CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Lecture 5: Analysis of Feature Models

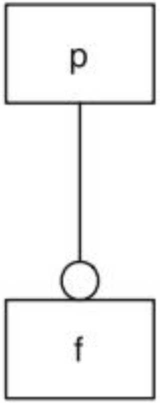
Gregory Gay

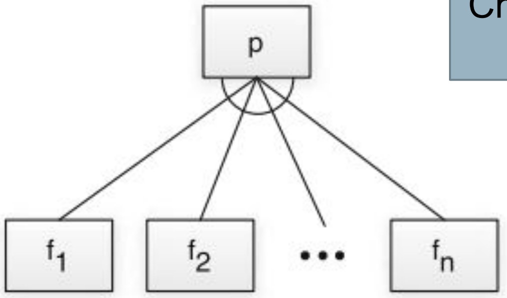
TDA/DIT 594 - November 17, 2020

Feature Diagrams

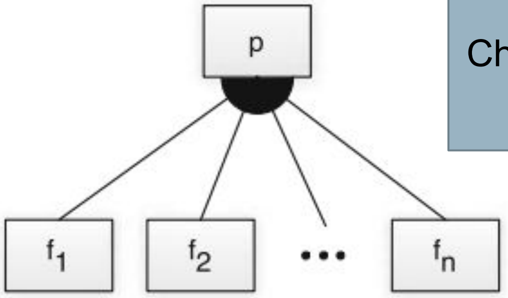
Mandatory Feature

Optional Feature



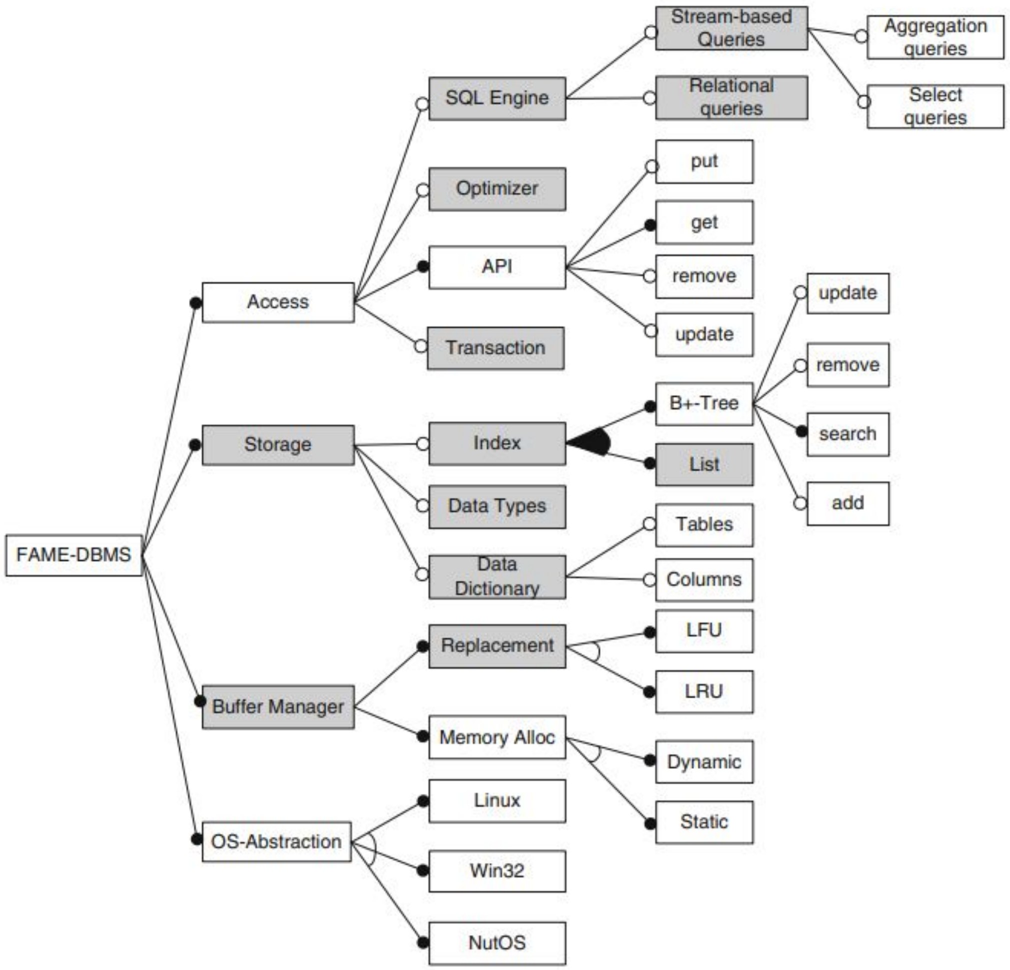


Mutually Exclusive Choice: Choose **exactly** one (**xor**)



Choose **at least** one (**or**)

Example - Data Management

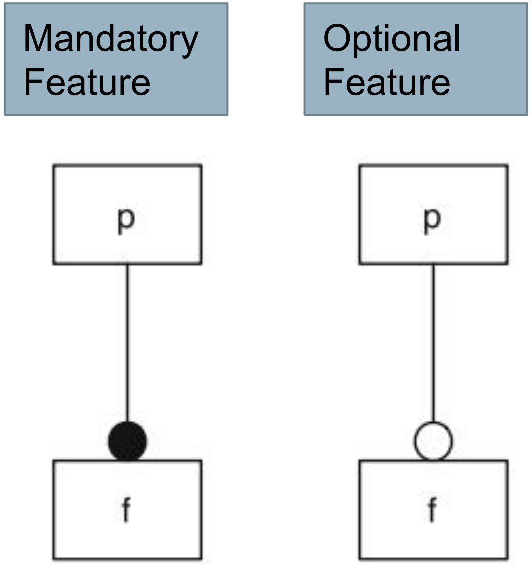


Propositional Logic

- **Cross-tree Constraints** are predicates imposing constraints between features.
  - `DataDictionary` $\Rightarrow$ `String`
    - (Storing a data dictionary **requires** support for strings)
  - `MinimumSpanningTree` $\Rightarrow$ `Undirected` $\land$ `Weighted`
    - (Computing a Minimum Spanning Tree **requires** support for undirected **and** weighted edges)
  - Constraints over Boolean variables and subexpressions.
    - (i.e., `(NumProcesses >= 5)`)

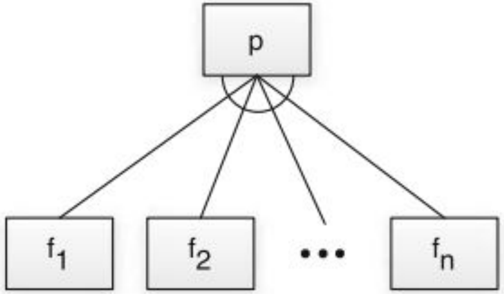
Propositional Logic

- **Mandatory:** If parent is selected, the child must be.
  - `mandatory(p, f) ≡ f ⇔ p`
- **Optional:** Child may only be chosen if the parent is.
  - `optional(p, f) ≡ f ⇒ p`

