systematic reuse of software components and services
- **constraints** reduce the risk of incorrect usage
- **interactive** – increases comfort for the user

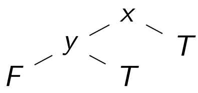
Binary Decision Diagrams (BDDs)

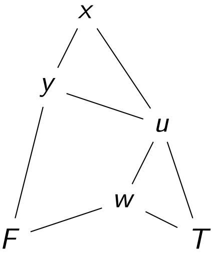


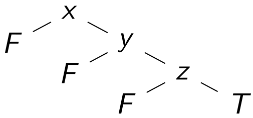
Binary Decision Diagrams (BDDs)



Binary Decision Diagrams (BDDs)







BDD continued

Why is constructing BDDs hard?

- the number of models of a formula grows exponentially

BDD continued

Why is constructing BDDs hard?

- the number of models of a formula grows exponentially
- the ordering of the variables matters

BDD continued

Why is constructing BDDs hard?

- the number of models of a formula grows exponentially
- the ordering of the variables matters
- the way of constructing the BDD matters

BDD continued

Why is constructing BDDs hard?

- the number of models of a formula grows exponentially
- the ordering of the variables matters
- the way of constructing the BDD matters
- heuristics are needed for the two above

Specific Problem

Thesis

Are modern SAT solvers suitable means for implementing assistance during interactive configuration?

. . . where the suitability is measured by the response time.

Why SAT solvers?

The problem solved is computationally challenging

- Deciding whether a variable must have a certain value is co-NP-complete.
- Deciding whether a variable can be safely eliminated is Π_2^P-complete.

Why SAT solvers?

The problem solved is computationally challenging

- Deciding whether a variable must have a certain value is co-NP-complete.
- Deciding whether a variable can be safely eliminated is Π_2^P-complete.

SAT Solvers are neat

- SAT solvers are means to deciding an NP-complete problem (formula satisfiability).
- Modern SAT solvers became very efficient.
- The tools can be used off-the-shelf.

Novelty

Existing Alternative Approaches to Assistance

- ad hoc
- precompilation
- not backtrack free or not complete

Novelty

Existing Alternative Approaches to Assistance

- ad hoc
- precompilation
- not backtrack free or not complete

Completing Configuration

- The presented formalization is novel.
- Similar algorithms exist but applied in different context.

Novelty

Existing Alternative Approaches to Assistance

- ad hoc
- precompilation
- not backtrack free or not complete

Completing Configuration

- The presented formalization is novel.
- Similar algorithms exist but applied in different context.

Syntactic Optimization

- Presented optimization is known but the focus on the proof reconstruction is novel.

Research Methodology

Constructive Research

- review of related work
- formulation of hypothesis
- formalization of the problem
- development of the algorithms
- an empirical evaluation of the implemented algorithms

A Case for the Thesis

Contributions

- study of a lazy approach to interactive configuration,
- study of approaches to configuration process completion.

A Case for the Thesis

Contributions

- study of a lazy approach to interactive configuration,
- study of approaches to configuration process completion.

Advantages over the precompilation methods

- precompilation may fail or may require specific expertise to be carried out,
- more informative explanations can be provided.

A Case for the Thesis

Contributions

- study of a lazy approach to interactive configuration,
- study of approaches to configuration process completion.

Advantages over the precompilation methods

- precompilation may fail or may require specific expertise to be carried out,
- more informative explanations can be provided.

Conclusions from the Evaluation

- the time of feedback during configuration was satisfactory,
- the runtime off the algorithm for configuration completion was satisfactory in majority of the cases but timed out in some.

How Do We Go About This?

Use a SAT Solver

- determines the satisfiability of a given Boolean formula
- operates on Conjunctive Normal Form (CNF)
- a certification of the response is produced
- nowadays SAT solvers are *very* efficient

How Do We Go About This?

