KBCV 2.0 – Automatic Completion Experiments

IWC 2013

Thomas Sternagel

Computational Logic
Institute of Computer Science
University of Innsbruck

June 28, 2013

Outline

- Basic facts
- Features
- Completion Refresher
- Optimization
- Experiments

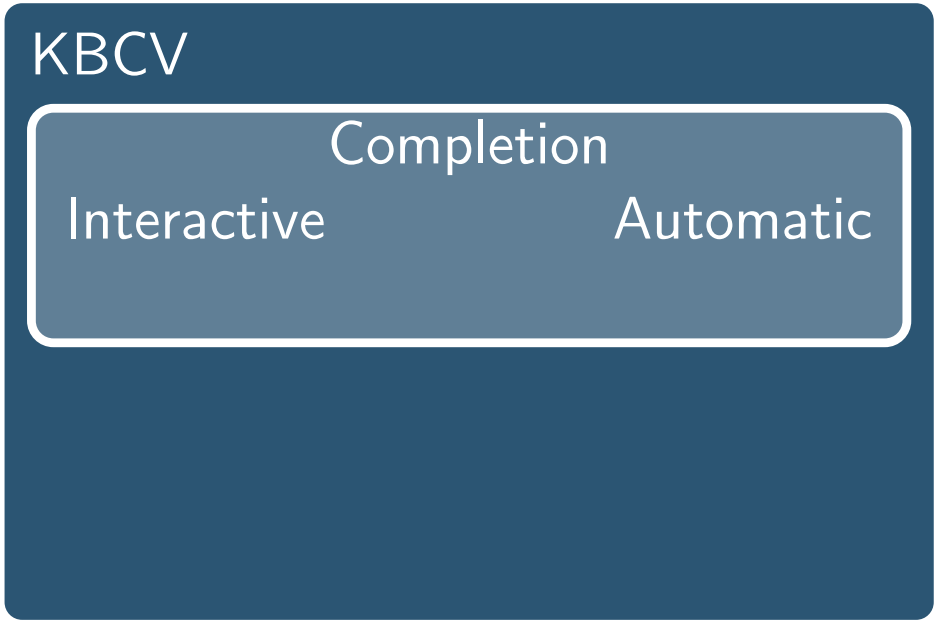
Basic facts

- `http://cl-informatik.uibk.ac.at/software/kbcv/`
- Programmed in Scala
- CLI & GUI & hybrid & web-interface
- Based on Bachmair and Dershowitz' inference system
  - External termination checker: $\mathsf{T}_\mathsf{T}\mathsf{T}_2$
  - Derivation history
- For students
  - Huet's algorithm
  - Freely apply inference rules
- Undo/Redo

