Proving properties of a minimal covering algorithm

Ioannis Filippidis and Richard M. Murray

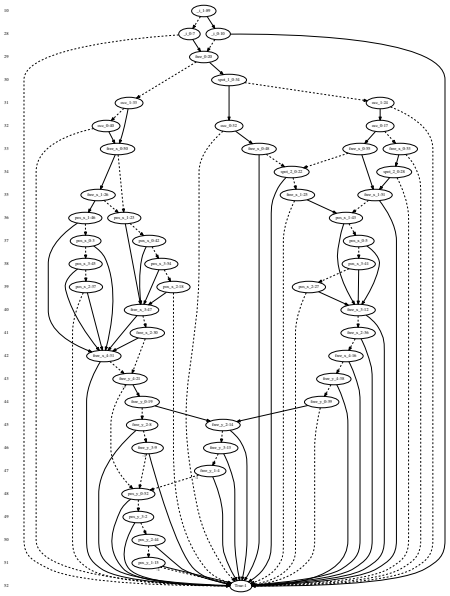
Control & Dynamical Systems
California Institute of Technology

July 18, 2018

Problem statement

- Motivation: Converting binary decision diagrams to minimal formulas in disjunctive normal form
- Formulation as a minimal covering problem
- Solution using an algorithm proposed for computing small implementations of circuits (Coudert '94)
- Proofs with TLAPS
- Python implementation

Application: Generating minimal specifications



- We use semantic methods of computation, based on BDDs.
- Specifications computed as BDDs.
- BDDs are unreadable.
- Impractical to enumerate satisfying assignments (3.9×10^6 for the above BDD)

Instead: Find a formula in disjunctive normal form with minimal size (No. of disjuncts).

$$
\begin{aligned}
&\wedge\ turn \in 1\,..\,2 \wedge free \in 0\,..\,1 \\
&\wedge\ free\_x \in 0\,..\,18 \wedge free\_y \in 0\,..\,18 \wedge occ \in 1\,.. \\
&\wedge\ pos\_x \in 1\,..\,15 \wedge pos\_y \in 1\,..\,15 \\
&\wedge\ spot\_1 \in 0\,..\,1\ \wedge spot\_2 \in 0\,..\,1 \\
&\wedge\ \ \vee\ \ \wedge (free\_x = 1) \wedge (free\_y = 1) \\
&\qquad\quad \wedge (occ \in 2\,..\,3) \\
&\qquad\quad \wedge (spot\_1 = 0) \wedge (spot\_2 = 1) \\
&\qquad \vee\ \ \wedge (free\_x = 2) \wedge (free\_y = 1) \\
&\qquad\quad \wedge (occ = 1) \\
&\qquad\quad \wedge (spot\_1 = 1) \wedge (spot\_2 = 0) \\
&\qquad \vee\ \ \wedge (free\_x \in 1\,..\,2) \wedge (free\_y = 1) \\
&\qquad\quad \wedge (occ = 3) \\
&\qquad\quad \wedge (spot\_1 = 0) \wedge (spot\_2 = 0) \\
&\qquad \vee (free = 0) \\
&\qquad \vee\ \wedge (free\_x = 2) \wedge (free\_y = 1) \wedge (occ = 3) \\
&\qquad\quad \wedge (spot\_2 = 0)
\end{aligned}
$$

Minimal covering

Given:

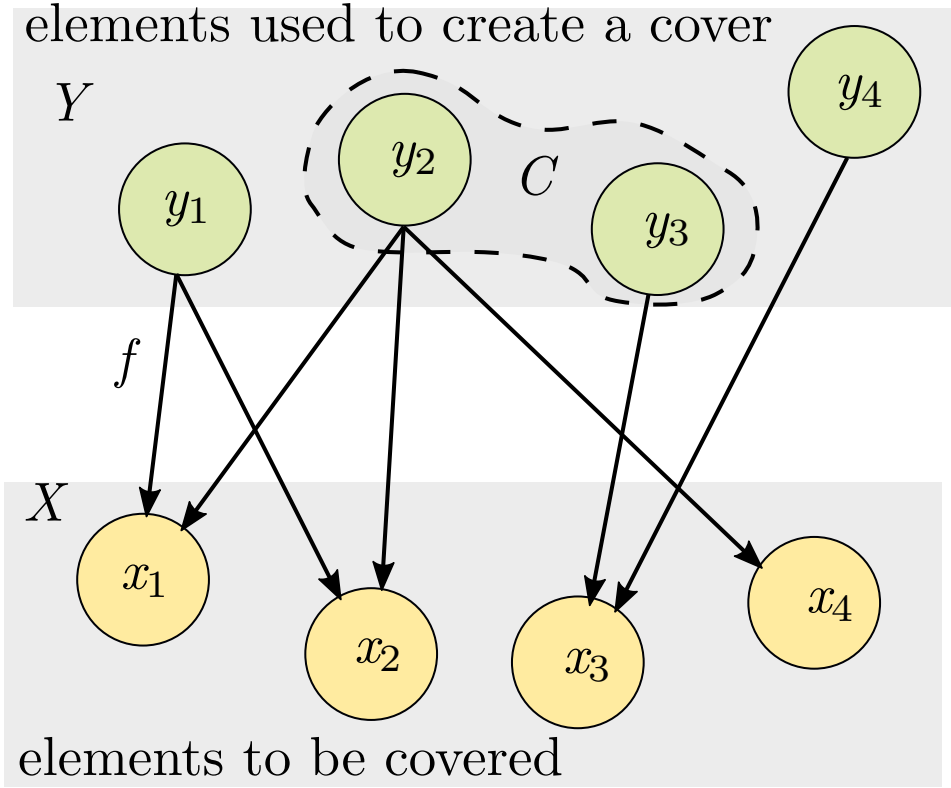
- a set X
- a set Y
- a function $f \in [X \times Y \to \text{BOOLEAN}\ ]$

Find a set $C \in \text{SUBSET}\ Y$ that:

1. "covers" the set X, as defined by f:

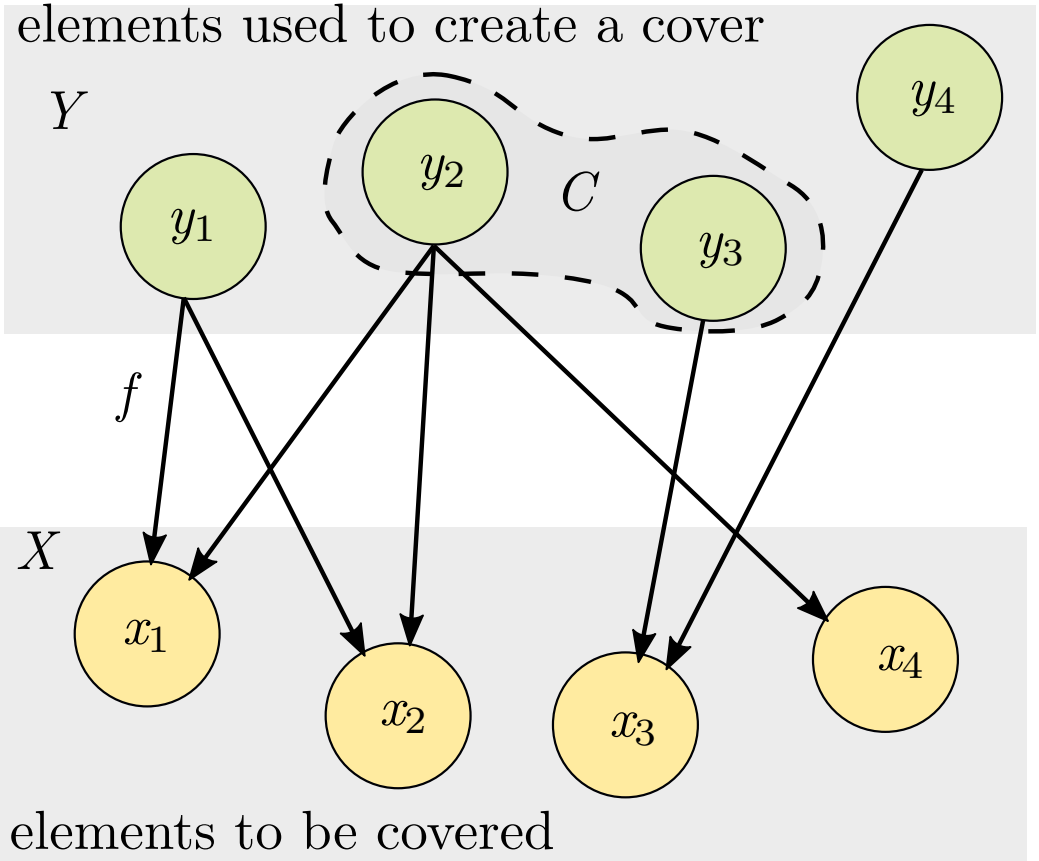
$$\forall u \in X\ :\ \exists v \in Y\ :\ f[u, v]$$