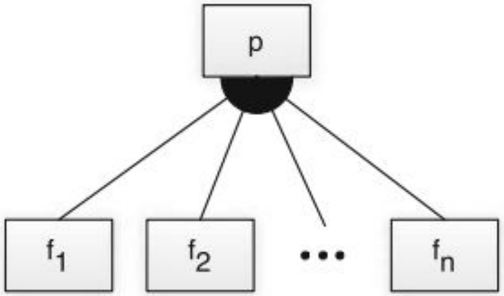


Propositional Logic

- **Alternative:** Choose **exactly** one
  - alternative(p, $\{f_1,...,f_n\}$) ≡ $((f_1 \lor \dots \lor f_n) \Leftrightarrow p) \land_{(fi,fj)} \neg(f_i \land f_j)$



- **Or:** Choose **at least** one
  - or(p, $\{f_1,...,f_n\}$) ≡ $((f_1 \lor \dots \lor f_n) \Leftrightarrow p)$



Today's Goals

- Analysis of Feature Models
- Introduction to Boolean Satisfiability (SAT)
  - SAT Solvers

Analysis of Feature Models

Variability-Aware Analysis

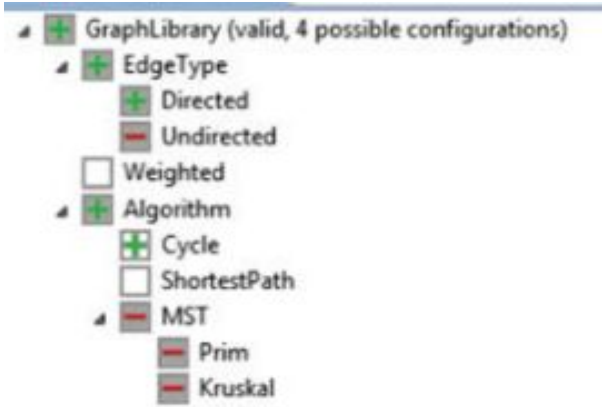
- Verification techniques do not extend to SPLs.
  - More product variations than atoms in the universe.
- Sometimes, can restrict to subset (HP printers).
- **Variability-Aware Analyses** can examine whole product line (or reasonable subset).

Analyses of Feature Models

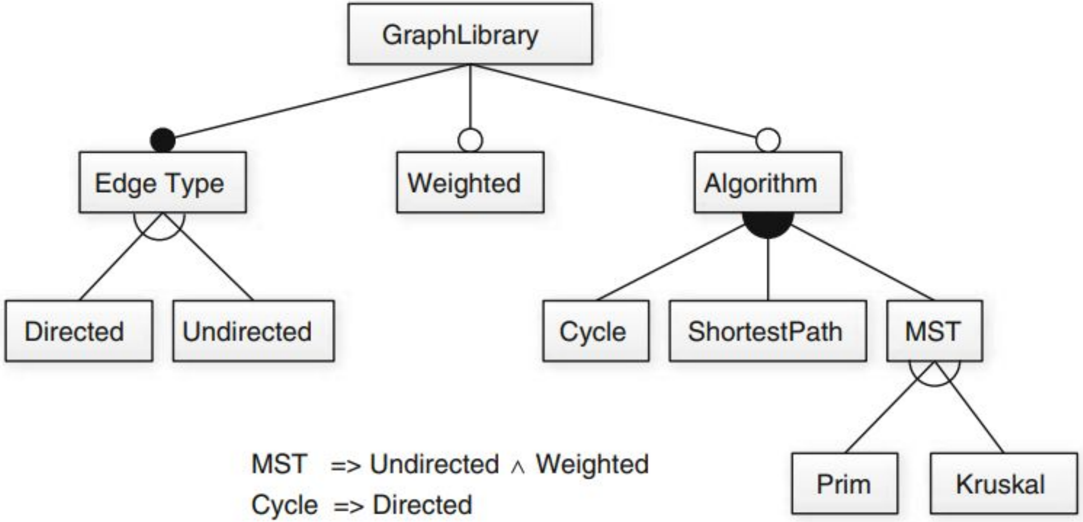
- Is a feature selection valid?
- Is the feature model consistent?
- Do our assumptions hold (testing)?
- Which features are mandatory?
- Which features can never be selected (dead)?
- How many valid selections does model have?
- Are two models equivalent?
- Given partial selection, what must be included?
- What selections give best cost/size/performance?

Valid Feature Selection

- Translate model into a propositional formula φ.
- Assign `true` to each selected feature, `false` to rest.
- Assess whether φ is `true`.
  - If yes, valid selection.

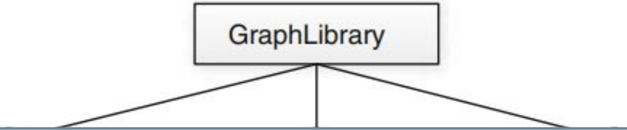


Example - Graph Library



$$
\begin{aligned}
\phi = & \texttt{GraphLibrary} \land \texttt{EdgeType} \land (\texttt{Directed} \lor \texttt{Undirected}) \land \neg(\texttt{Directed} \land \texttt{Undirected}) \\
& \land ((\texttt{Cycle} \lor \texttt{ShortestPath} \lor \texttt{MST}) \Leftrightarrow \texttt{Algorithm}) \land (\texttt{Cycle} \Rightarrow \texttt{Directed}) \\
& \land ((\texttt{Prim} \lor \texttt{Kruskal}) \Leftrightarrow \texttt{MST}) \land \neg(\texttt{Prim} \land \texttt{Kruskal}) \land (\texttt{MST} \Rightarrow (\texttt{Undirected} \land \texttt{Weighted}))
\end{aligned}
$$

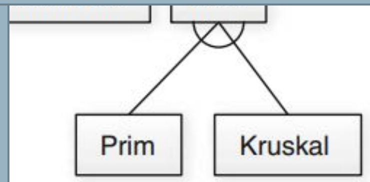
Example - Graph Library

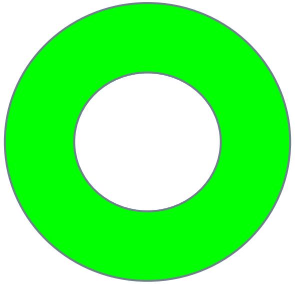


Selection:
{GraphLibrary, EdgeType, Directed}

$$\varphi = T \wedge T \wedge (T \vee F) \wedge \neg(T \wedge F) \wedge ((F \vee F \vee F) \Leftrightarrow F) \wedge (F \Rightarrow F) \wedge ((F \vee F) \Leftrightarrow F) \wedge \neg(F \wedge F) \wedge (F \Rightarrow (F \wedge F))$$