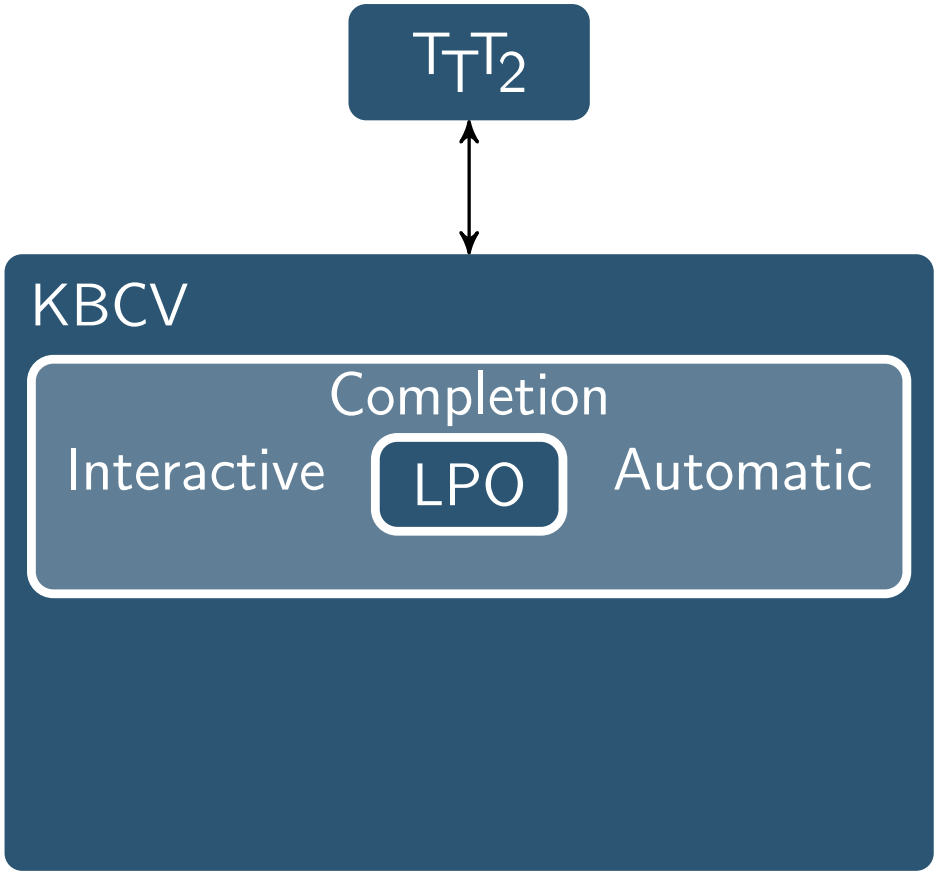
Features

KBCV

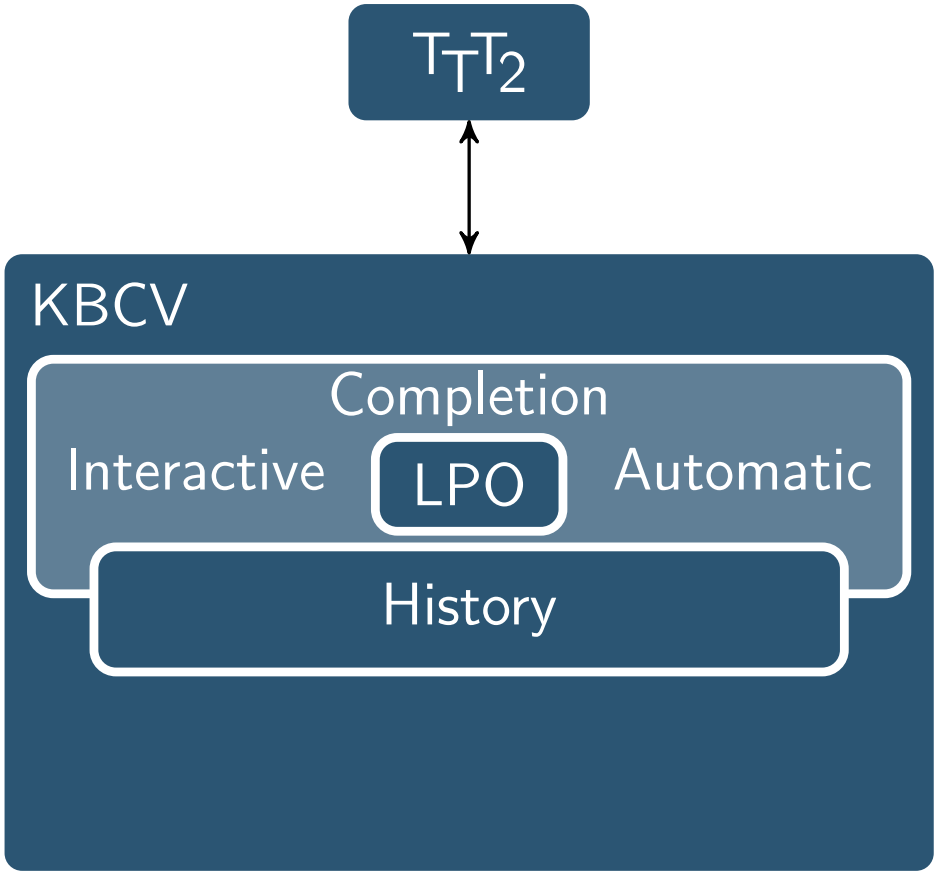
Features



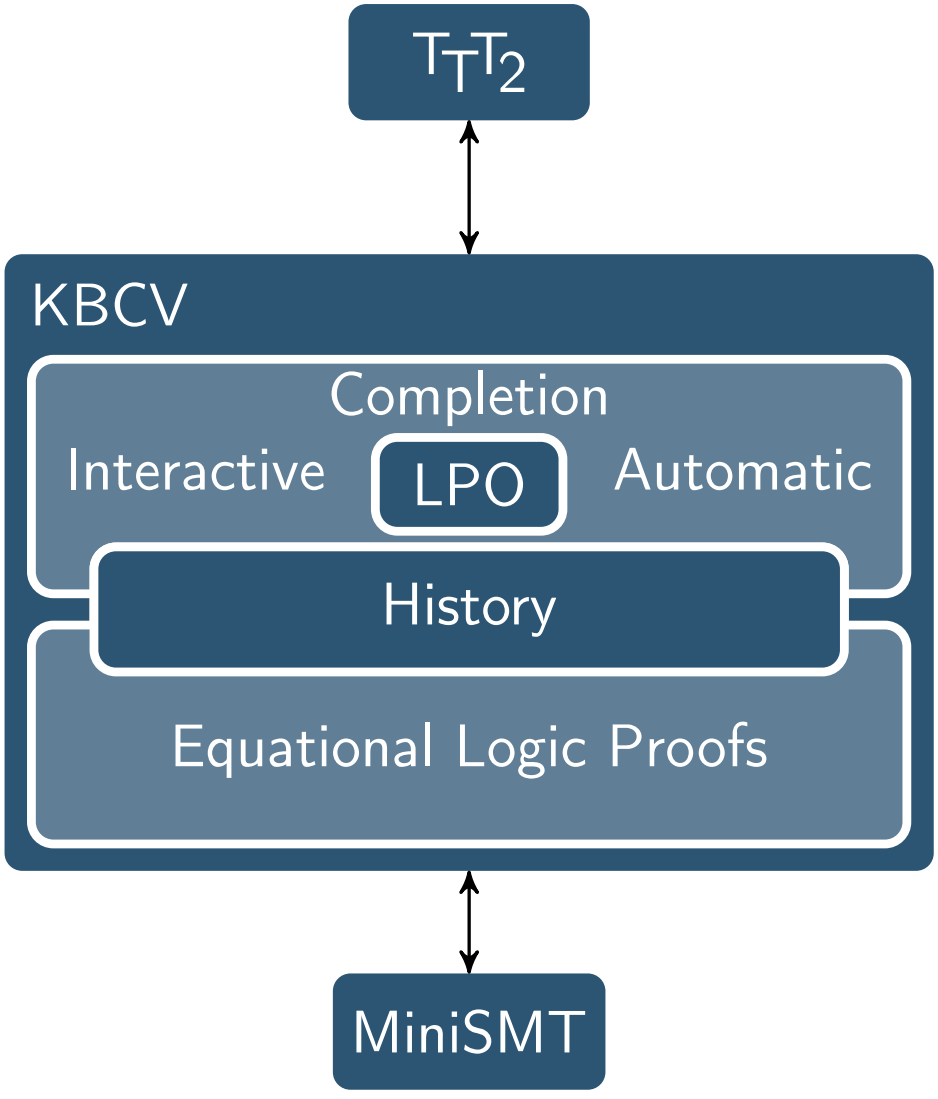
Features



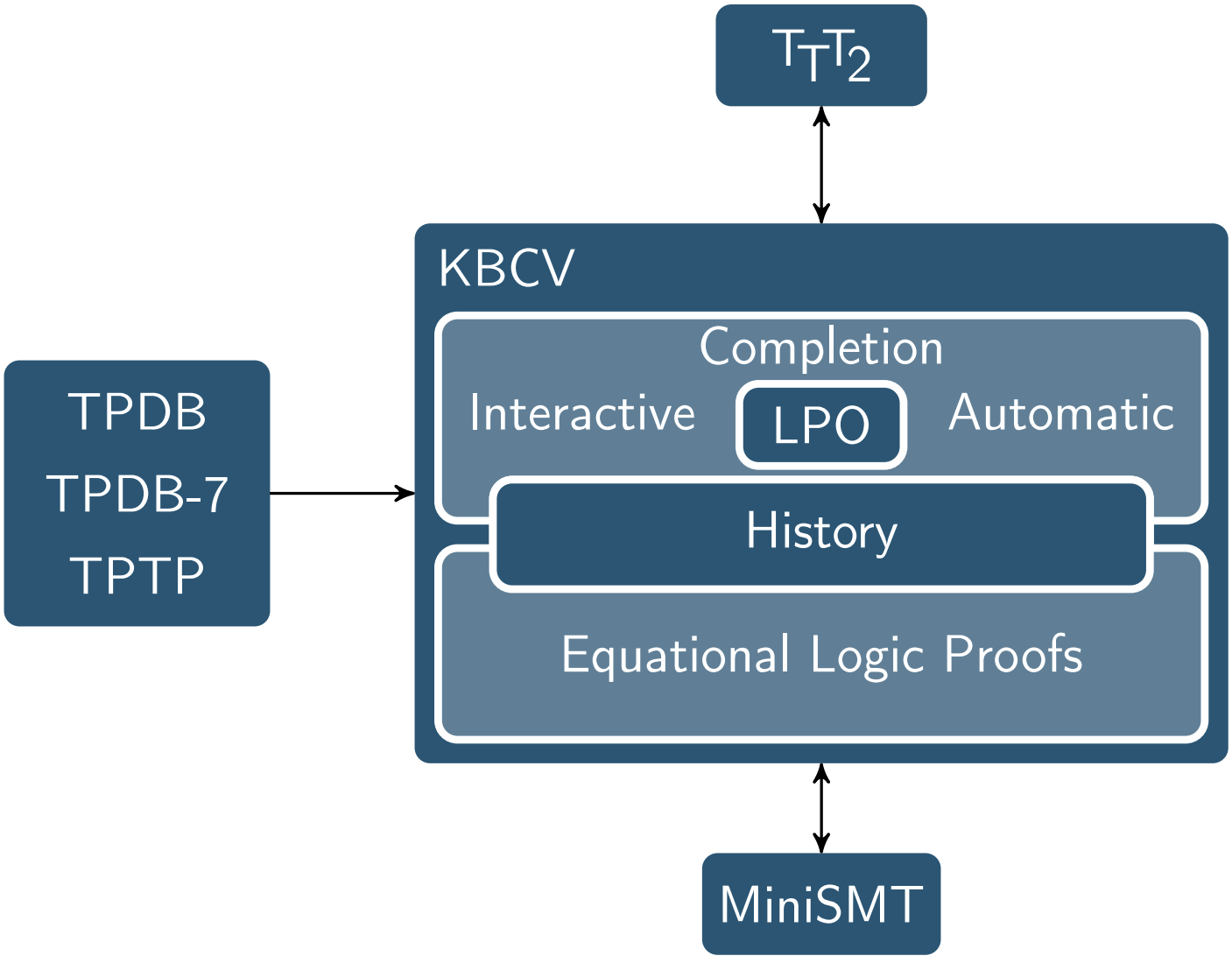
Features



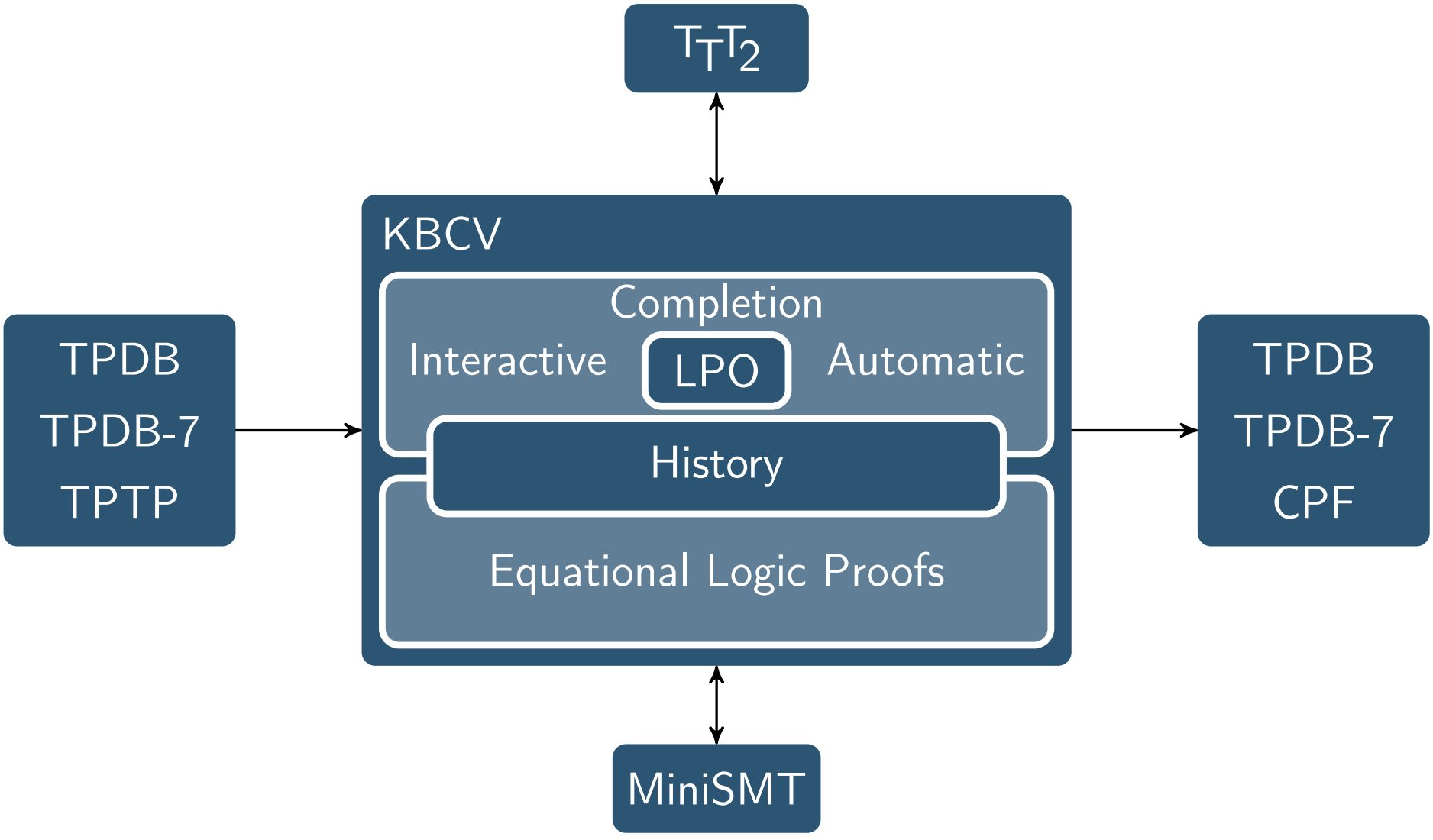
Features


TTT2
KBCV
Completion
Interactive
LPO
Automatic
History
Equational Logic Proofs
MiniSMT

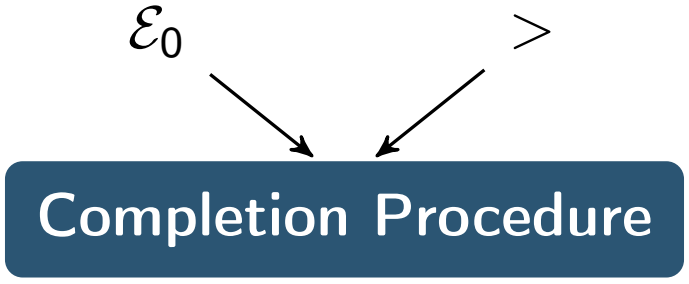
Features


TTT2
KBCV
Completion
Interactive
LPO
Automatic
History
Equational Logic Proofs
TPDB
TPDB-7
TPTP
MiniSMT