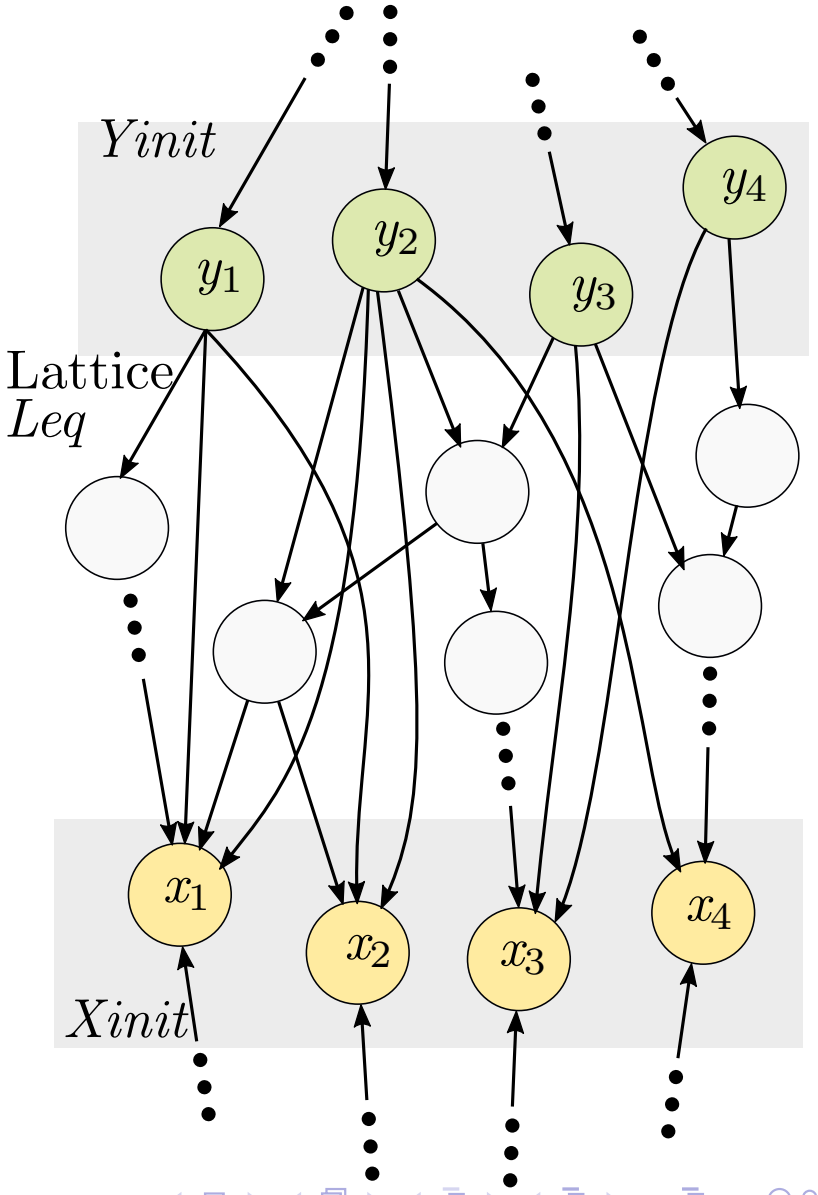
2. has minimal cardinality among covers.



Minimal covering in a lattice

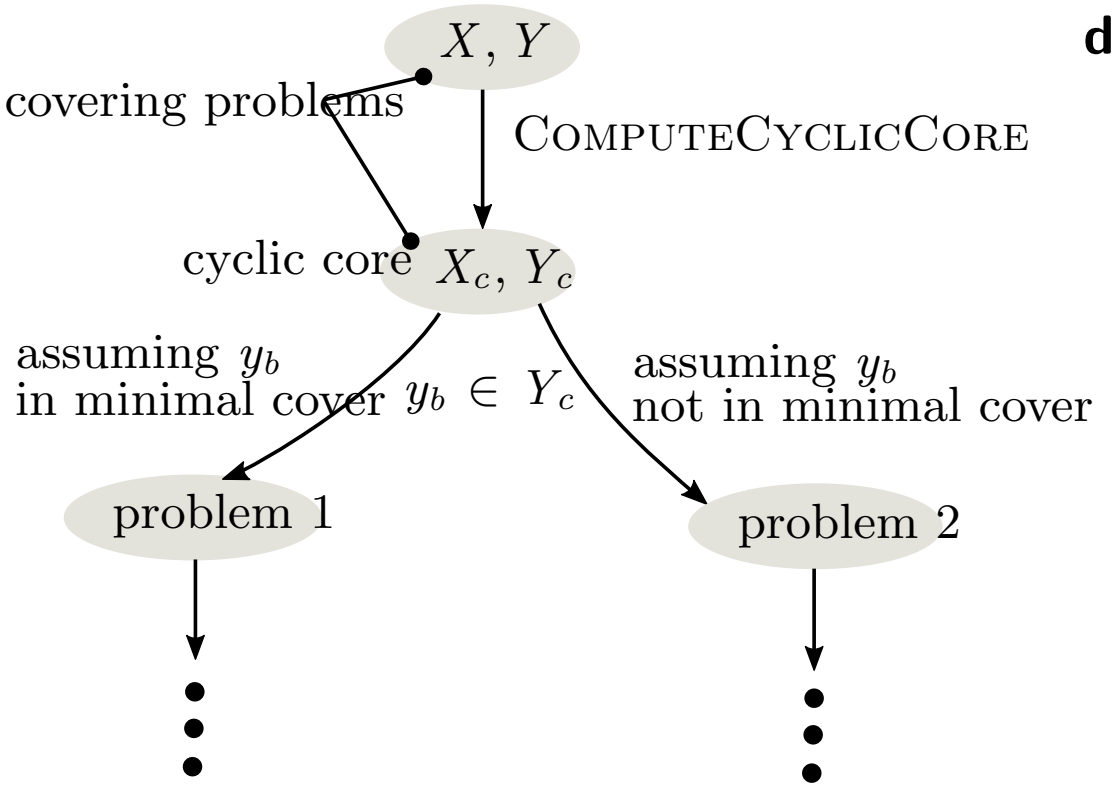
Placing the covering problem in a lattice Leq, to allow solving it by transformations.