Use a SAT Solver

- determines the satisfiability of a given Boolean formula
- operates on Conjunctive Normal Form (CNF)
- a certification of the response is produced
- nowadays SAT solvers are *very* efficient

Assumptions

- constraints encoded in a CNF
- decisions so far encoded as a conjunction of literals

$$\phi \equiv f_1 \wedge \neg f_8 \wedge \dots$$

Towards Satisfiability

Proposition

for a formula ϕ and a literal l

$$\phi \models l \quad \textit{iff} \quad (\phi \wedge \neg l) \text{ is unsatisfiable}$$

Example

for $\phi = (x \Rightarrow y) \wedge x$

$$\phi \models y$$

$$\phi \wedge \neg y \text{ is unsatisfiable}$$

SAT Solver for Configuration

Testing all unassigned variables after each user's decision

```
TEST-VARS()
foreach $x$ that was not assigned to by the user
    do $CanBeTrue \leftarrow \mathrm{TEST\text{-}SAT}(\phi, x)$
       $CanBeFalse \leftarrow \mathrm{TEST\text{-}SAT}(\phi, \neg x)$
       if $\neg CanBeTrue \wedge \neg CanBeFalse$
          then error "Unsatisfiable constraint!"
       if $\neg CanBeTrue$ then SET($x$, FALSE)
       if $\neg CanBeFalse$ then SET($x$, TRUE)
       if $CanBeTrue \wedge CanBeFalse$
          then RESET($x$)
               UNLOCK($x$)
          else LOCK($x$)
```

Can We Improve This?

SAT

- For satisfiable queries, the SAT solver returns with a satisfying assignment.
- All the values in this assignment are satisfiable and don't need to be queried for.

Can We Improve This?

SAT

- For satisfiable queries, the SAT solver returns with a satisfying assignment.
- All the values in this assignment are satisfiable and don't need to be queried for.

Example

if we have

$$SAT((x \Rightarrow y) \wedge y) = \{x, y\}$$

then

$$((x \Rightarrow y) \wedge x)$$

is satisfiable

Can We Improve This?

UNSAT

- Can a negative response of the solver help in the future?
- Example

$$\left.\begin{array}{l} f_1 => f_2 \\ \neg f_2 \\ \ldots \end{array}\right\} \neg f_1$$

- Recording disabled values *may* help with further queries.

Satisfiability with Caching

- *KnownValues* represent values known to be SAT
- *DisabledValues* represent values known to be UNSAT

```
$\textsc{Test-SAT}(\phi\colon \mathit{Formula}, l\colon \mathit{Literal}) : \mathit{Boolean}$
if $l \in \mathit{KnownValues}$ then return $\textsc{True}$
if $l \in \mathit{DisabledValues}$ then return $\textsc{False}$
$L \leftarrow \mathrm{SAT}(\phi \wedge l \wedge \bigwedge_{k \in \mathit{DisabledValues}} \neg k)$
if $L \neq$ null
    then $\mathit{KnownValues} \leftarrow \mathit{KnownValues} \cup L$
    else $\mathit{DisabledValues} \leftarrow \mathit{DisabledValues} \cup \{l\}$
return $L \neq$ null
```

Explanations

- The solver produces a unsatisfiable subset of given formulas.
- This may not be minimal, several techniques how to minimize.
- In the tool an iterative technique by Zhang and Malik.

How is a configuration process completed?

Conditions

- All variables must have a value.
- Values must conform to the constraints.

How is a configuration process completed?

Conditions

- All variables must have a value.
- Values must conform to the constraints.

Scenarios

- **manual** — user fills in everything

How is a configuration process completed?

Conditions

- All variables must have a value.
- Values must conform to the constraints.

Scenarios

- **manual** — user fills in everything
- **blind completion** — automated tool fills in everything

How is a configuration process completed?