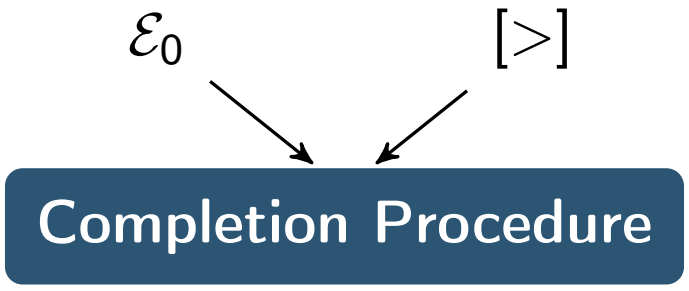
Features

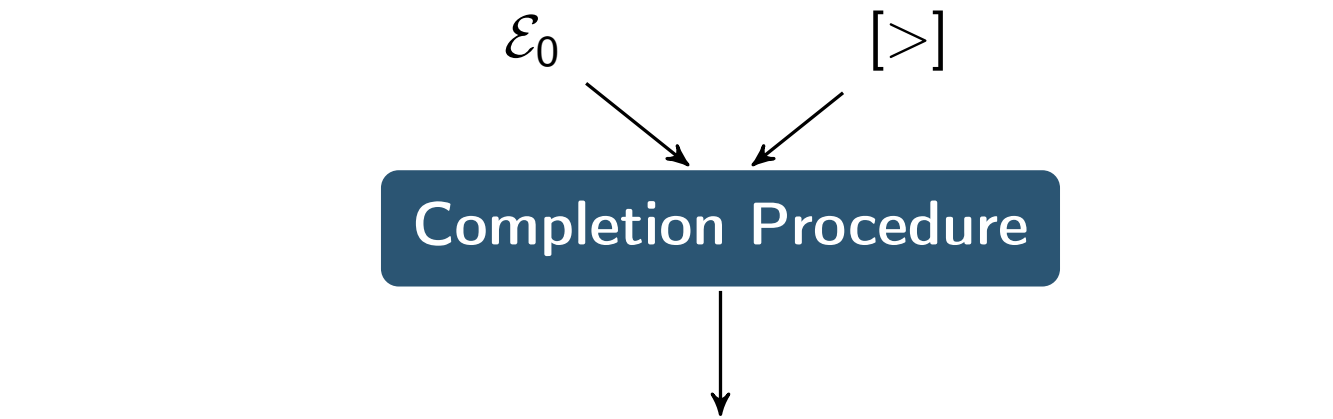

TTT2
KBCV
Completion
Interactive
LPO
Automatic
History
Equational Logic Proofs
TPDB
TPDB-7
TPTP
TPDB
TPDB-7
CPF
MiniSMT

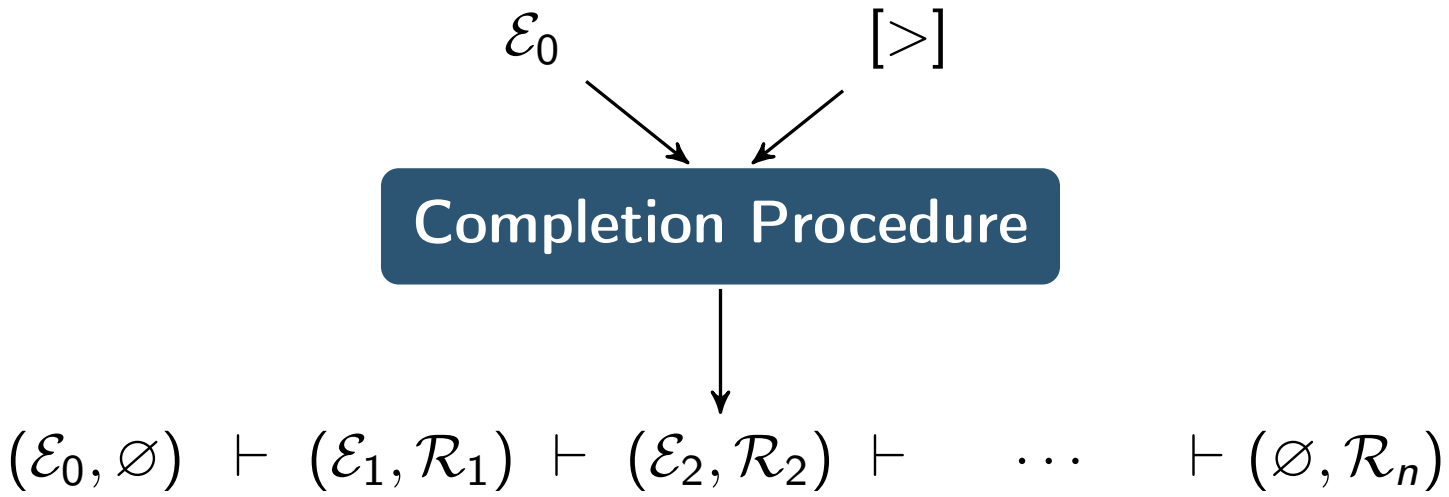
Completion Refresher



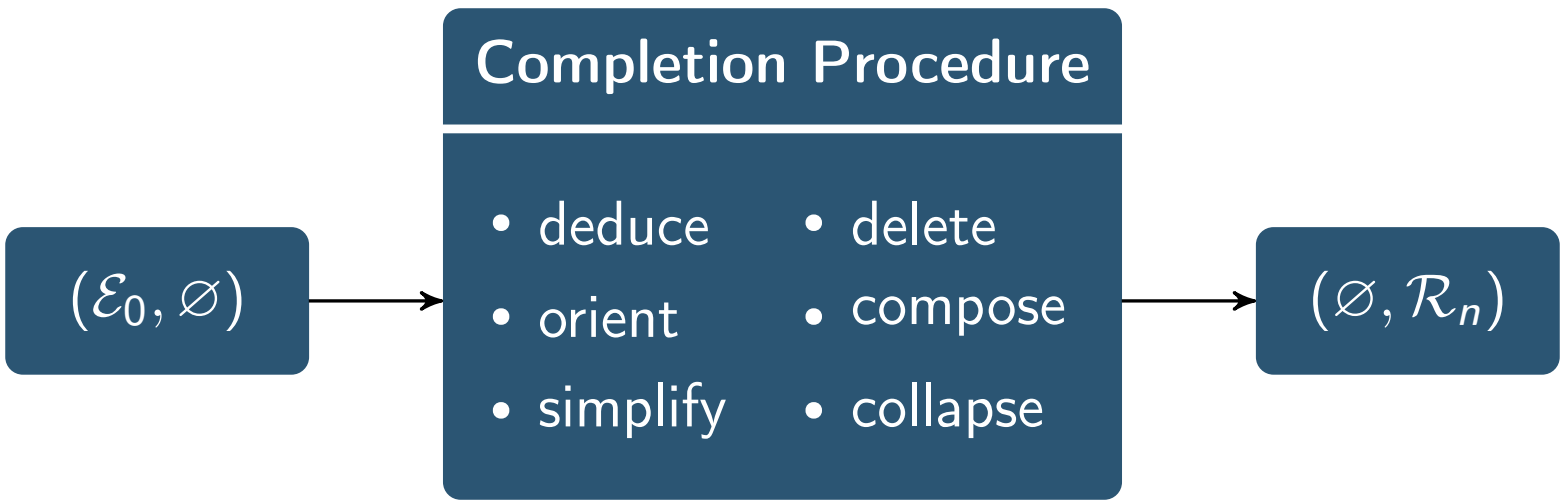
Completion Refresher



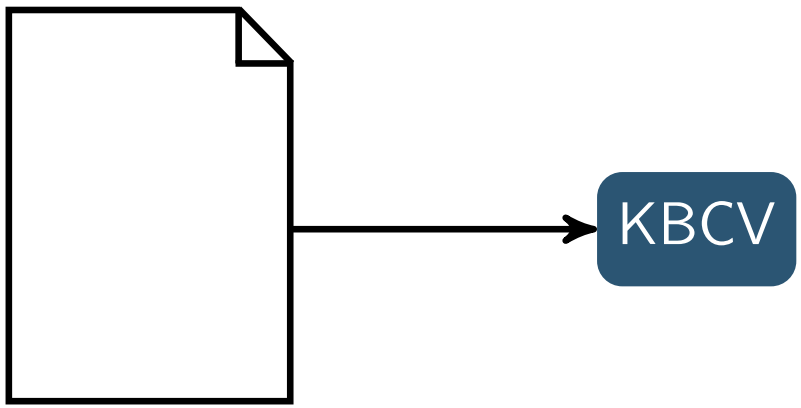

$\mathcal{E}_0$
$[>]$
Completion Procedure
$(\mathcal{E}_0, \varnothing) \vdash (\mathcal{E}_1, \mathcal{R}_1) \vdash (\mathcal{E}_2, \mathcal{R}_2) \vdash (\mathcal{E}_3, \mathcal{R}_3) \vdash \cdots$