Structure of the covering algorithm

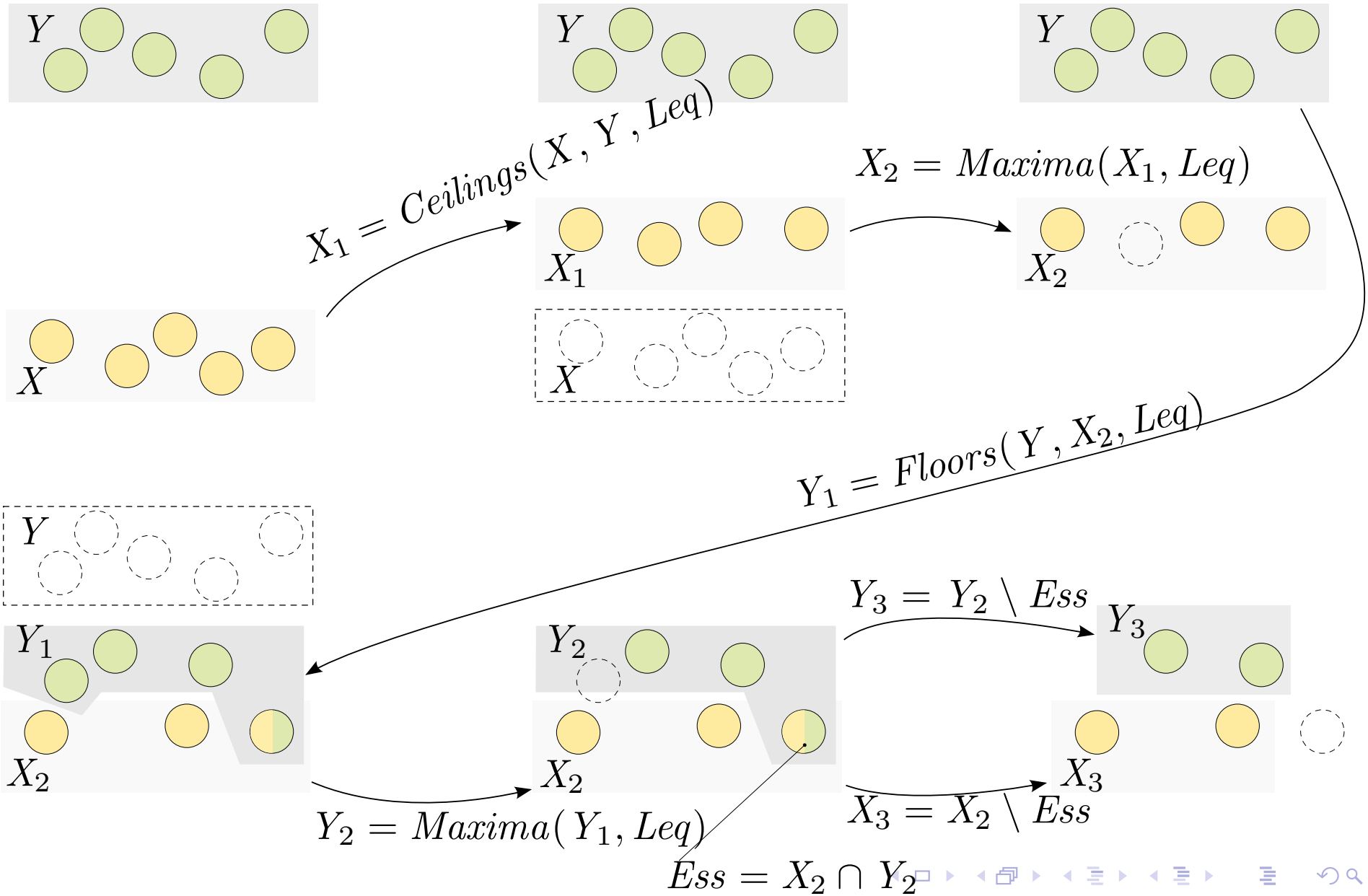
Branch-and-bound search



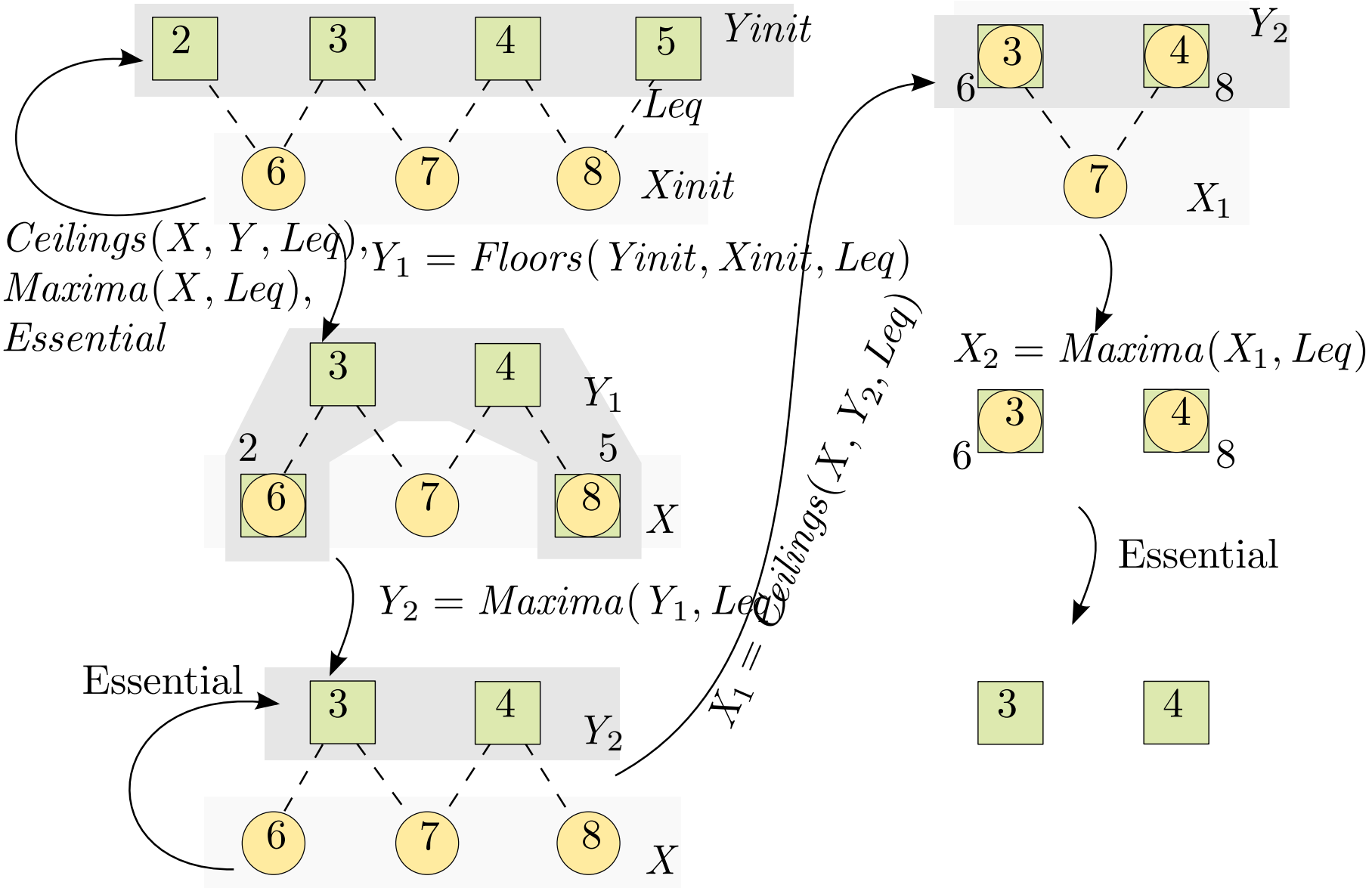
Cyclic core computation

def ComputeCyclicCore(
$Xinit, Yinit, Leq$) :
$X, Y := Xinit, Yinit$
$Xold, Yold, E := \{\}, \{\}, \{\}$
while $\langle X, Y \rangle \neq \langle Xold, Yold \rangle$:
$Xold, Yold := X, Y$
$Y := MaxFloors(Y, X, Leq)$
$X := MaxCeilings(X, Y, Leq)$
$Essential := X \cap Y$
$X := X \setminus Essential$
$Y := Y \setminus Essential$
$E := E \cup Essential$
return X, Y, E

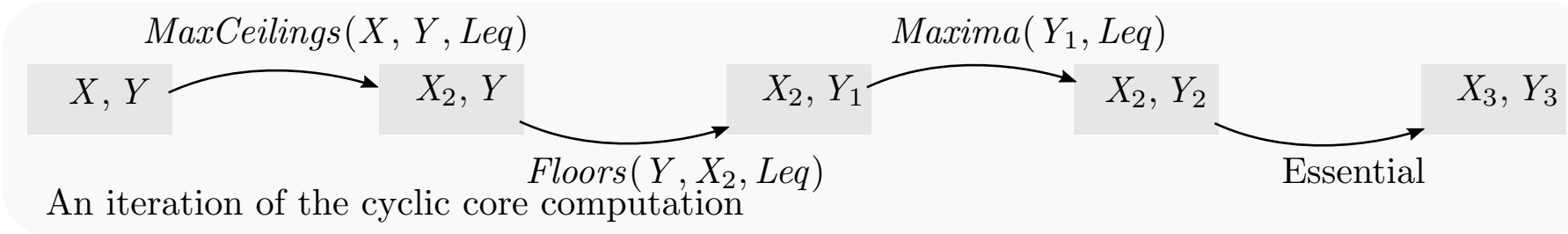
An iteration within ComputeCyclicCore

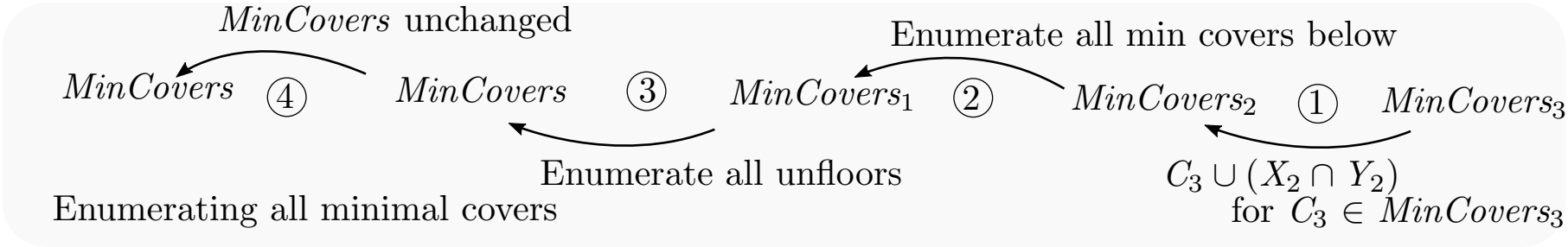


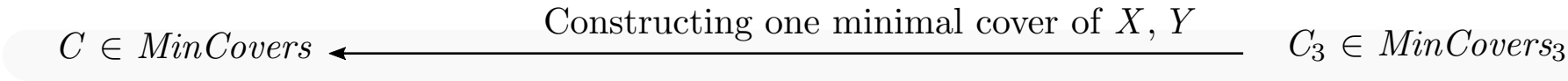
An example fixpoint



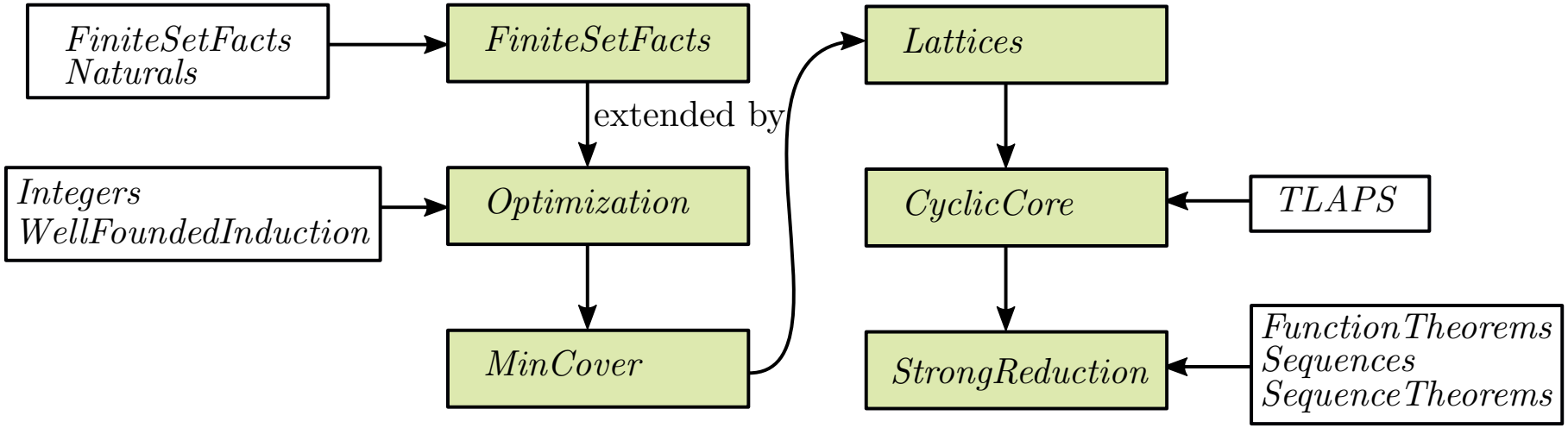
Constructing solutions of the input problem







Specification modules



Module	Content	TLAPS	Obligations
FiniteSetFacts	Addendum to *FiniteSetTheorems*	5.8 s	27
Optimization	Min/maxum/al elements, Antichains	31 s	311
MinCover	Minimal covers and their properties	30 s	237
Lattices	Floor, Ceiling, Essential elements	5 min	1334
CyclicCore	Spec and safety properties	8 min	1561
StrongReduction	Spec of all mincovers below, proofs	45 min	3038

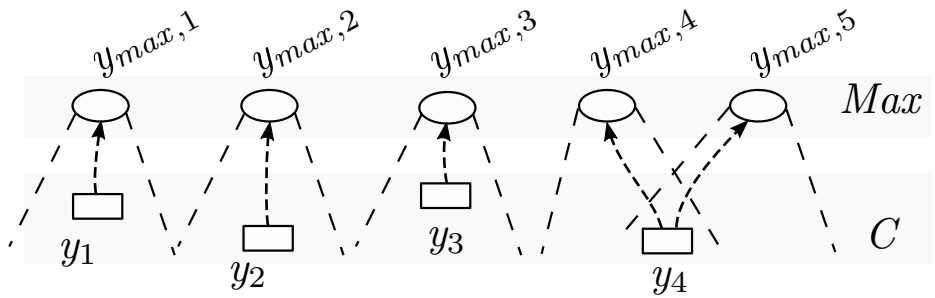
Checking time using TLAPS is with CVC3, Zenon, LS4, and Isabelle.