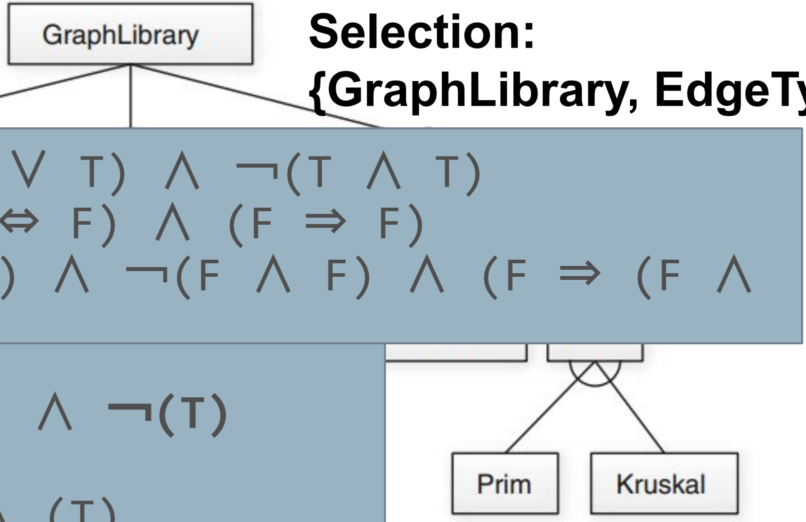
$$\varphi = T \wedge T \wedge (T) \wedge \neg(F) \wedge (T) \wedge (T) \wedge (T) \wedge \neg(F) \wedge (T)$$





$$\phi = GraphLibrary \wedge EdgeType \wedge (Directed \vee Undirected) \wedge \neg(Directed \wedge Undirected) \wedge ((Cycle \vee ShortestPath \vee MST) \Leftrightarrow Algorithm) \wedge (Cycle \Rightarrow Directed) \wedge ((Prim \vee Kruskal) \Leftrightarrow MST) \wedge \neg(Prim \wedge Kruskal) \wedge (MST \Rightarrow (Undirected \wedge Weighted))$$

Example - Graph Library



Selection:
{GraphLibrary, EdgeType, Directed, Undirected}

$$\varphi = T \land T \land (T \lor T) \land \neg(T \land T) \land ((F \lor F \lor F) \Leftrightarrow F) \land (F \Rightarrow F) \land ((F \lor F) \Leftrightarrow F) \land \neg(F \land F) \land (F \Rightarrow (F \land F))$$

$$\varphi = T \land T \land (T) \land \neg(T) \land (T) \land (T) \land (T) \land \neg(F) \land (T)$$

$$\phi = \mathtt{GraphLibrary} \land \mathtt{EdgeType} \land (\mathtt{Directed} \lor \mathtt{Undirected}) \land \neg(\mathtt{Directed} \land \mathtt{Undirected})$$
$$\land ((\mathtt{Cycle} \lor \mathtt{ShortestPath} \lor \mathtt{MST}) \Leftrightarrow \mathtt{Algorithm}) \land (\mathtt{Cycle} \Rightarrow \mathtt{Directed})$$
$$\land ((\mathtt{Prim} \lor \mathtt{Kruskal}) \Leftrightarrow \mathtt{MST}) \land \neg(\mathtt{Prim} \land \mathtt{Kruskal}) \land (\mathtt{MST} \Rightarrow (\mathtt{Undirected} \land \mathtt{Weighted}))$$

Consistent Feature Models

- A **consistent** model has 1+ valid selections.
  - **Inconsistent** models do not have any valid selection.
- Contradictory constraints are common.
- Find feature selection that results in φ = `true`
  - NP-complete problem, but SAT solvers can often find solutions quickly.

Boolean Satisfiability (SAT)

- Find assignments to Boolean variables $X_1, X_2, \ldots, X_n$ that results in expression φ evaluating to true.
- Defined over expressions written in **conjunctive normal form**.
  - $\varphi = (X_1 \lor \neg X_2) \land (\neg X_1 \lor X_2)$
  - $(X_1 \lor \neg X_2)$ is a **clause**, made of variables, ¬, ∨
  - Clauses are joined with ∧

Conjunctive Normal Form

- Variables: X_1, X_2, X_3, X_4, X_5
- Clauses (using only ∨ (or) and ¬ (not)):
  - $(\neg X_2 \lor X_5)$, $(X_1 \lor \neg X_3 \lor X_4)$, $(X_4 \lor \neg X_5)$, $(X_1 \lor X_2)$
- Expression φ joins clauses with ∧ (and)
  - $(\neg X_2 \lor X_5) \land (X_1 \lor \neg X_3 \lor X_4) \land (X_4 \lor \neg X_5) \land (X_1 \lor X_2)$

Boolean Satisfiability

- Find assignment to X_1, X_2, X_3, X_4, X_5 to solve
  - $(\neg X_2 \lor X_5) \land (X_1 \lor \neg X_3 \lor X_4) \land (X_4 \lor \neg X_5) \land (X_1 \lor X_2)$
- One solution: 1, 0, 1, 1, 1
  - $(\neg X_2 \lor X_5) \land (X_1 \lor \neg X_3 \lor X_4) \land (X_4 \lor \neg X_5) \land (X_1 \lor X_2)$
  - $(\neg 0 \lor 1) \land (1 \lor \neg 1 \lor 1) \land (1 \lor \neg 1) \land (1 \lor 0)$
  - $(1) \land (1) \land (1) \land (1)$
  - 1

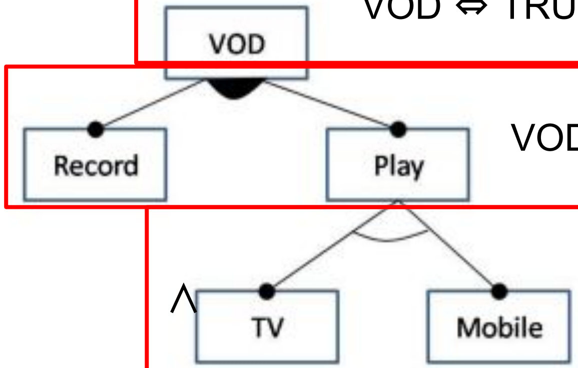
Transformation Rules

- De Morgan's Laws
  - $\neg(X \lor Y) \equiv \neg X \land \neg Y$
  - $\neg(X \land Y) \equiv \neg X \lor \neg Y$
- Distributivity
  - $X \lor (Y \land Z) \equiv (X \lor Y) \land (X \lor Z)$
  - $X \land (Y \lor Z) \equiv (X \land Y) \lor (X \land Z)$
- Double Negation
  - $\neg\neg X \equiv X$

Transformation Rules

- $X \Leftrightarrow Y$
  - X is equivalent to Y
- $\equiv (X \Rightarrow Y) \land (Y \Rightarrow X)$
  - $(X \Rightarrow Y) \equiv (\neg X \lor Y)$
  - If X is true, Y is also true.
  - If X is false, Y can be either true or false.
- $\equiv (\neg X \lor Y) \land (\neg Y \lor X)$