$\mathcal{E}_0$
$[>]$
Completion Procedure
$(\mathcal{E}_0, \varnothing) \vdash (\mathcal{E}_1, \mathcal{R}_1) \vdash (\mathcal{E}_2, \mathcal{R}_2) \vdash \cdots \vdash (\varnothing, \mathcal{R}_n)$

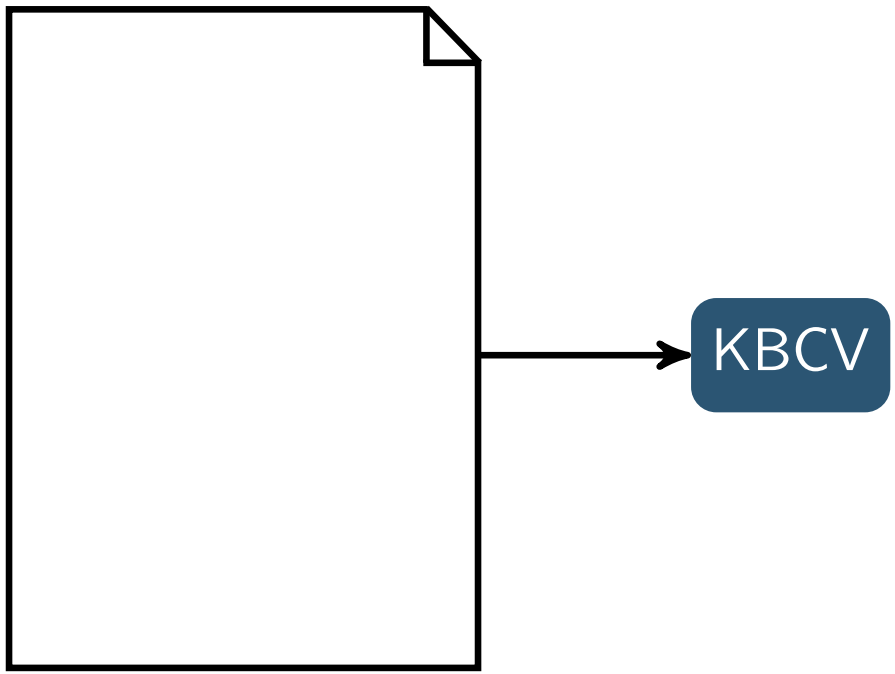