Proof structure

1. Cyclic core computation
   - module $Lattices$: Definitions and theorems about transformations by $Floor$, $Ceiling$, $Maxima$, and essential elements.
   - module $CyclicCore$: Specification of the procedure ComputeCyclicCore, proofs of safety properties with TLAPS, and termination proof by human.
2. Enumeration of minimal covers: module $StrongReduction$
   - Specification of the enumeration algorithm
   - Auxiliary theorems about minimal covers, refinement, bijections between minimal covers
   - Proofs of safety properties
     1. Soundness
     2. Completeness

Enumerating minimal covers

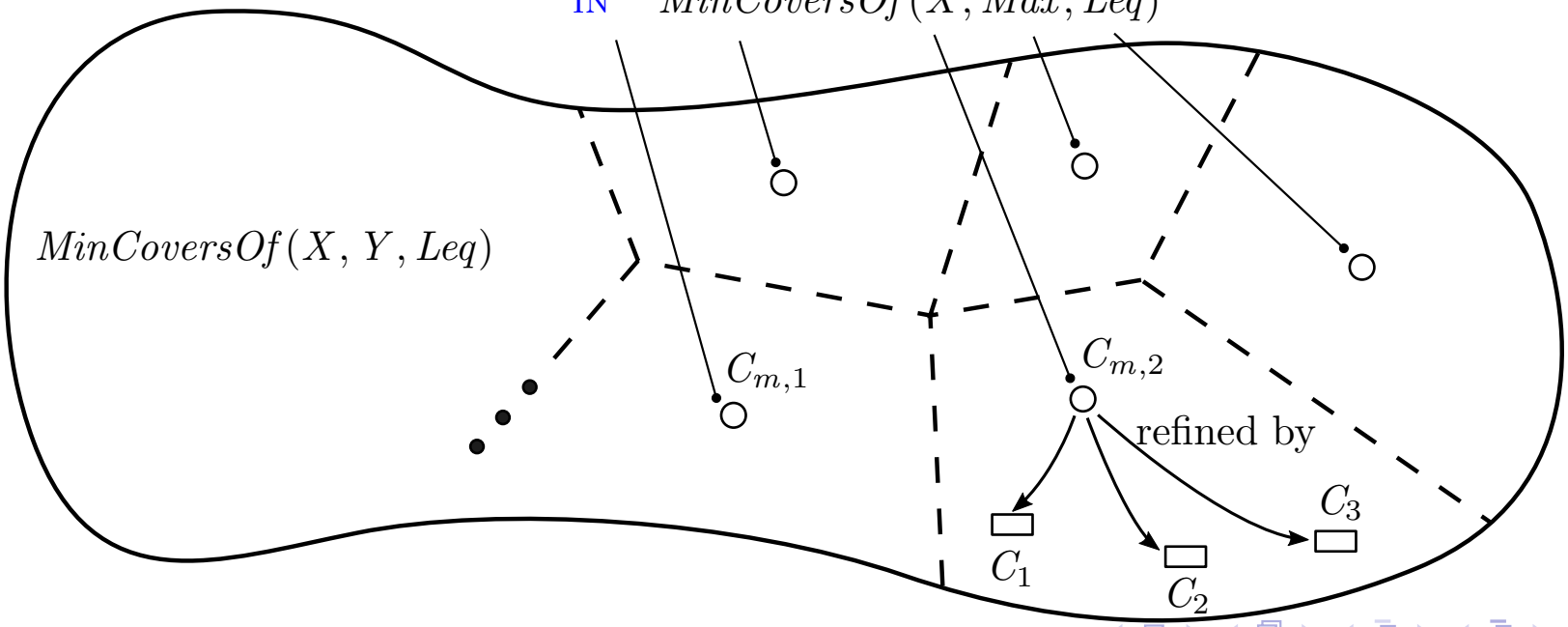
Covers of maximal elements



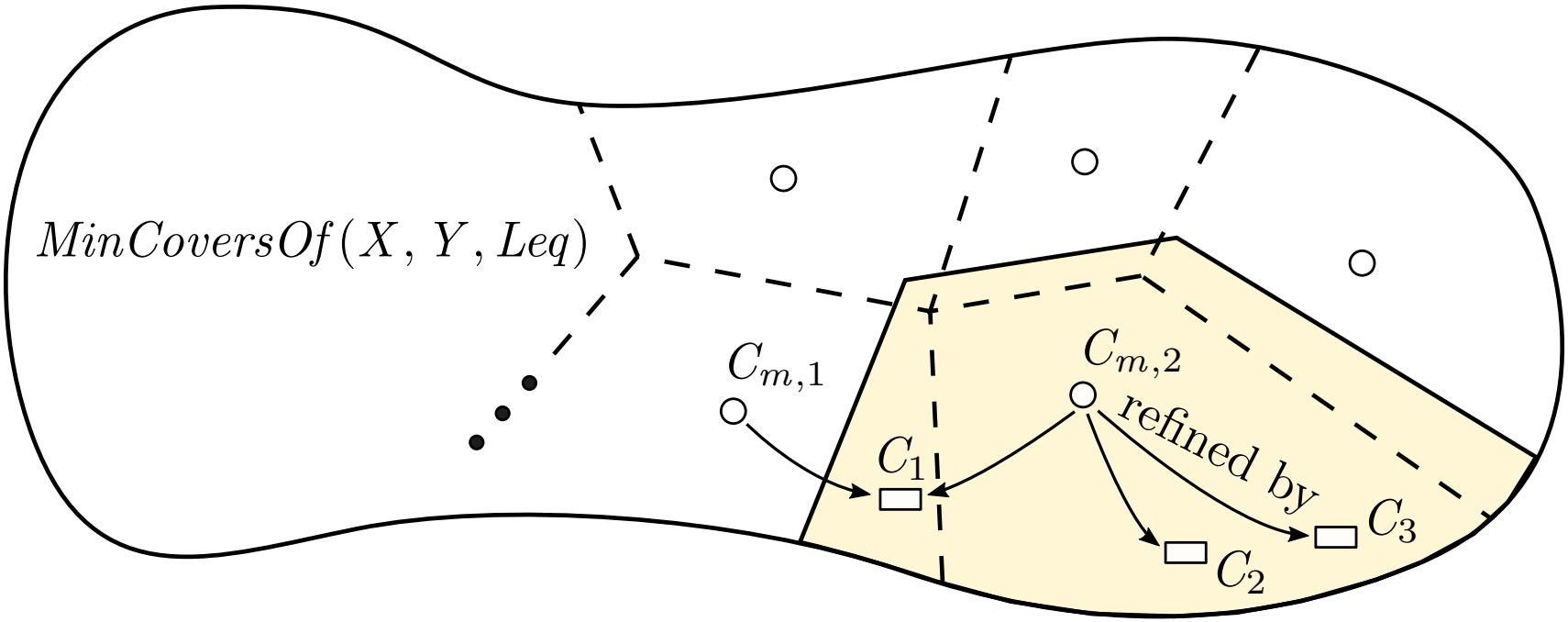
$Leq[y_1, y_{max,1}]$

Each $y_i \in C$ can be mapped to some $y_{max,j} \in Max \triangleq Maxima(Y, Leq)$, with $Leq[y_i, y_{max,j}]$, thus each cover C to some cover of maximal elements.