Transformation into CNF



$VOD \Leftrightarrow TRUE \equiv VOD$

$VOD \Leftrightarrow (Record \lor Play)$

$((Mobile \lor TV) \Leftrightarrow Play)$

$\land$

$\neg(Movile \land TV) \equiv$

$(Mobile \Leftrightarrow Play \land \neg TV) \land (TV \Leftrightarrow Play \land \neg Mobile)$

mandatory(p, f) $\equiv f \Leftrightarrow p$

optional(p, f) $\equiv f \Rightarrow p$

alternative(p, $\{f_1,\ldots,f_n\}$) $\equiv$
$((f_1 \lor \ldots \lor f_n) \Leftrightarrow p) \land \forall(f_i,f_j)\, \neg(f_i \land f_j)$

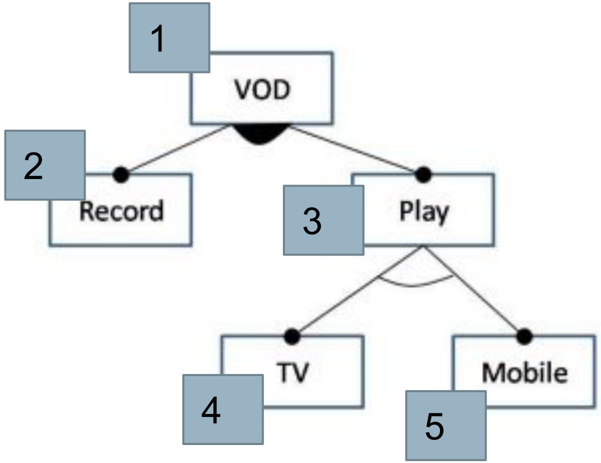
or(p, $\{f_1,\ldots,f_n\}$) $\equiv ((f_1 \lor \ldots \lor f_n) \Leftrightarrow p)$

$VOD \land (VOD \Leftrightarrow (Record \lor Play)) \land (Mobile \Leftrightarrow Play \land \neg TV) \land (TV \Leftrightarrow Play \land \neg Mobile)$

Transformation into CNF

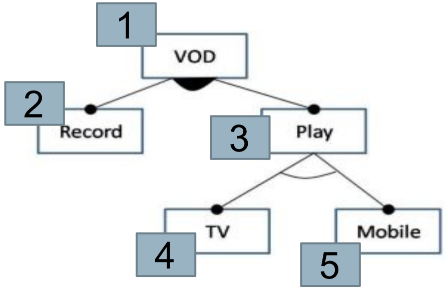
- VOD ∧ (VOD ⇔ (Record ∨ Play)) ∧ (Mobile ⇔ (Play ∧ ¬TV)) ∧ (TV ⇔ (Play ∧ ¬Mobile))
  - (VOD ⇔ (Record ∨ Play))
    - ≡ (VOD ⇒ (Record ∨ Play)) ∧ ((Record ∨ Play) ⇒ VOD)
    - ≡ (¬VOD ∨ (Record ∨ Play)) ∧ (¬(Record ∨ Play) ∨ VOD)
    - ≡ (¬VOD ∨ (Record ∨ Play)) ∧ (¬Record ∨ VOD) ∧ (¬Play ∨ VOD)
  - (Mobile ⇔ (Play ∧ ¬TV))
    - ≡ (Mobile ∨ TV ∨ ¬Play) ∧ (¬Mobile ∨ Play) ∧ (¬Mobile ∨ ¬TV)
  - (TV ⇔ (Play ∧ ¬Mobile))
    - ≡ (TV ∨ Mobile ∨ ¬Play) ∧ (¬TV ∨ Play) ∧ (¬TV ∨ ¬Mobile)

DIMACS Format



- Map feature names to integer IDs.
  - VOD = 1
  - Record = 2
  - Play = 3
  - TV = 4
  - Mobile = 5

DIMACS Format



VOD ∧

(¬VOD ∨ (Record ∨ Play)) ∧ (¬Record ∨ VOD) ∧ (¬Play ∨ VOD) ∧

(Mobile ∨ TV ∨ ¬Play) ∧ (¬Mobile ∨ Play) ∧ (¬Mobile ∨ ¬TV) ∧

(TV ∨ Mobile ∨ ¬Play) ∧ (¬TV ∨ Play) ∧ (¬TV ∨ ¬Mobile)

1 ∧

(¬1 ∨ (2 ∨ 3)) ∧ (¬2 ∨ 1) ∧ (¬3∨ 1) ∧

(5 ∨ 4 ∨ ¬3) ∧ (¬5 ∨ 3) ∧ (¬5 ∨ ¬4) ∧

(4 ∨ 5 ∨ ¬3) ∧ (¬4 ∨ 3) ∧ (¬4∨ ¬5)

DIMACS Format

1 ∧
(¬1 ∨ (2 ∨ 3)) ∧ (¬2 ∨ 1) ∧ (¬3 ∨ 1)
∧
(5 ∨ 4 ∨ ¬3) ∧ (¬5 ∨ 3) ∧ (¬5 ∨
¬4) ∧
(4 ∨ 5 ∨ ¬3) ∧ (¬4 ∨ 3) ∧ (¬4 ∨
¬5)

- Each clause is stored in a row, with ∧(AND) omitted.
- Negation (¬) translated into negative (-)

1
-1 ∨ 2 ∨ 3
-2 ∨ 1
-3 ∨ 1
5 ∨ 4 ∨ -3
-5 ∨ 3
-5 ∨ -4
4 ∨ 5 ∨ -3
-4 ∨ 3
-4 ∨ -5

```
1
-1 ∨ 2 ∨ 3
-2 ∨ 1
-3 ∨ 1
5 ∨ 4 ∨ -3
-5 ∨ 3
-5 ∨ -4
4 ∨ 5 ∨ -3
-4 ∨ 3
-4 ∨ -5
```

- Remove disjunction signs (∨)
- Add DIMACs header
  - Comments
  - Indicates CNF format
  - Number of variables
  - Number of CNF clauses

```
c comments
p cnf 5 10
1
-1 2 3
-2 1
-3 1
5 4 -3
-5 3
-5 -4
4 5 -3
-4 3
-4-5
```