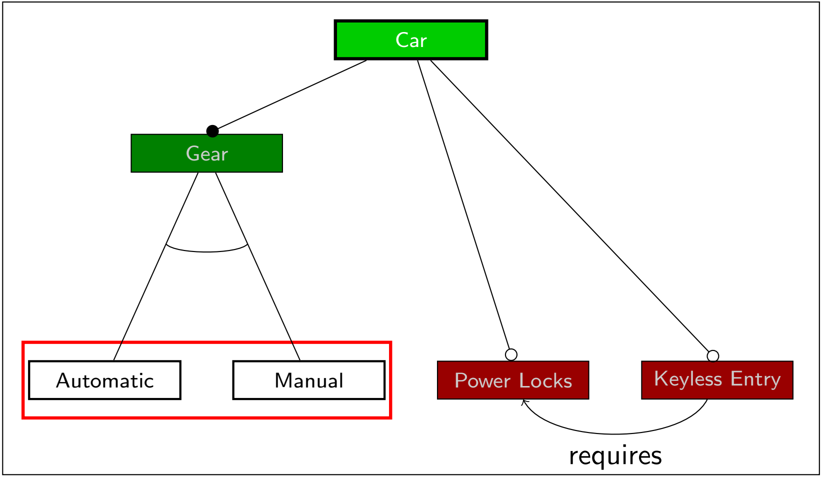
Conditions

- All variables must have a value.
- Values must conform to the constraints.

Scenarios

- **manual** — user fills in everything
- **blind completion** — automated tool fills in everything
- **smart completion** — automated tool fills as much as possible *without making decisions* for the user

Smart Completion



Car

Gear

Automatic

Manual

Power Locks

Keyless Entry

requires

Legend

Selection

Eliminate

Autoselect

Autoeliminate

Decisions and Dispensable Variables

- A set of variables is eliminable *iff* they can be all eliminated all at once.

$$\mathcal{D}(\phi, X) \stackrel{\text{def}}{=} SAT\left(\phi \wedge \bigwedge_{v \in X} \neg v\right)$$

Decisions and Dispensable Variables

- A set of variables is eliminable *iff* they can be all eliminated all at once.

$$\mathcal{D}(\phi, X) \stackrel{\mathrm{def}}{=} SAT\left(\phi \wedge \bigwedge_{v \in X} \neg v\right)$$

- A set of variables X must be decided *iff*
  1. X it is not eliminable.
  2. All of the proper subsets of X are eliminable.

$$\neg\mathcal{D}(\phi, X) \wedge (\forall Y \subsetneq X)(\mathcal{D}(\phi, Y))$$

Decisions and Dispensable Variables

- A set of variables is eliminable *iff* they can be all eliminated all at once.

$$\mathcal{D}(\phi, X) \stackrel{\text{def}}{=} SAT\left(\phi \wedge \bigwedge_{v \in X} \neg v\right)$$

- A set of variables X must be decided *iff*
    1. X it is not eliminable.
    2. All of the proper subsets of X are eliminable.

$$\neg\mathcal{D}(\phi, X) \wedge (\forall Y \subsetneq X)(\mathcal{D}(\phi, Y))$$

- A variable is dispensable *iff* it does not belong to any set that must be decided.

Examples

$$\mathbf{x} \vee \mathbf{y} \vee \mathbf{z}$$

- Deselectable: $\{x, y\}$, $\{x, z\}$, $\{y, z\}$, $\{x\}$, $\{y\}$, $\{z\}$, and $\emptyset$
- Not eliminable: $\{x, y, z\}$
- $\{x, y, z\}$ must be decided and none of the variables are dispensable.

Examples

$$\mathbf{x} \vee \mathbf{y} \vee \mathbf{z}$$

- Deselectable: $\{x, y\}$, $\{x, z\}$, $\{y, z\}$, $\{x\}$, $\{y\}$, $\{z\}$, and $\emptyset$
- Not eliminable: $\{x, y, z\}$
- $\{x, y, z\}$ must be decided and none of the variables are dispensable.

$$\mathbf{x} \Rightarrow (\mathbf{y} \vee \mathbf{z})$$

- $\{x, y, z\}$ is eliminable therefore does not have to be decided and all variables are dispensable.

Examples

$$\mathbf{x} \vee \mathbf{y} \vee \mathbf{z}$$

- Deselectable: $\{x, y\}$, $\{x, z\}$, $\{y, z\}$, $\{x\}$, $\{y\}$, $\{z\}$, and $\emptyset$
- Not eliminable: $\{x, y, z\}$
- $\{x, y, z\}$ must be decided and none of the variables are dispensable.

$$\mathbf{x} \Rightarrow (\mathbf{y} \vee \mathbf{z})$$

- $\{x, y, z\}$ is eliminable therefore does not have to be decided and all variables are dispensable.

$$(\mathbf{x} \Rightarrow (\mathbf{y} \vee \mathbf{z})) \wedge \mathbf{x}$$

- $\{y, z\}$ must be decided.