LET $Max \triangleq Maxima(Y, Leq)$
IN $MinCoversOf(X, Max, Leq)$

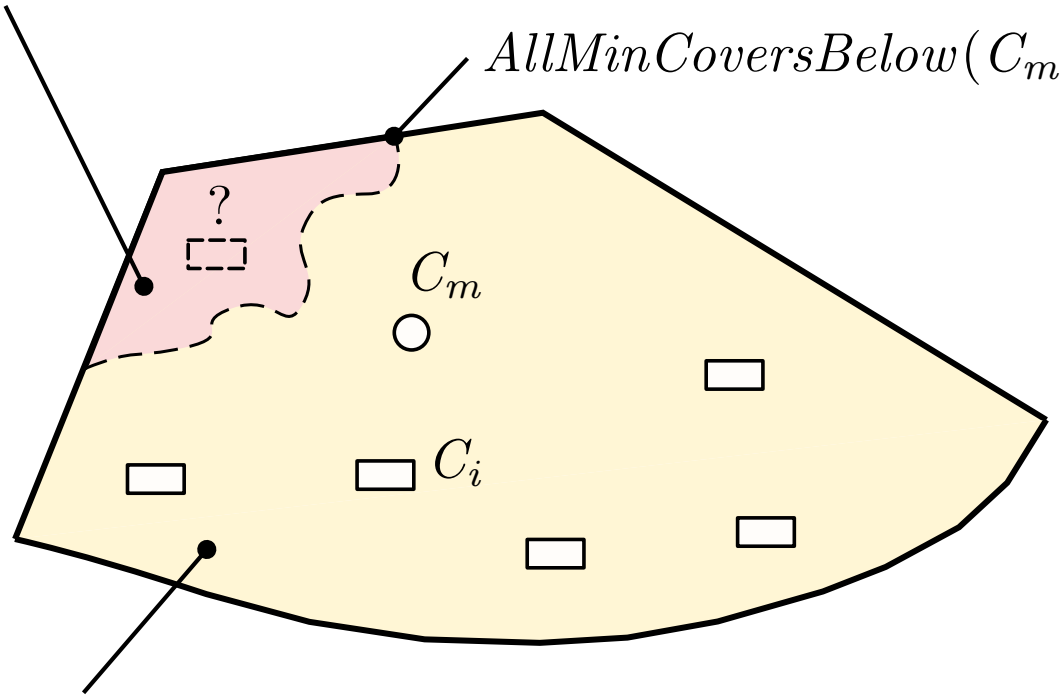


Enumerating minimal covers



Two directions of the proof

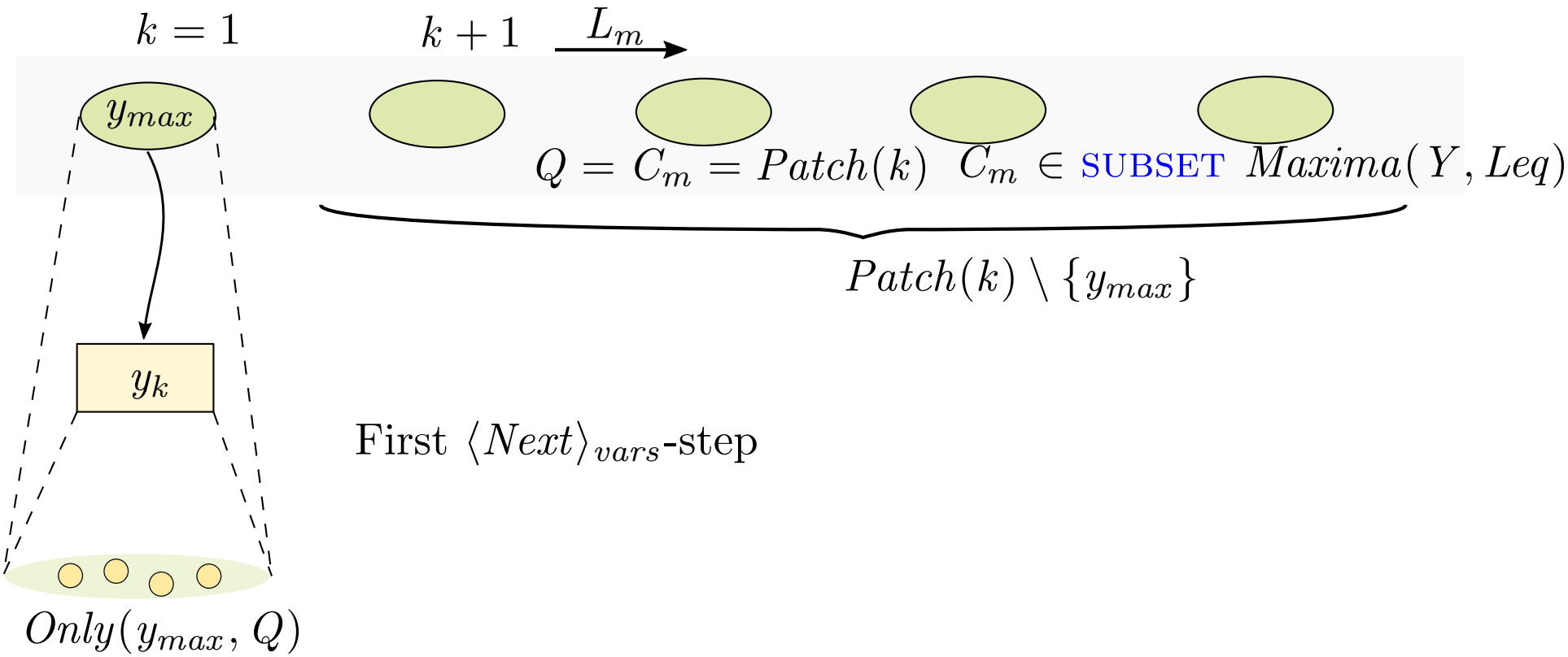
Completeness: Is enumeration missing any minimal covers that refine C_m?



Soundness: Enumerating from C_m yields only minimal covers.