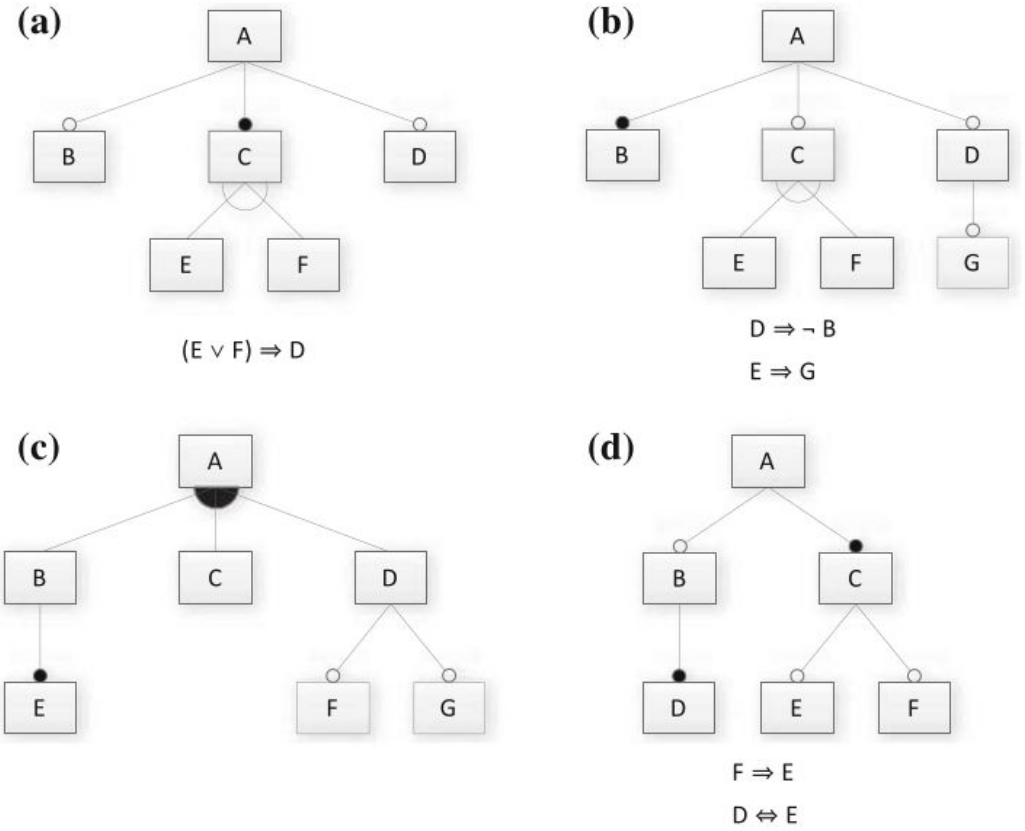
Using a SAT Solver

- Identify assignment that results in true outcome.
  - VOD ∧ (¬VOD ∨ (Record ∨ Play)) ∧ (¬Record ∨ VOD) ∧ (¬Play ∨ VOD) ∧ (Mobile ∨ TV ∨ ¬Play) ∧ (¬Mobile ∨ Play) ∧ (¬Mobile ∨ ¬TV) ∧ (TV ∨ Mobile ∨ ¬Play) ∧ (¬TV ∨ Play) ∧ (¬TV ∨ ¬Mobile)
  - A satisfying assignment: (1, 1, 1, 1, 0)
- Returns satisfying assignment.
  - May return all satisfying assignments found.
  - If not satisfiable, may offer information on why.

Let's take a break!

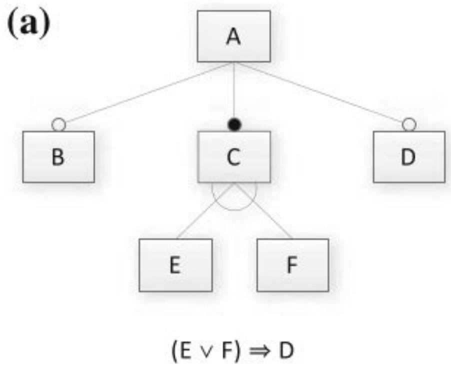
Activity

- Start with A/B.
    - Do C/D if time.
- Translate model into propositional logic formula.
- Provide two valid and two invalid features.
- Is it consistent? If not, why not?



Solution (A)

- Translate model into propositional logic formula.
- Provide two valid and two invalid features.
- Is it consistent? If not, why not?

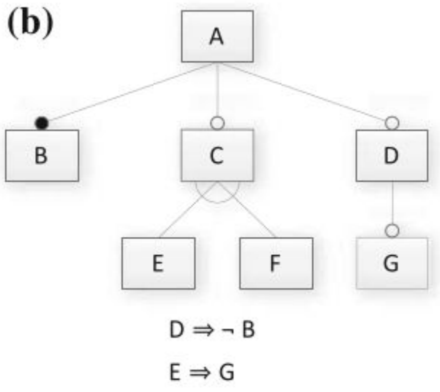


A ∧ (B ⇒ A) ∧ (C ⇔ A) ∧ (D ⇒ A) ∧
((C ⇔ (E ∨ F)) ∧ ¬(E ∧ F)) ∧ ((E ∨ F) ⇒ D))

- Valid: A, B, C, D, F ; A, C, D, E
- Invalid: A, B, C, D, E, F ; A, B, C, E
- Is it consistent: Yes

Solution (B)

- Translate model into propositional logic formula.
- Provide two valid and two invalid features.
- Is it consistent? If not, why not?



A ∧ (B ⇔ A) ∧ (C ⇒ A) ∧ (D ⇒ A) ∧
((C ⇔ (E ∨ F)) ∧ ¬(E ∧ F)) ∧ (G ⇒ D) ∧ (D ⇒ ¬B)
∧
(E ⇒ G)

- Valid: A, B ; A, B, C, F
- Invalid: A, B, D, G ; A, B, C, E
- It is consistent: Yes, but D, E, and G are dead features (because B is mandatory).

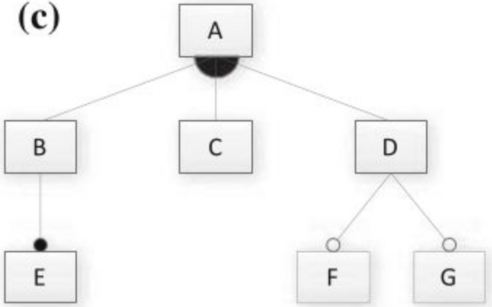
Solution (C)

- Translate model into propositional logic formula.
- Provide two valid and two invalid features.
- Is it consistent? If not, why not?

A ∧ ((B ∨ C ∨ D) ⇔ A) ∧ (E ⇔ B) ∧ (F ⇒ D) ∧ (G ⇒ D)

- Valid: A, C ; A, B, C, D, E, F, G
- Invalid: A, B, C; A, C, E
- It is consistent: Yes (just remember that B and E need to come as a pair)

(c)

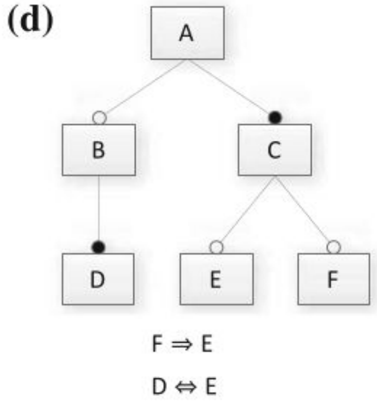


Solution (D)

- Translate model into propositional logic formula.
- Provide two valid and two invalid features.
- Is it consistent? If not, why not?