Completion Refresher



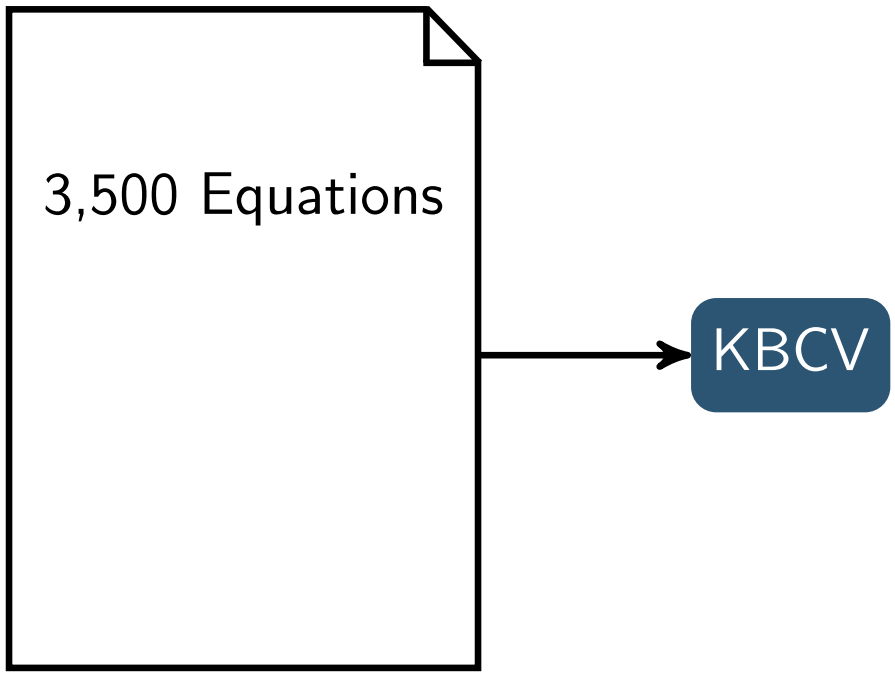
Motivation



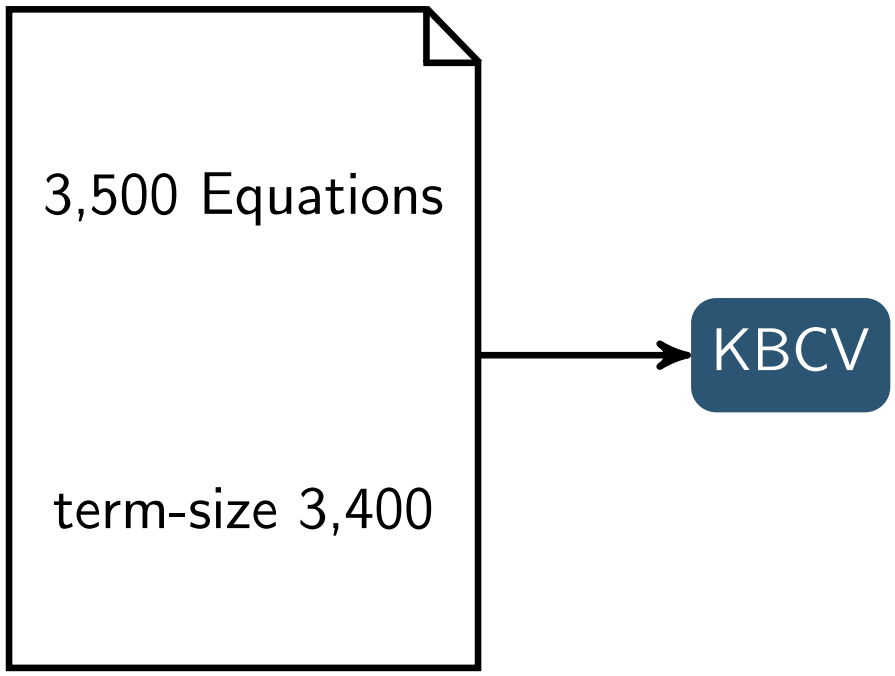
Motivation



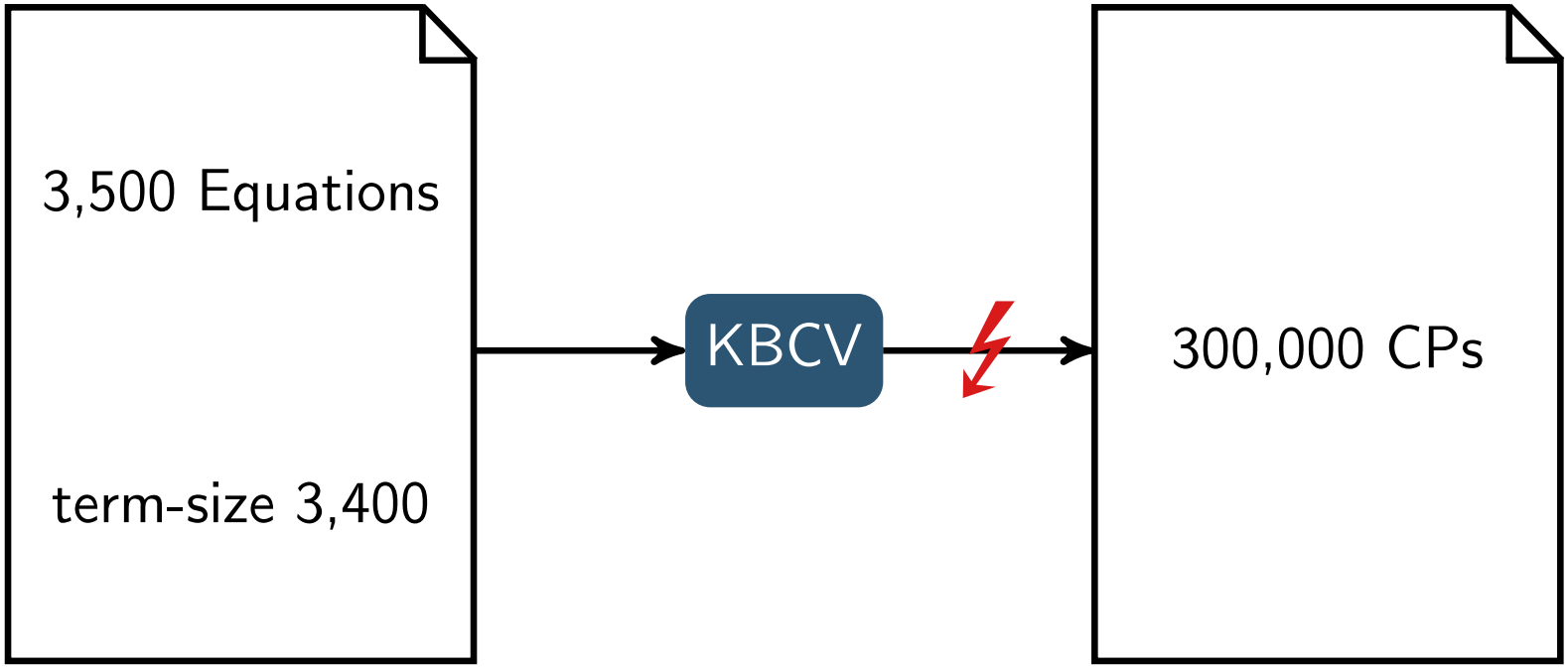
Motivation



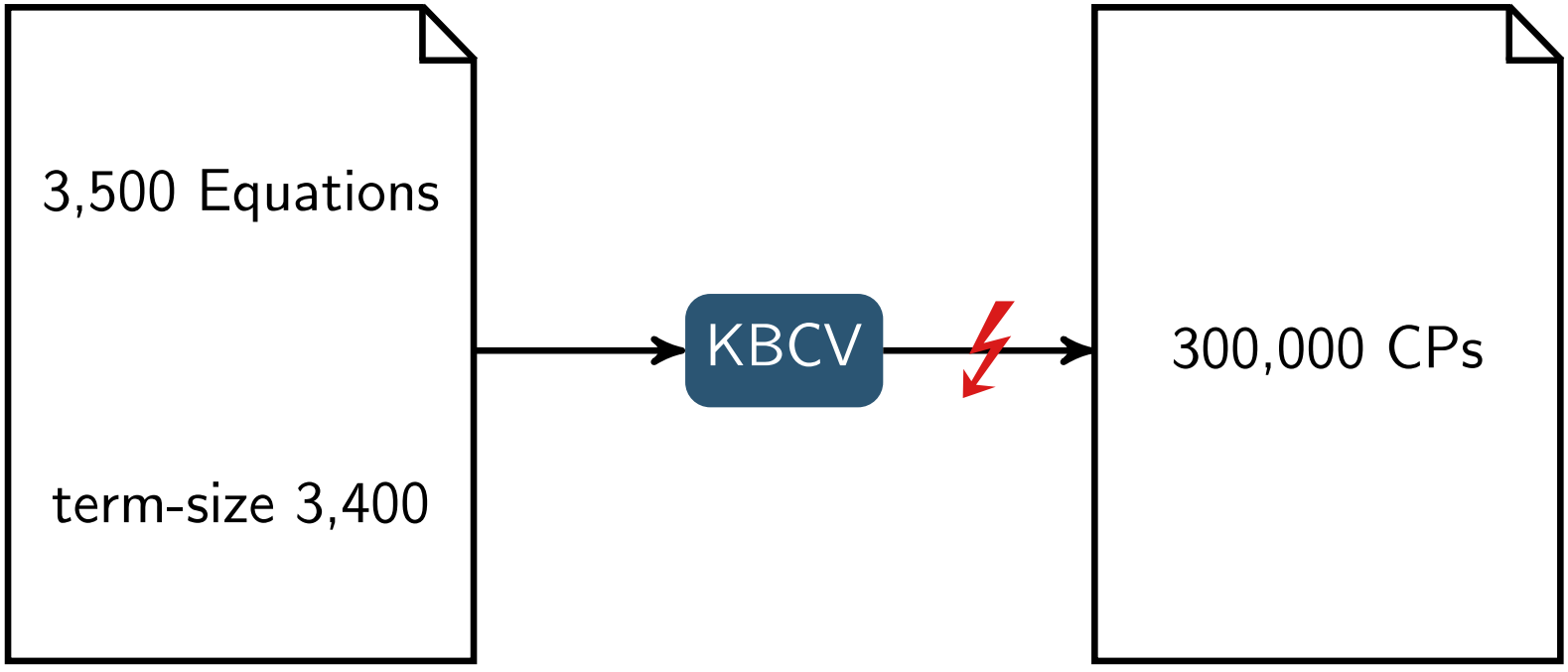
Motivation