Soundness proof

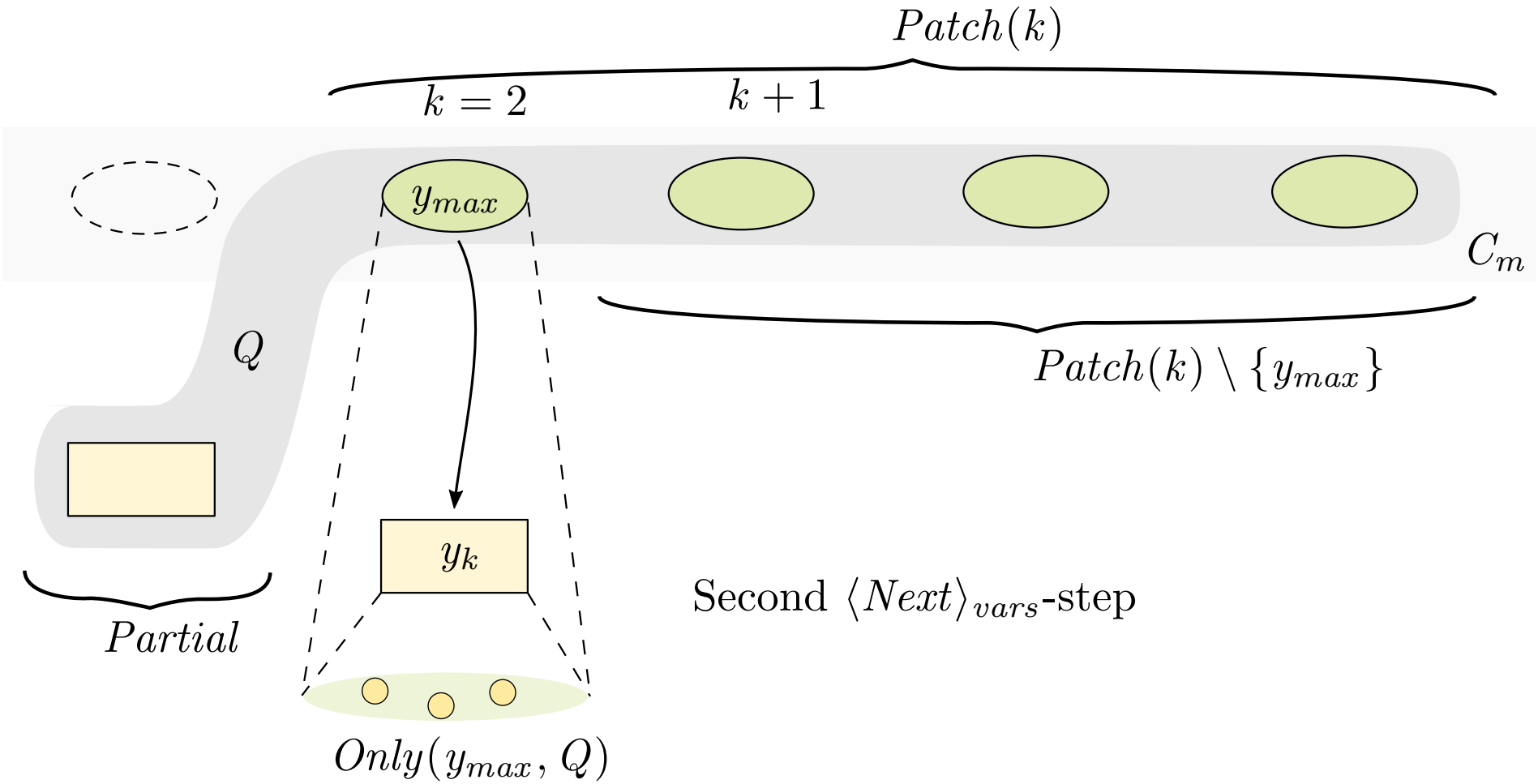
First iteration



First $\langle Next \rangle_{vars}$-step

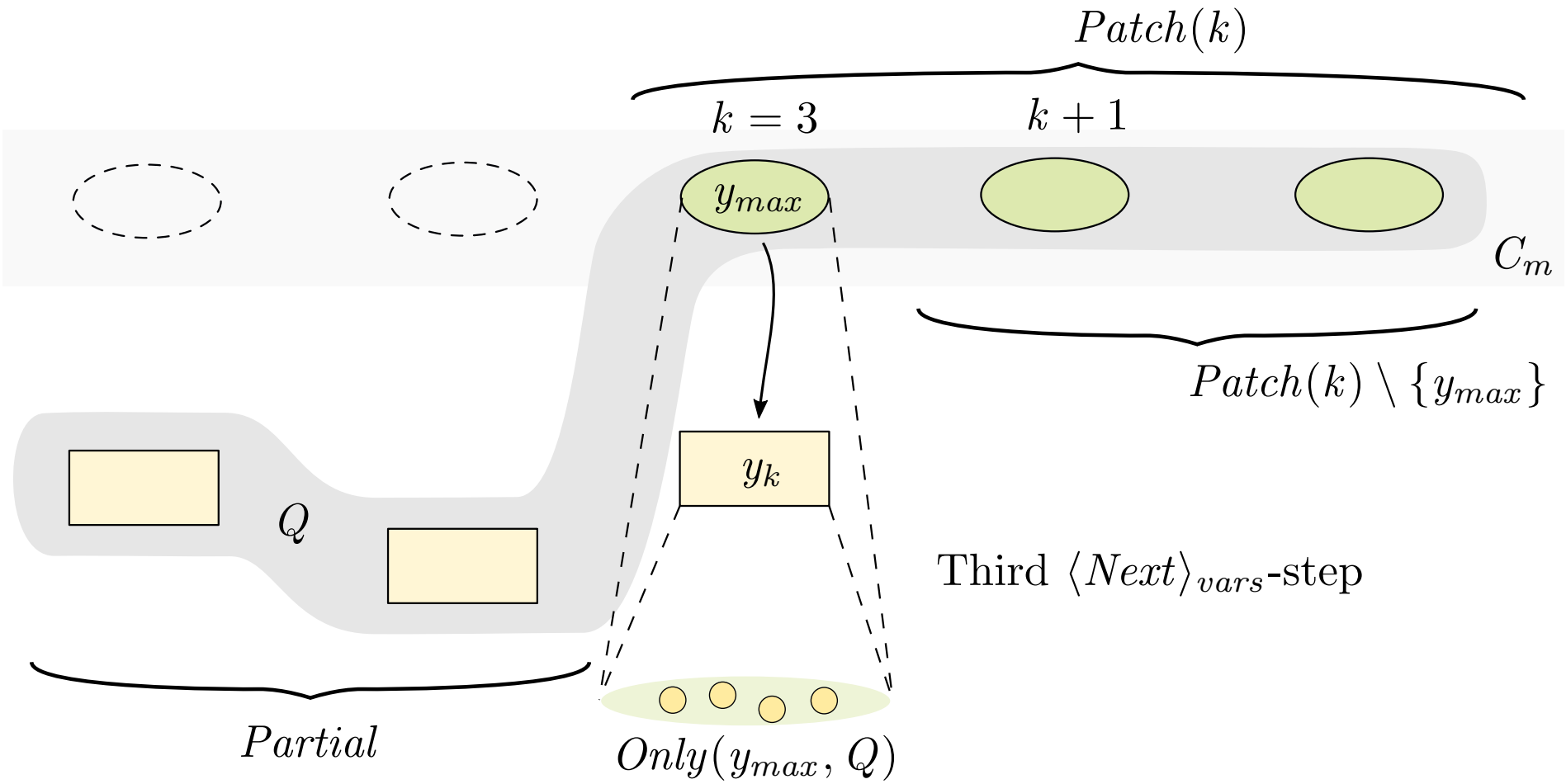
Soundness proof

Second iteration

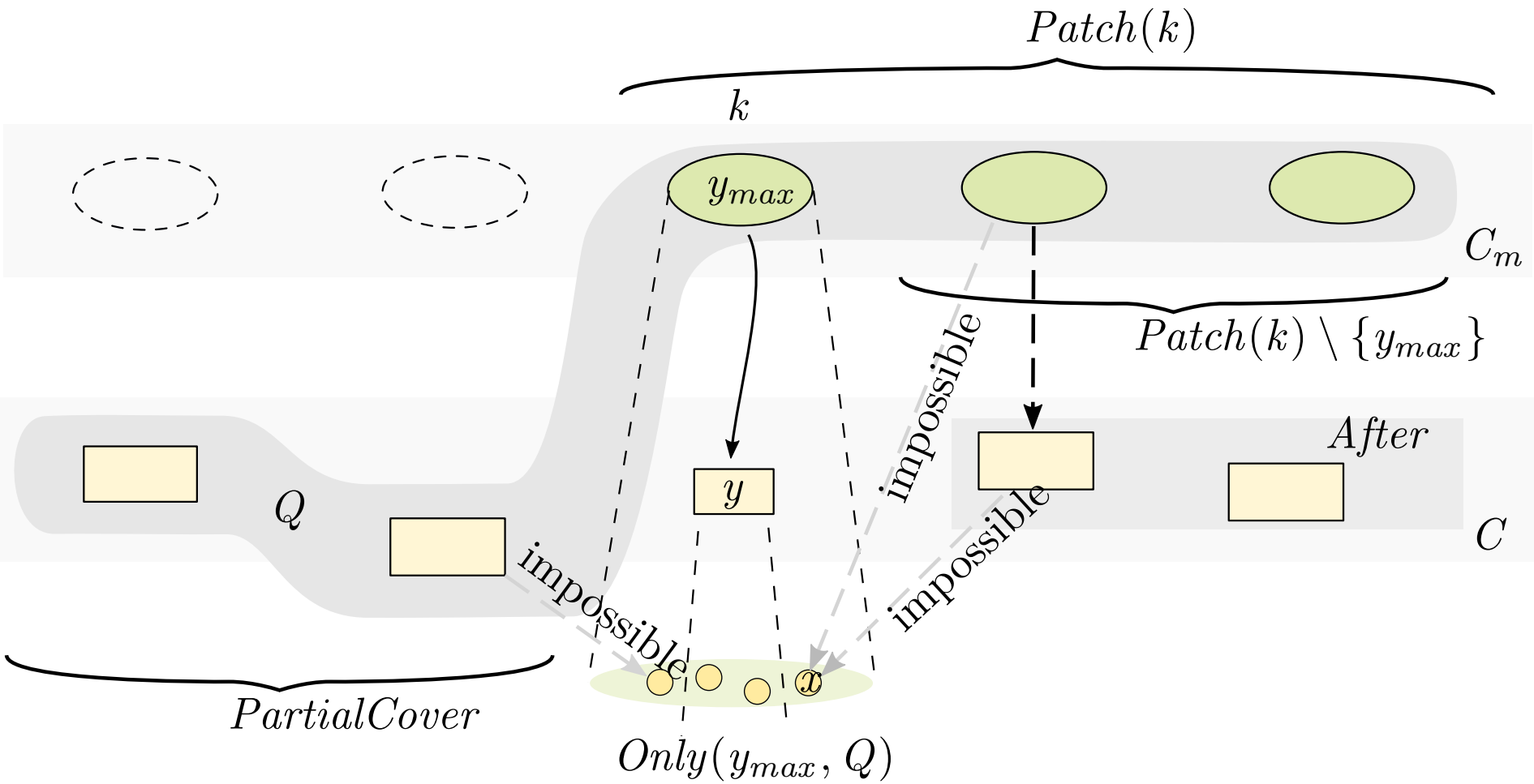


Soundness proof

Third iteration



Completeness proof



Practical observations

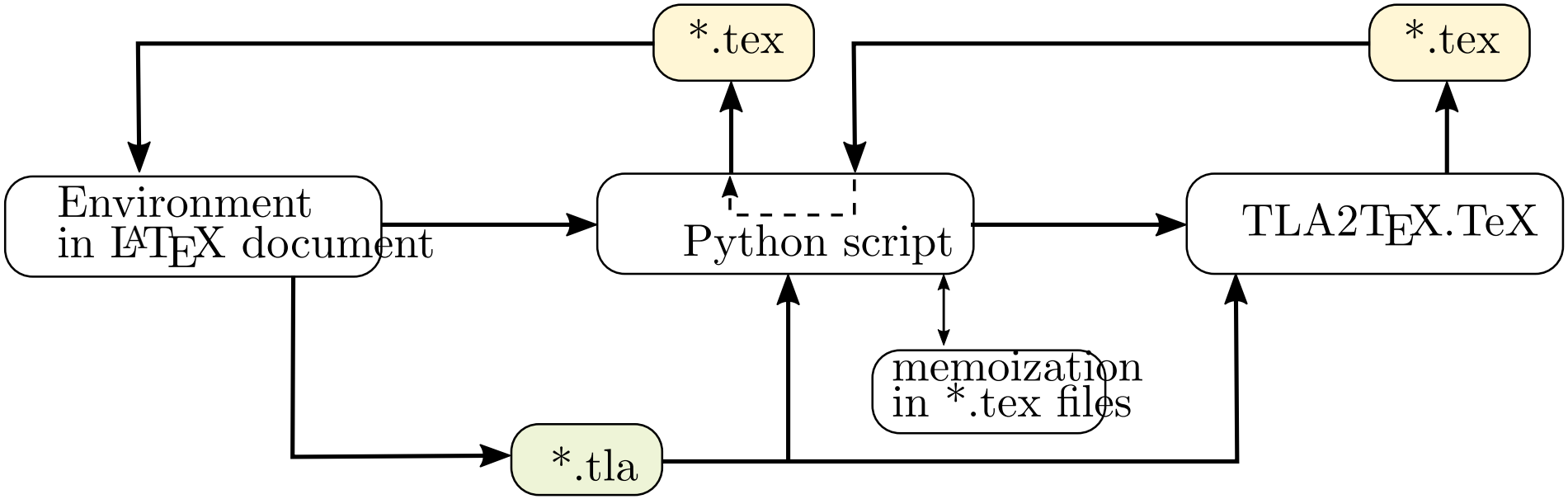
- Naming of theorems, indentation style conventions
- Introducing assumptions about constants, instead of instantiation and substitution
- Scripts for:
  - creating "header" files by removing proofs
  - typesetting a collection of TLA$^+$ modules as a single document
  - a LaTeX environment that typesets TLA$^+$ by calling TLA2TeX
  - temporary and memoized files under `__tlatex__`

Would be useful:

- Automated reporting of next unused step number
- Theorem names as directives when calling the proof manager

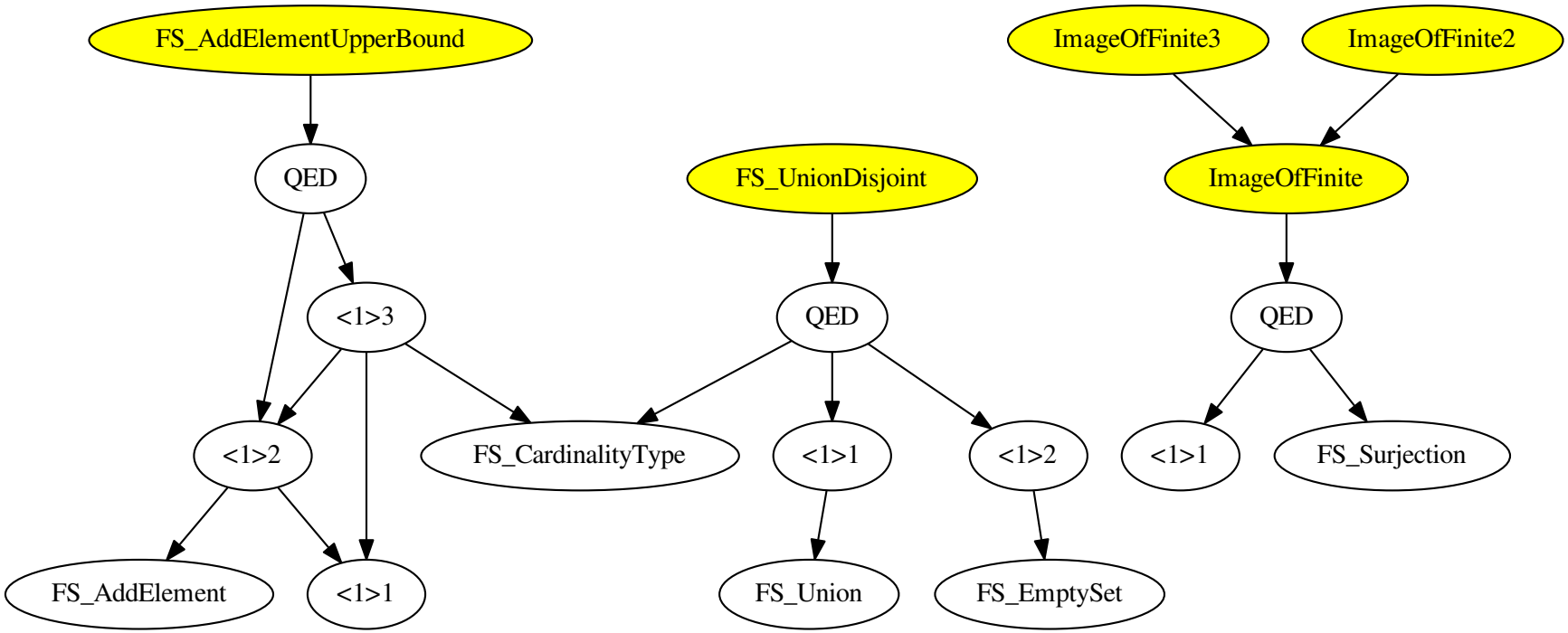
An environment for typesetting using TLA2TEX.TEX