$A \land (B \Rightarrow A) \land (C \Leftrightarrow A) \land (D \Leftrightarrow B) \land (E \Rightarrow C) \land (F \Rightarrow C) \land$
$(F \Rightarrow E) \land (D \Leftrightarrow E)$

- Valid: A, C ; A, B, C, D, E
- Invalid: A, B, C, D ; A, C, F
- It is consistent: Yes, but remember that if you have F, you need E, D, and B as well.



SAT Solver Process

- Express in conjunctive normal form:
  - $\varphi = (\neg x2 \lor x5) \land (x1 \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (x1 \lor x2)$
- Choose assignment based on how it affects each clause it appears in.
  - What happens if we assign x2 = true?
  - If any clauses now false, don’t apply that value.
  - Continue until CNF expression is satisfied.

Branch & Bound Algorithm

- Set variable to true or false.
- Apply that value.
- Does value satisfy the clauses that it appears in?
  - If so, assign a value to the next variable.
  - If not, backtrack (bound) and apply the other value.
- Prunes branches of the boolean decision tree as values are applied.

Branch & Bound Algorithm

$\varphi = (\neg x2 \lor x5) \land (x1 \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (x1 \lor x2)$

1. **Set x1 to false.**
   $\varphi = (\neg x2 \lor x5) \land (\mathbf{0} \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (\mathbf{0} \lor x2)$
2. **Set x2 to false.**
   $\varphi = (\mathbf{1} \lor x5) \land (\mathbf{0} \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (\mathbf{0} \lor \mathbf{0})$
3. **Backtrack and set x2 to true.**
   $\varphi = (\mathbf{0} \lor x5) \land (\mathbf{0} \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (\mathbf{0} \lor \mathbf{1})$

DPLL Algorithm

- Set a variable to true/false.
  - Apply that value to the expression.
  - Remove all satisfied clauses.
  - If assignment does not satisfy a clause, then remove that variable from that clause.
  - If this leaves any **unit clauses** (single variable clauses), assign a value that removes those next.
- Repeat until a solution is found.

DPLL Algorithm

$\varphi = (\neg x2 \lor x5) \land (x1 \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (x1 \lor x2)$

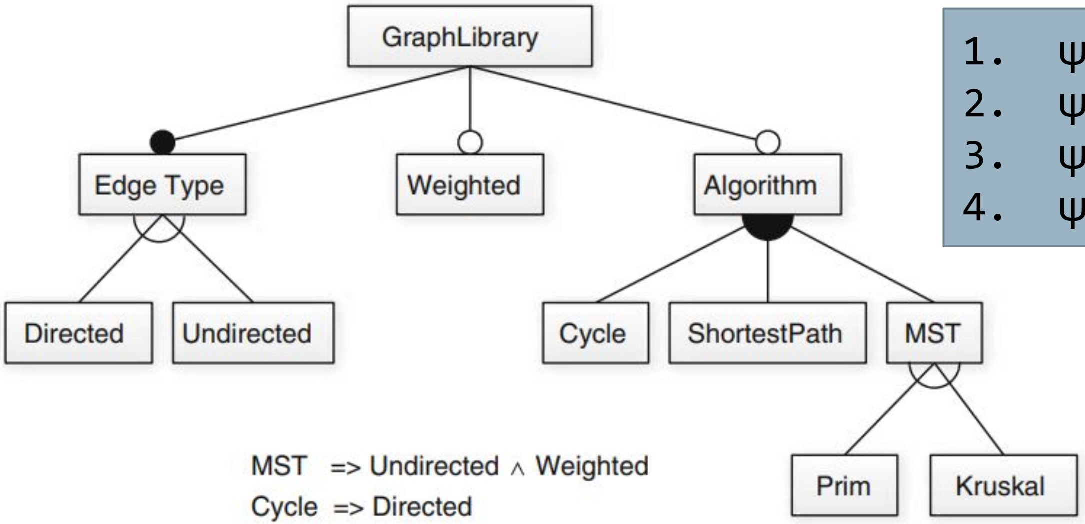
1. **Set x2 to false.**
   $\varphi = (\neg \mathbf{0} \lor x5) \land (x1 \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (x1 \lor \mathbf{0})$
   $\varphi = (x1 \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (x1)$
2. **Set x1 to true.**
   $\varphi = (\mathbf{1} \lor \neg x3 \lor x4) \land (x4 \lor \neg x5) \land (\mathbf{1})$
   $\varphi = (x4 \lor \neg x5)$
3. **Set x4 to false, then x5 to false.**
   $\varphi = (\mathbf{0} \lor \neg x5)$
   $\varphi = (\neg \mathbf{0})$

Let’s take a break!

Testing Facts about Models

- Encode a fact that *should be true* as propositional formula ψ.
- Check whether $\varphi \land \neg\psi$ is satisfiable.
  - Is there a valid feature selection for φ that does not satisfy the constraint ψ?
  - If yes, there is a problem with the model.

Example - Graph Library



1. ψ = Kruskal ⇒ Weighted
2. ψ = Prim ⇒ Weighted
3. ψ = ¬(Prim ∧ Kruskal)
4. ψ = Weighted ⇒ MST

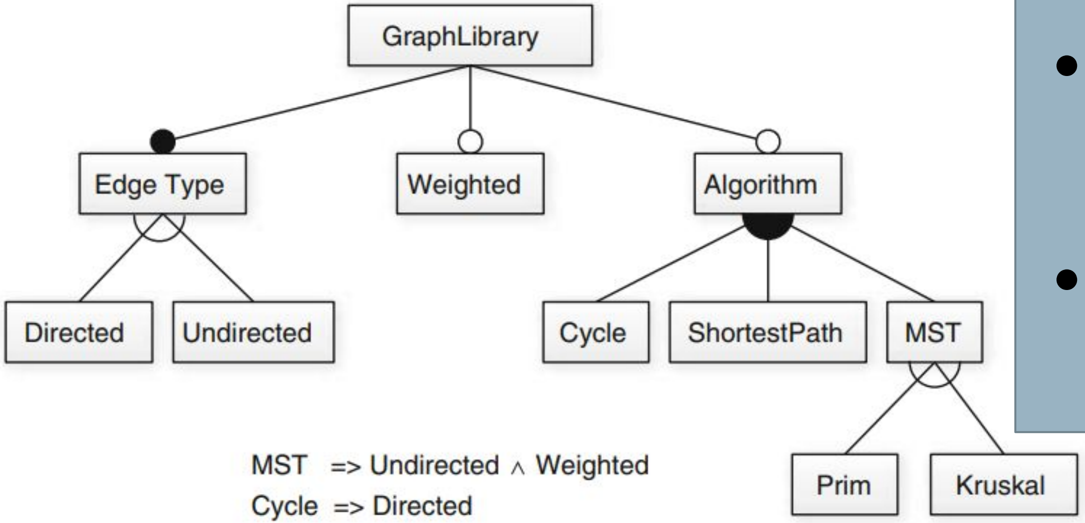
For each fact, assess whether (φ ∧ ¬ψ) is satisfiable.