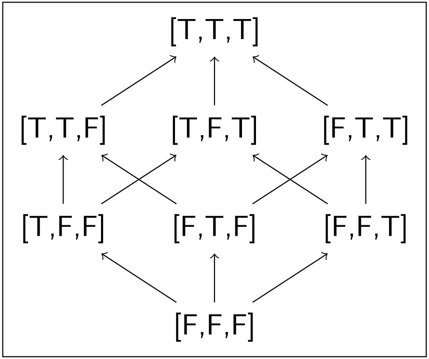
Relation to Minimal Models

- A model of a formula is (point-wise) minimal *iff* flipping some **T**rue values to **F**alse, yields a non-model.

Relation to Minimal Models

- A model of a formula is (point-wise) minimal *iff* flipping some **T**rue values to **F**alse, yields a non-model.

$\mathbf{x} \vee (\mathbf{y} \wedge \mathbf{z})$



[T,T,T]

[T,T,F] [T,F,T] [F,T,T]

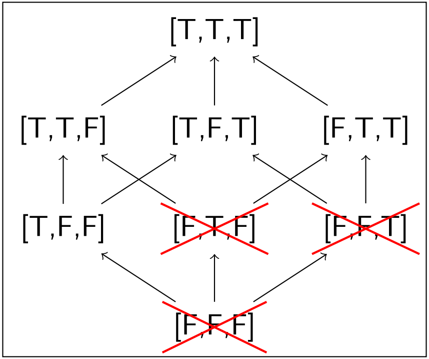
[T,F,F] [F,T,F] [F,F,T]

[F,F,F]

Relation to Minimal Models

- A model of a formula is (point-wise) minimal *iff* flipping some **T**rue values to **F**alse, yields a non-model.

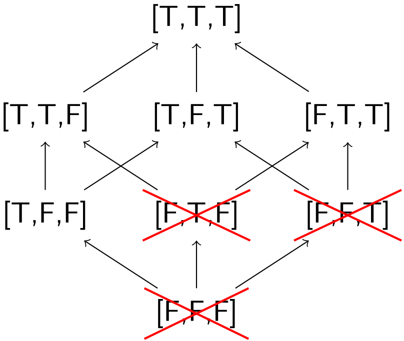
$\mathbf{x} \vee (\mathbf{y} \wedge \mathbf{z})$



Relation to Minimal Models

- A model of a formula is (point-wise) minimal *iff* flipping some **T**rue values to **F**alse, yields a non-model.

$\mathbf{x} \vee (\mathbf{y} \wedge \mathbf{z})$



[T,T,T]

[T,T,F] [T,F,T] [F,T,T]

[T,F,F] ~~[F,T,F]~~ ~~[F,F,T]~~

~~[F,F,F]~~

Dispensable Variables and Minimal Models

- A variable is dispensable *iff* it is False in all minimal models.

Dispensable Variables and Minimal Models

- A variable is dispensable *iff* it is False in all minimal models.

$$\mathbf{x} \vee \mathbf{y} \vee \mathbf{z}$$

x	**y**	**z**
T	F	F
F	T	F
F	F	T

Dispensable Variables and Minimal Models

- A variable is dispensable *iff* it is False in all minimal models.

$$\mathbf{x} \vee \mathbf{y} \vee \mathbf{z}$$

x	**y**	**z**
T	F	F
F	T	F
F	F	T

$$\mathbf{x} \Rightarrow (\mathbf{y} \vee \mathbf{z})$$

x	**y**	**z**
F	F	F

Enumerating All Minimal Models

schema

```
while ($\phi$ is satisfiable)
    do $M \leftarrow$ some minimal model of $\phi$
       remove from $\phi$ all models that
           are greater or equal to $M$
```