1. A LATEX environment called `tla` for typesetting its content verbatim
2. A LATEX command called `\includetla` for typesetting external TLA$^+$ files

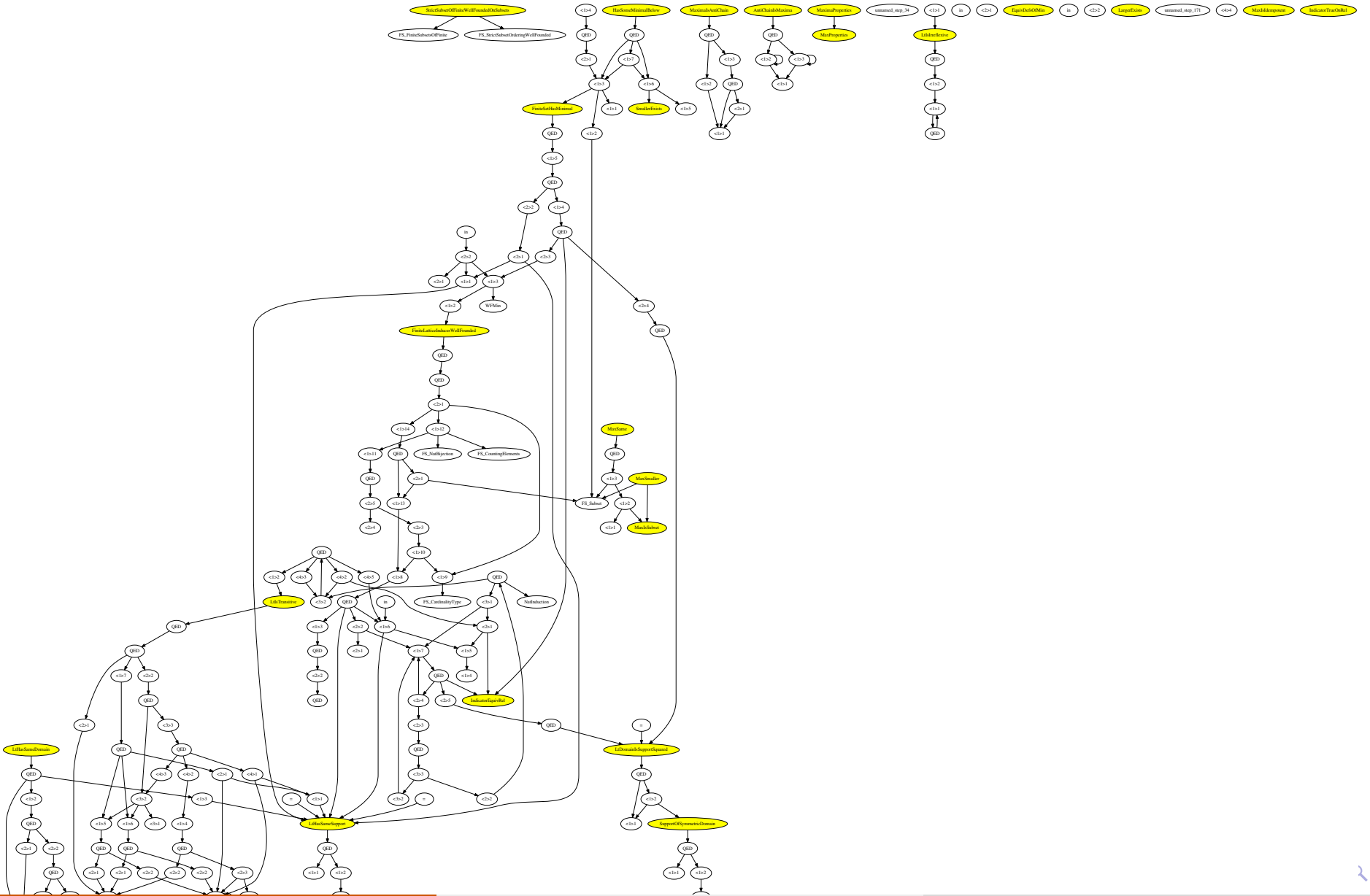


- One `*.tex` file is generated for each `tla` environment
- Unused `*.tex` files are deleted
- `\includeonly` statements are taken into account

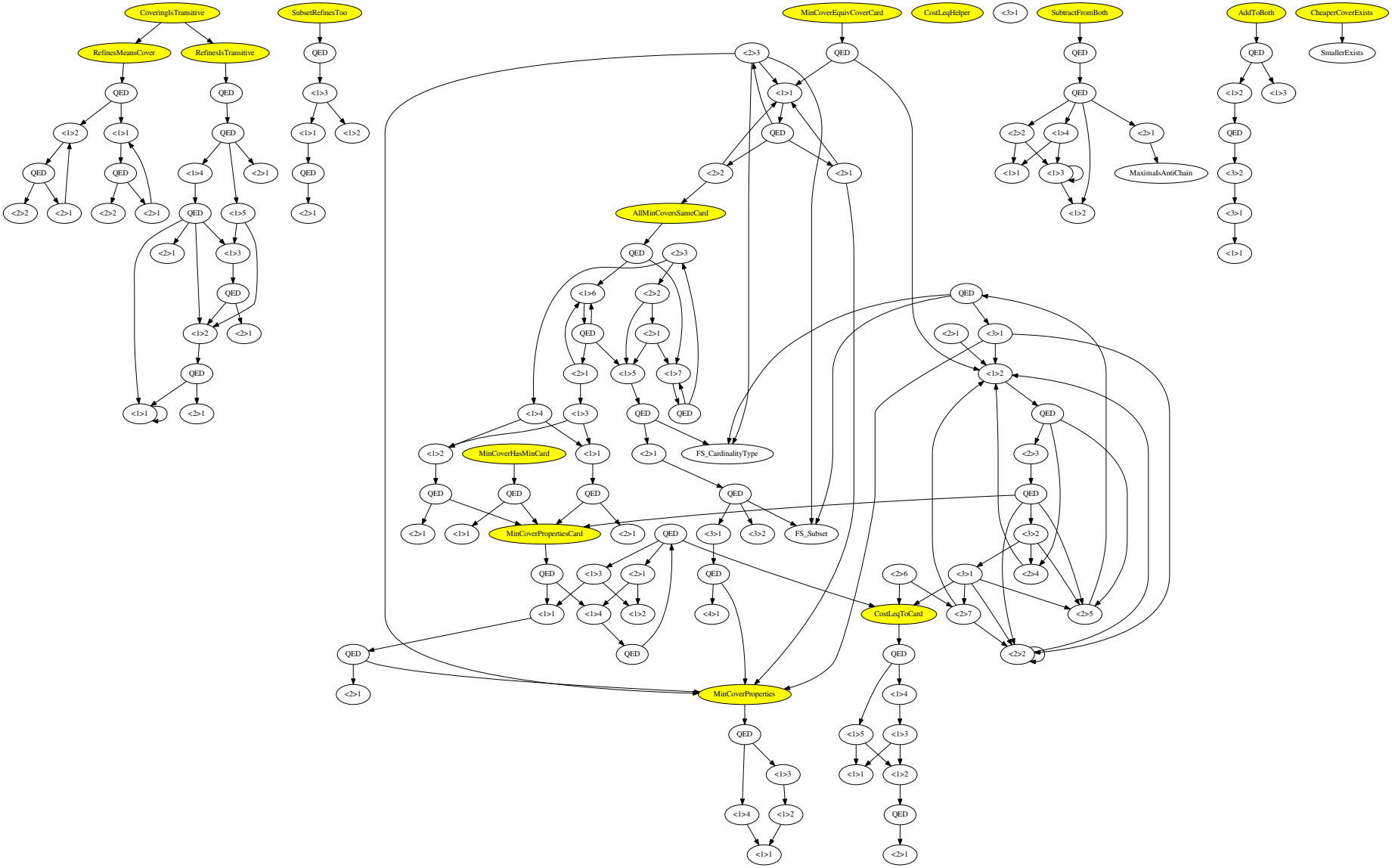
Proof graph for the module *FiniteSetFacts*