$$\begin{aligned}\phi = &\texttt{GraphLibrary} \land \texttt{EdgeType} \land (\texttt{Directed} \lor \texttt{Undirected}) \land \neg(\texttt{Directed} \land \texttt{Undirected}) \\ &\land ((\texttt{Cycle} \lor \texttt{ShortestPath} \lor \texttt{MST}) \Leftrightarrow \texttt{Algorithm}) \land (\texttt{Cycle} \Rightarrow \texttt{Directed}) \\ &\land ((\texttt{Prim} \lor \texttt{Kruskal}) \Leftrightarrow \texttt{MST}) \land \neg(\texttt{Prim} \land \texttt{Kruskal}) \land (\texttt{MST} \Rightarrow (\texttt{Undirected} \land \texttt{Weighted}))\end{aligned}$$

Dead and Mandatory Features

- A **dead** feature is never used.
- A **mandatory** feature is always used.
- Given model φ and feature F:
  - 1+ valid selection **with F if (φ ∧ F)** is satisfiable.
  - 1+ valid selection **without F if (φ ∧ ¬F)** is satisfiable.
  - Feature is dead if no selection with it (**¬(φ ∧ F)**)
  - Feature is mandatory if no selection without it (**¬(φ ∧ ¬F)**)

Example - Graph Library



- No dead features.
  - If Undirected made mandatory, Directed and Cycle would be dead.
- GraphLibrary and EdgeType are mandatory.

$$\phi = \mathtt{GraphLibrary} \wedge \mathtt{EdgeType} \wedge (\mathtt{Directed} \vee \mathtt{Undirected}) \wedge \neg(\mathtt{Directed} \wedge \mathtt{Undirected})$$
$$\wedge ((\mathtt{Cycle} \vee \mathtt{ShortestPath} \vee \mathtt{MST}) \Leftrightarrow \mathtt{Algorithm}) \wedge (\mathtt{Cycle} \Rightarrow \mathtt{Directed})$$
$$\wedge ((\mathtt{Prim} \vee \mathtt{Kruskal}) \Leftrightarrow \mathtt{MST}) \wedge \neg(\mathtt{Prim} \wedge \mathtt{Kruskal}) \wedge (\mathtt{MST} \Rightarrow (\mathtt{Undirected} \wedge \mathtt{Weighted}))$$

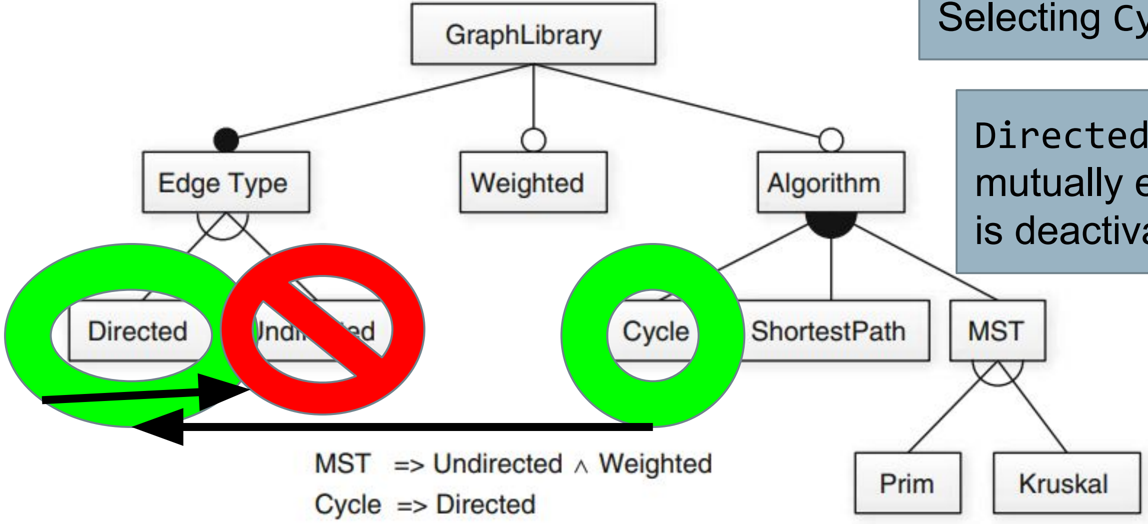
Constraint Propagation

- **Constraint Propagation** - hiding unavailable features after we make **partial selections**.
- Feature selection often iterative:
  - Feature selected, deselected, or no decision made.
- Partial feature selection:
  - Set of selected features ($S \subseteq F$)
  - Set of deselected features ($D \subseteq F$, with $S \cap D = \emptyset$)

Constraint Propagation

- Partial feature selection
  - $pfs(S,D) = \forall(s \in S)\ s \land \forall(d \in D)\ \neg d$
- Partial selection is valid if **($\varphi \land pfs(S,D)$)** satisfiable
- Feature F **deactivated** if **($\varphi \land pfs(S,D) \land F$)** is **not** satisfiable.
- Feature F **activated** if **($\varphi \land pfs(S,D) \land \neg F$)** is **not** satisfiable.

Example - Graph Library



Selecting `Cycle` activated `Directed`.

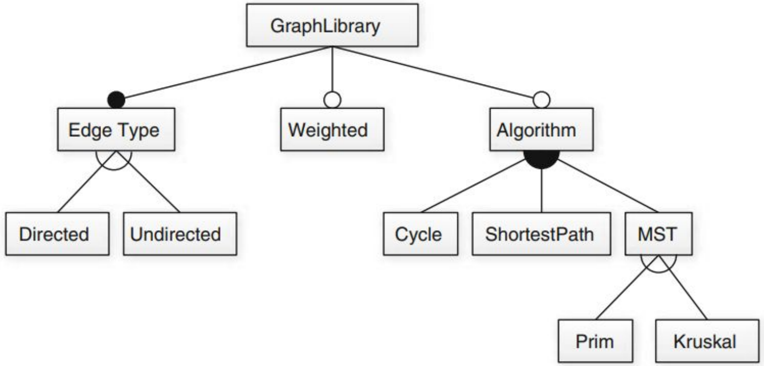
`Directed` and `Undirected` are mutually exclusive, so `Undirected` is deactivated.

$$\phi = \texttt{GraphLibrary} \wedge \texttt{EdgeType} \wedge (\texttt{Directed} \vee \texttt{Undirected}) \wedge \neg(\texttt{Directed} \wedge \texttt{Undirected})$$
$$\wedge ((\texttt{Cycle} \vee \texttt{ShortestPath} \vee \texttt{MST}) \Leftrightarrow \texttt{Algorithm}) \wedge (\texttt{Cycle} \Rightarrow \texttt{Directed})$$
$$\wedge ((\texttt{Prim} \vee \texttt{Kruskal}) \Leftrightarrow \texttt{MST}) \wedge \neg(\texttt{Prim} \wedge \texttt{Kruskal}) \wedge (\texttt{MST} \Rightarrow (\texttt{Undirected} \wedge \texttt{Weighted}))$$