Motivation

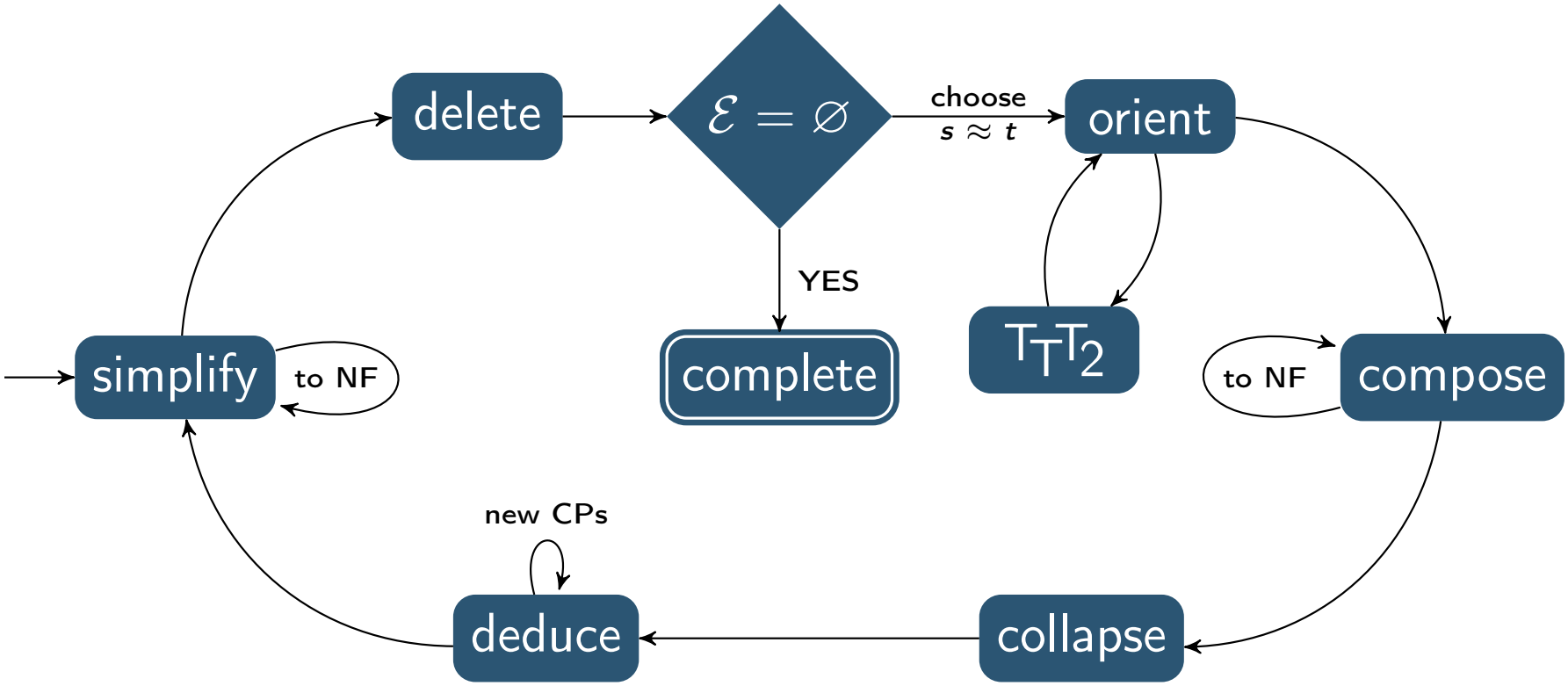


Motivation

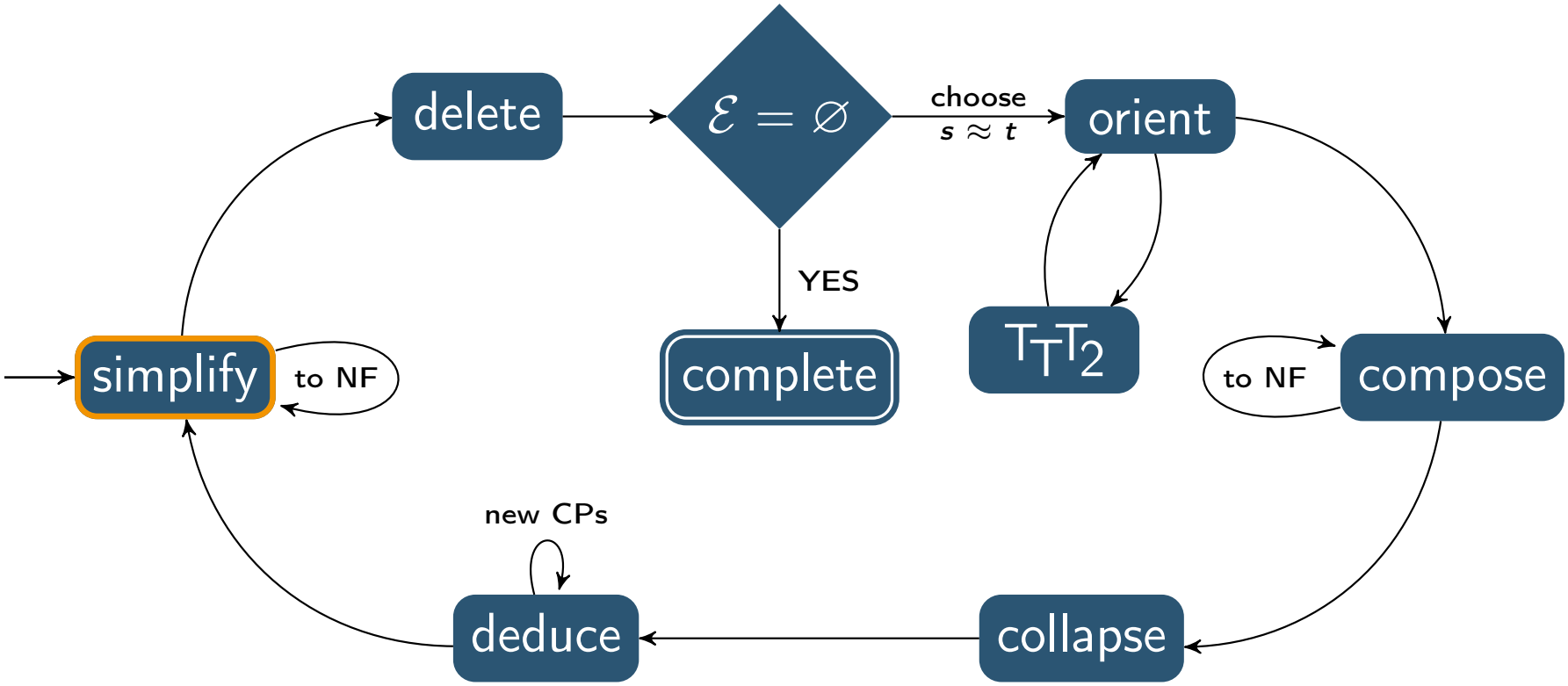


Initial Experiments on Deriving a Complete HOL Simplification Set
Cezary Kaliszyk and Thomas Sternagel
PxTP 2013, EPiC Series, 2013.