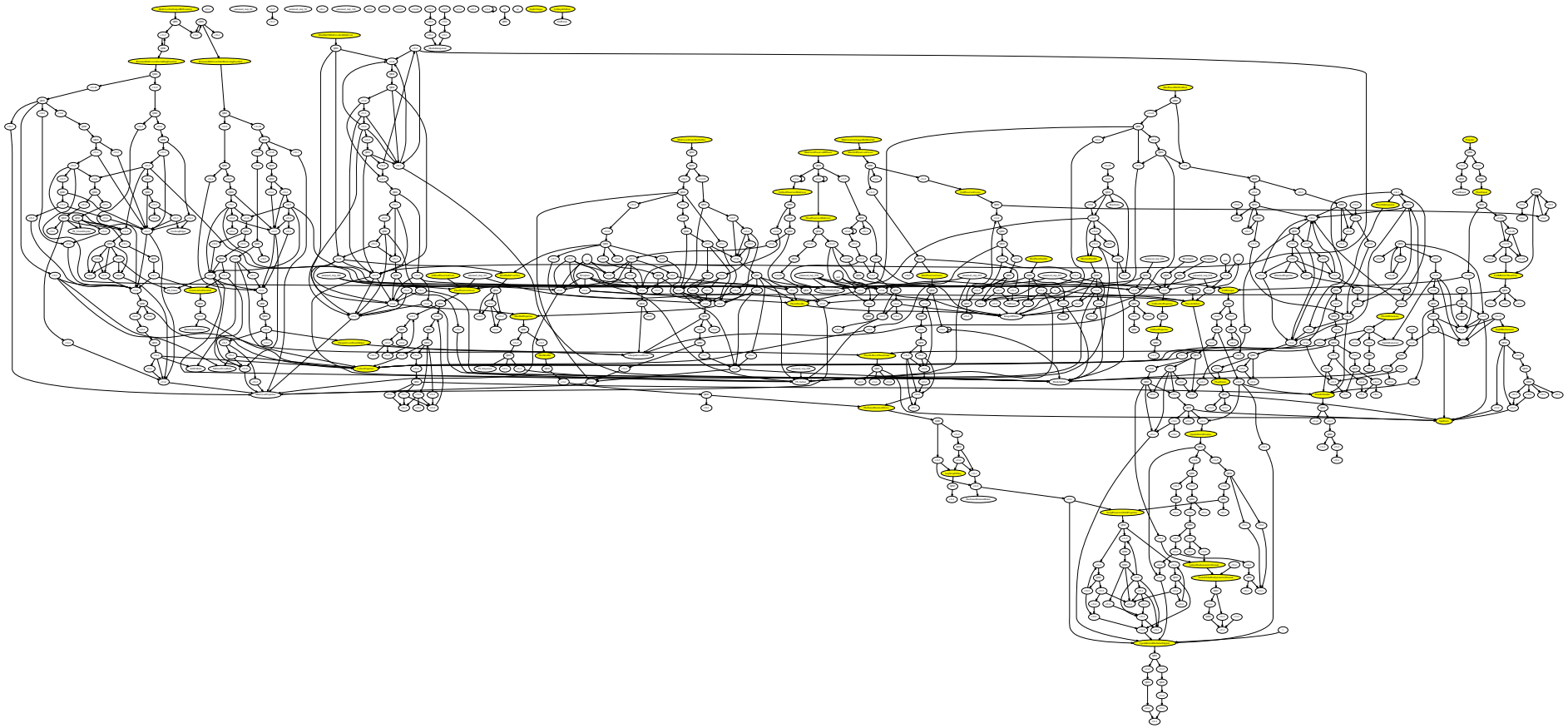
Proof graph for the module *Optimization*



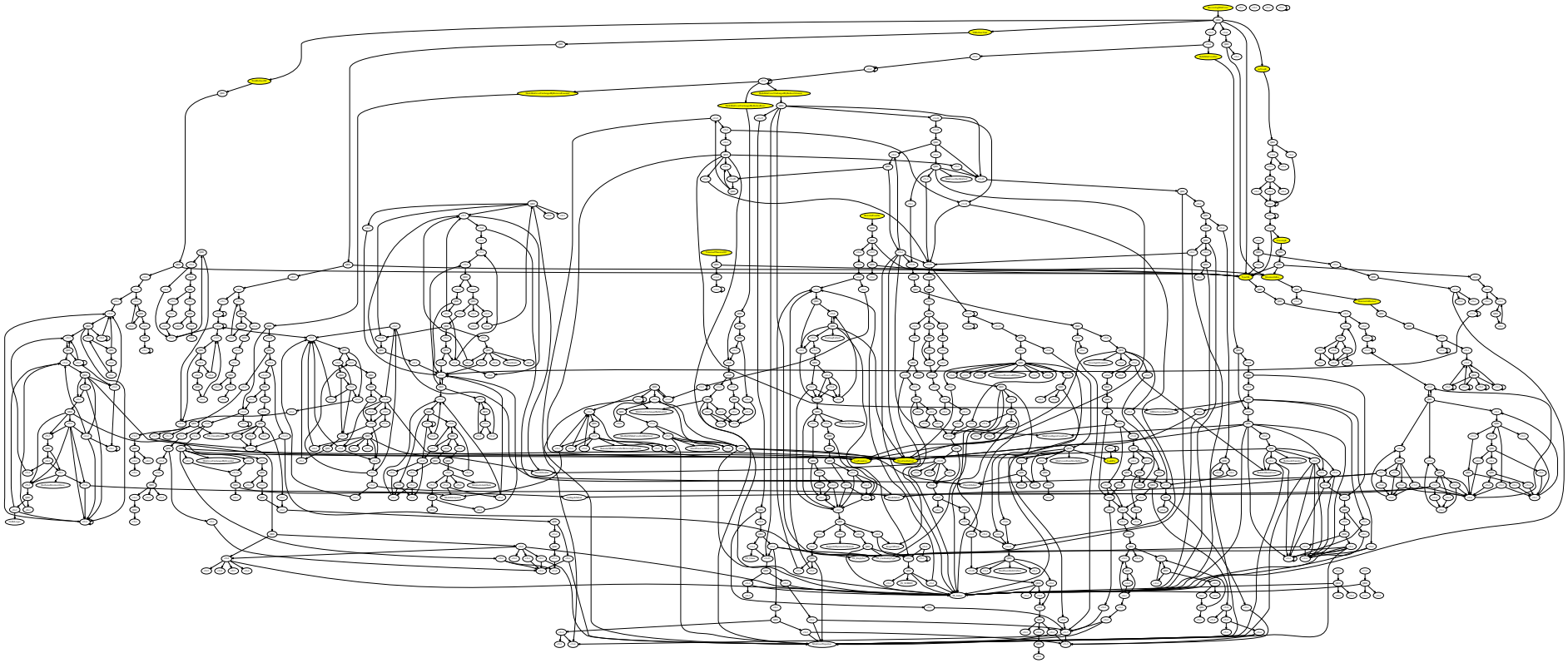
Proof graph for the module *MinCover*



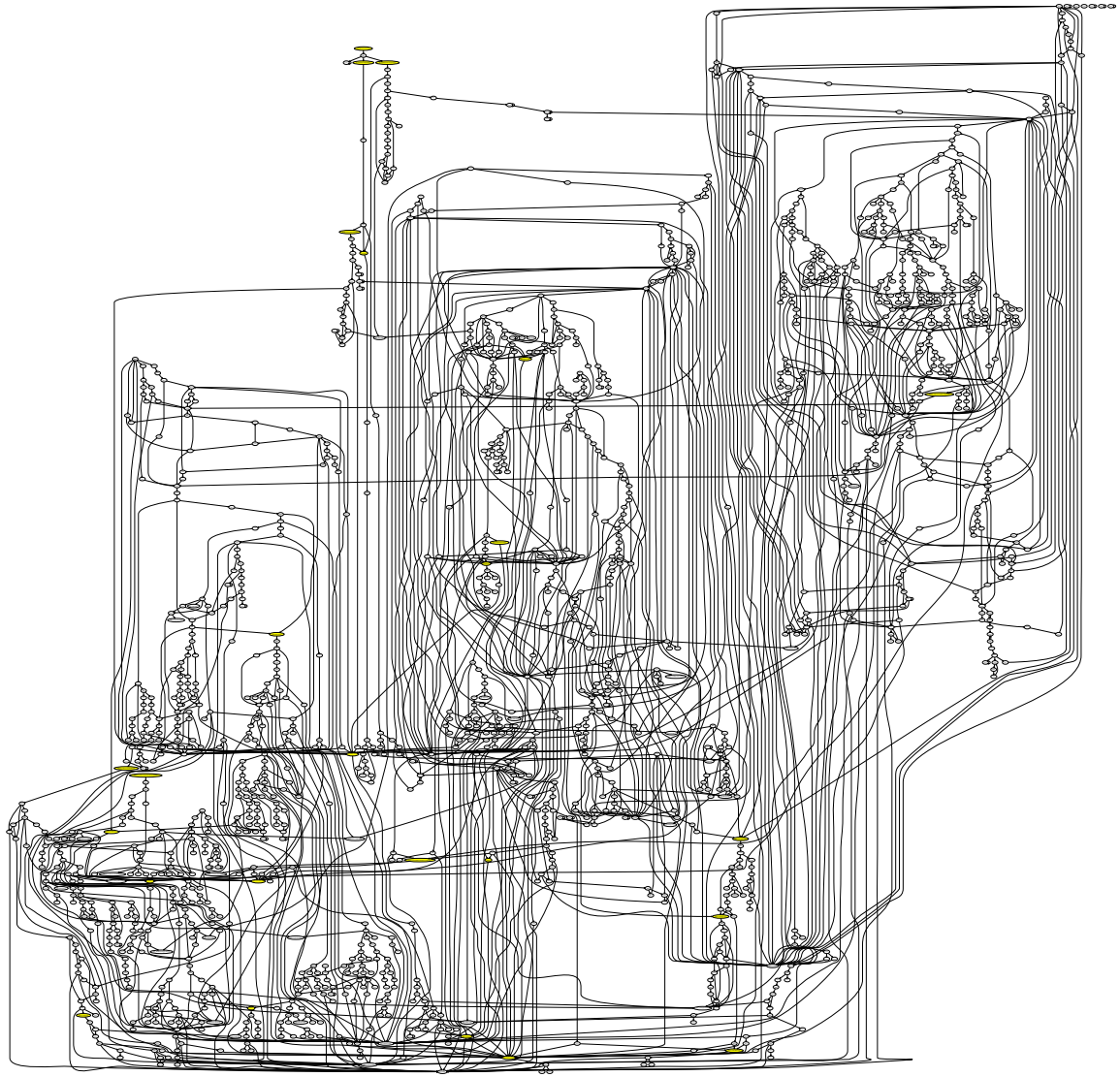
Proof graph for the module *Lattices*



Proof graph for the module *CyclicCore*



Proof graph for the module *StrongReduction*



Conclusions and future work

- Specified a minimal covering algorithm using TLA$^+$
- Proved safety properties with TLAPS
- Developed theorems relevant to optimization, minimal covers, lattices
- Developed auxiliary scripts for working with TLA$^+$ modules, and typesetting within LaTeX documents

Future directions:

- Automating the termination proof
- Specifying and proving a set-based variant of the enumeration algorithm
- Extending the proof to finite covering problems within infinite lattices