Other Relations

Non-monotonic Reasoning

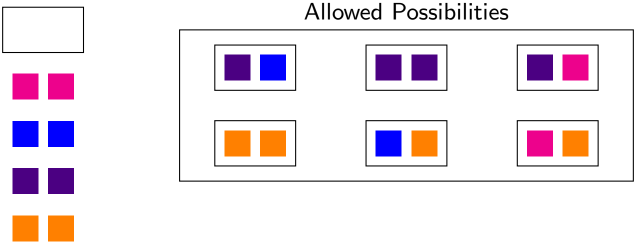
- Propositional Circumscription

$$\phi \models_{\min} \psi$$

- Generalized Closed World Assumption (GCWA)

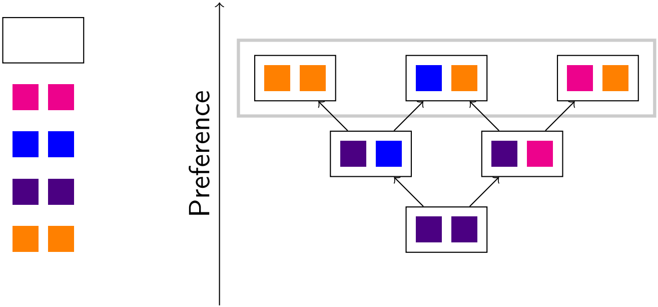
What about non-boolean configuration?

- For a general set of possibilities, it is hard to help the user.



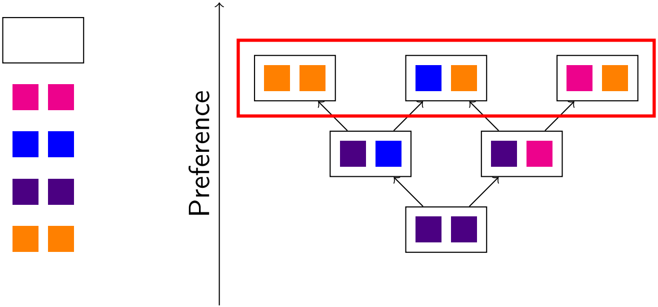
What about non-boolean configuration?

- For a general set of possibilities, it is hard to help the user.
- But it is possible, if there is a preference on the possibilities.



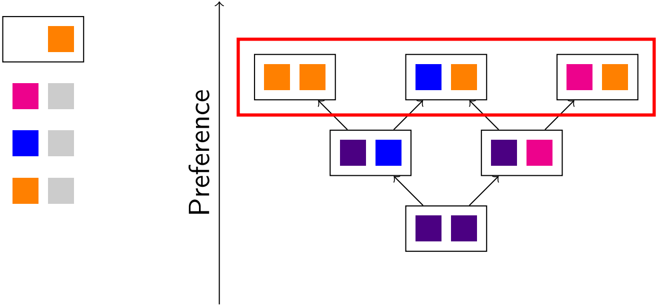
What about non-boolean configuration?

- For a general set of possibilities, it is hard to help the user.
- But it is possible, if there is a preference on the possibilities.
- We can focus on the most preferred possibilities.



What about non-boolean configuration?

- For a general set of possibilities, it is hard to help the user.
- But it is possible, if there is a preference on the possibilities.
- We can focus on the most preferred possibilities.



General Case Meets the Boolean Case

- A value c is settled for a variable v *iff* v has the value c in all most preferred possibilities.

General Case Meets the Boolean Case

- A value c is settled for a variable v *iff* v has the value c in all most preferred possibilities.

Settled Values and Dispensable Variables

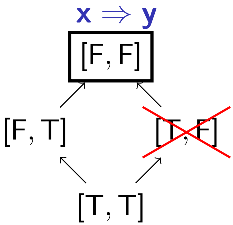
- A variable is dispensable iff False is settled for it under the point-wise preference for False.

General Case Meets the Boolean Case

- A value c is settled for a variable v *iff* v has the value c in all most preferred possibilities.

Settled Values and Dispensable Variables

- A variable is dispensable iff False is settled for it under the point-wise preference for False.