Number of Valid Selections

- Upper bound counted recursively:
  - *count* root(c) = *count*(c)
  - *count* mandatory(c) = *count*(c)
  - *count* optional(c) = *count*(c) + 1
  - *count* and(c_1,..., c_n) = *count*(c_1) ∗ ... ∗ *count*(c_n)
  - *count* alternative(c_1,..., c_n) = *count*(c_1) + ... + *count*(c_n)
  - *count* or(c_1,..., c_n) = (*count*(c_1) + 1) ∗ ... ∗ (*count*(c_n) + 1) − 1
  - *count* leaf = 1

Example - Graph Library



count(f) = 1 //for all leaf nodes

count(EdgeType) = count(Directed) + count(Undirected)
= 1 + 1 = 2

count(MST) = count(Prim) + count(Kruskal)
= 1 + 1 = 2

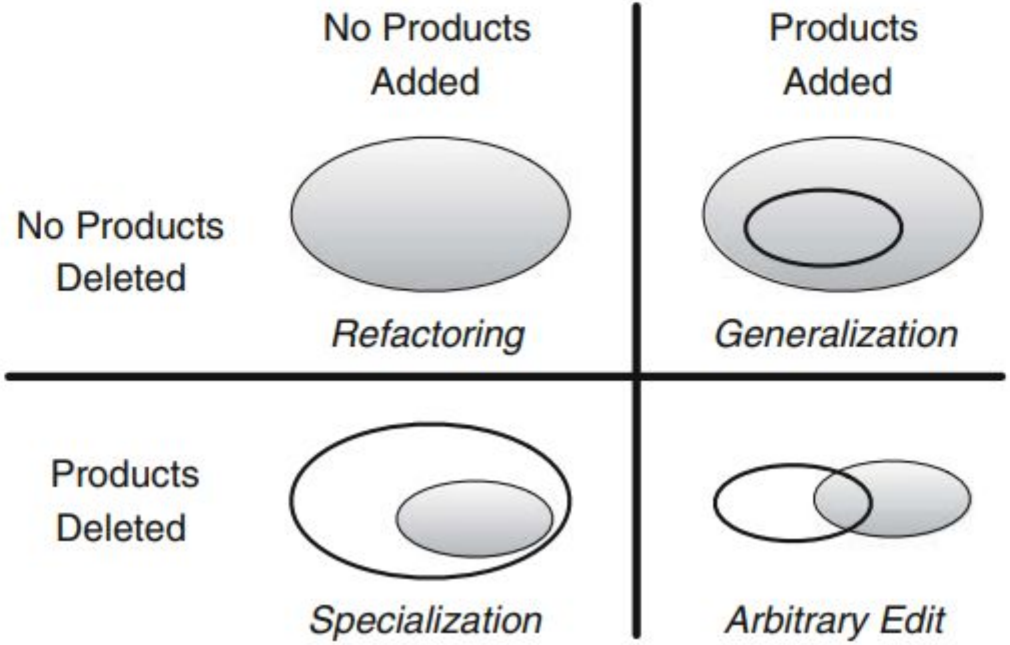
count(Algorithm) = (count(Cycle) + 1) ∗
(count(ShortestPath) + 1) ∗ (count(MST) + 1) − 1
= (1 + 1) ∗ (1 + 1) ∗ (2 + 1) − 1 = 11

count(GraphLibrary) = count(Mandatory(EdgeType)) ∗
count(Optional(Weighted)) ∗
count(Optional(Algorithm))
= 2 ∗ (1 + 1) ∗ (11 + 1) = 48

Number of Valid Selections

- This provides upper bound.
  - Constraints lower the number of actual valid selections.
- Generally, do not need exact number.
  - Upper bound used for estimating worst-case scenarios.

Comparing Feature Models

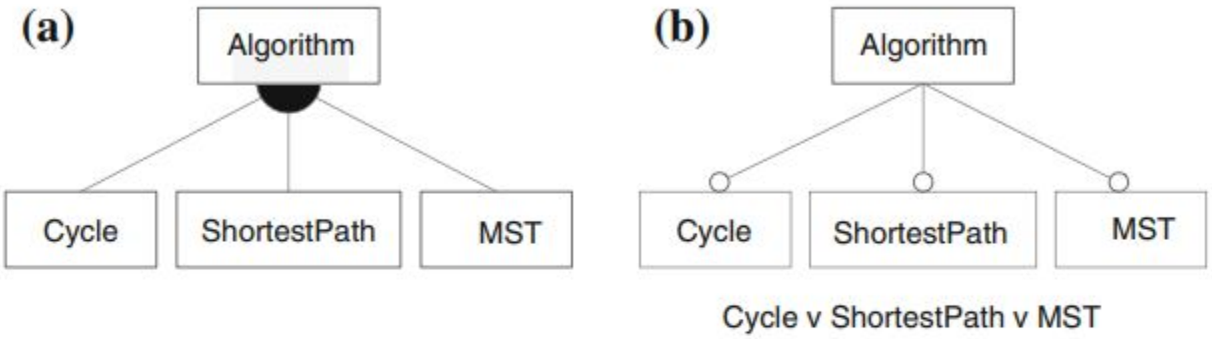


Comparing Feature Models

- Models are equivalent if formulae are equivalent.
  - $\neg(\varphi_1 \Leftrightarrow \varphi_2)$ is **not satisfiable**.
- φ_1 is a specialization of φ_2 if $(\varphi_2 \Rightarrow \varphi_1)$
  - and φ_2 is a generalization of φ_1
- SAT solver can compare two models and identify relationships.

Example - Graph Library

Use SAT Solver to prove
$\varphi_{right} \Leftrightarrow \varphi_{left}$



$$\phi_{\texttt{left}} = \text{Algorithm} \wedge ((\text{Cycle} \vee \text{ShortestPath} \vee \text{MST}) \Leftrightarrow \text{Algorithm})$$

$$\phi_{\texttt{right}} = \text{Algorithm} \wedge (\text{Cycle} \Rightarrow \text{Algorithm}) \wedge (\text{ShortestPath} \Rightarrow \text{Algorithm}) \wedge (\text{MST} \Rightarrow \text{Algorithm}) \wedge (\text{Cycle} \vee \text{ShortestPath} \vee \text{MST})$$

Where We Stand

- Feature Models can be expressed using propositional logic formulae (φ).
  - Based on model and cross-tree constaints.
- Valid feature selections result in (φ = true).
- SAT Solvers can identify valid configurations.
  - If none can be found, the model is inconsistent.
  - Enables many different model analyses.

Next Time

- Mapping of models to code
- Methods of implementing variability

- Assignment 2 is up.
  - Domain analysis, feature model creation, model analysis.
  - Due November 29

UNIVERSITY OF GOTHENBURG

CHALMERS

UNIVERSITY OF TECHNOLOGY