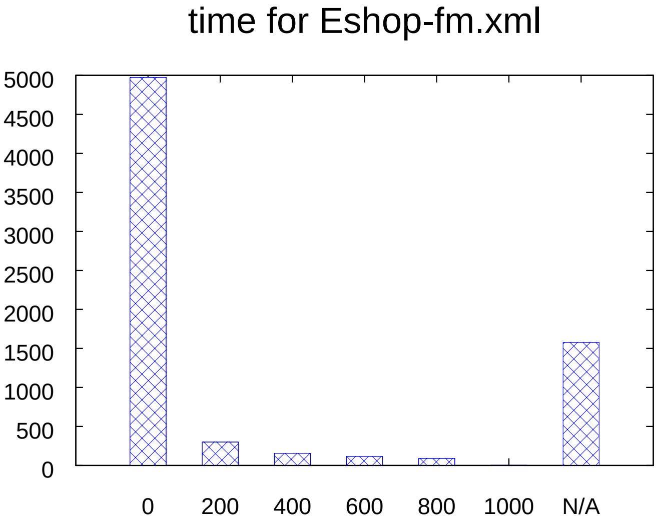


Experimental Results

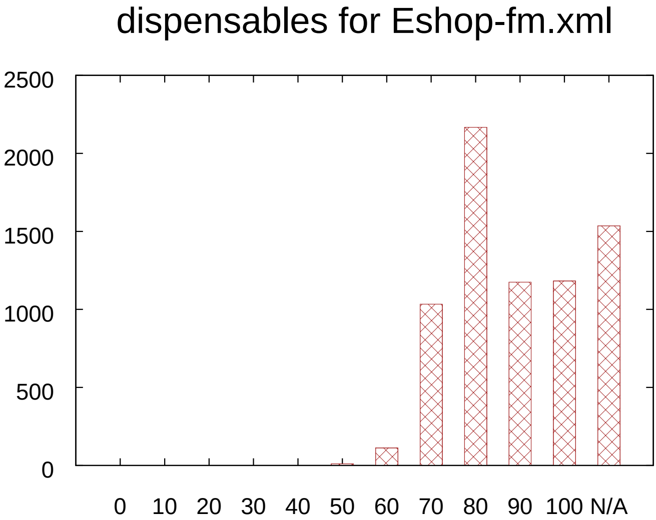
Name	#variables	#clauses
E-shop	287	420
Violet	170	341
Berkeley	94	183
T-shirt	16	40
3blocks	283	9690
rocket	351	2398

Name	Length	#decisions	noop	BIS	Cat	Ord	BCO
E-shop	144.3	7215	53	42	53	9	7
Violet	50.34	2517	16	15	16	4	4
Berkeley	25.12	1256	14	11	14	7	5
T-shirt	4.7	235	8	7	8	5	5
3blocks	6.7	335	655	616	557	3418	2671
rocket	7.52	376	166	163	158	274	260

Times for Minimal Model Enumeration



Percentages of Dispensable Variables



Summary

interactive configuration

- Interactive configuration is about helping a user to find a solution to the problem
- This is solved by multiple calls to a SAT solver

Summary

interactive configuration

- Interactive configuration is about helping a user to find a solution to the problem
- This is solved by multiple calls to a SAT solver

smart completion

- To provide smart completion we introduce dispensable variables.

Summary

interactive configuration

- Interactive configuration is about helping a user to find a solution to the problem
- This is solved by multiple calls to a SAT solver

smart completion

- To provide smart completion we introduce dispensable variables.
- Dispensable variables are closely related to CWA.

Summary

interactive configuration

- Interactive configuration is about helping a user to find a solution to the problem
- This is solved by multiple calls to a SAT solver

smart completion

- To provide smart completion we introduce dispensable variables.
- Dispensable variables are closely related to CWA.
- In non-propositional case we needed preference.

Summary

interactive configuration

- Interactive configuration is about helping a user to find a solution to the problem
- This is solved by multiple calls to a SAT solver

smart completion

- To provide smart completion we introduce dispensable variables.
- Dispensable variables are closely related to CWA.
- In non-propositional case we needed preference.
- Dispensable variables can be seen as a *preference for deselecting*.
- Analogously CWA as a preference for False.

- SAT solvers are fast enough to be used as oracles in more challenging problems