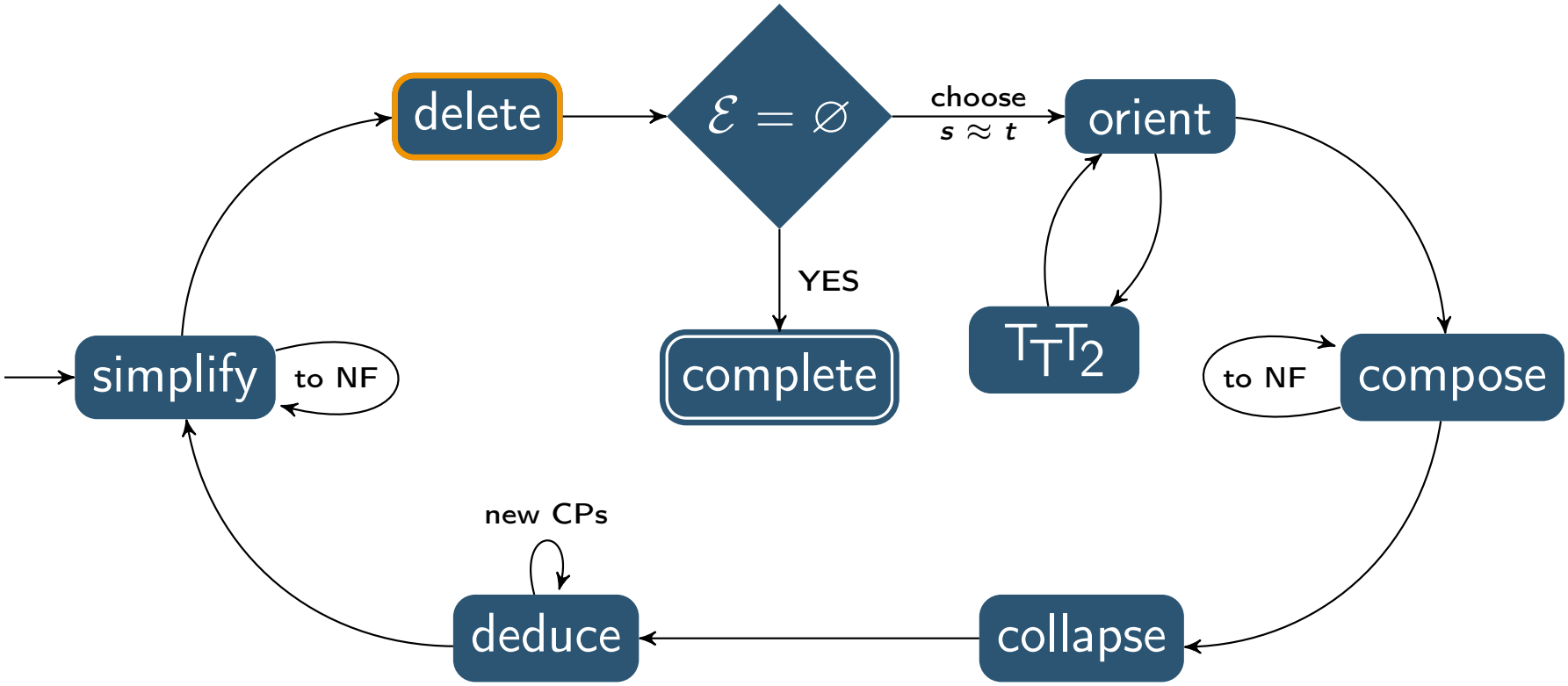
Optimization



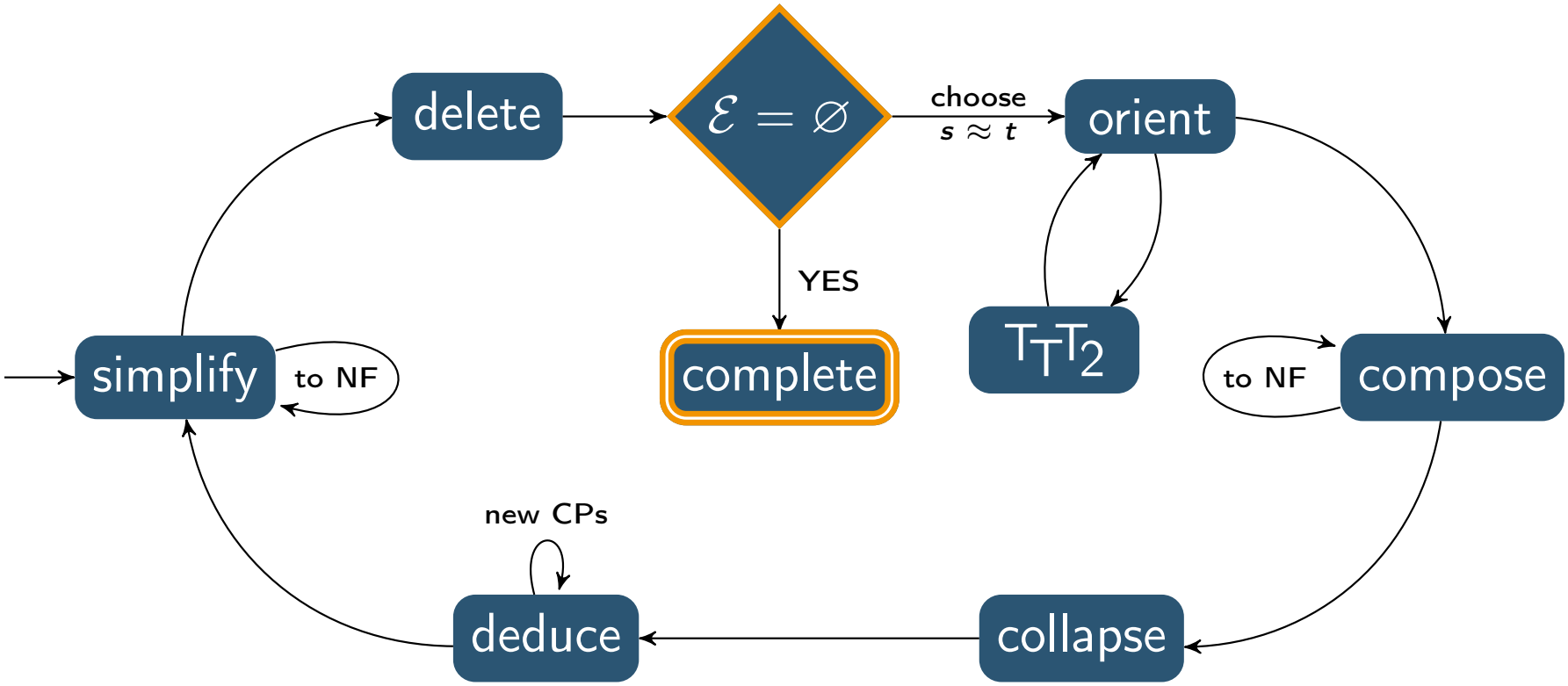
Optimization



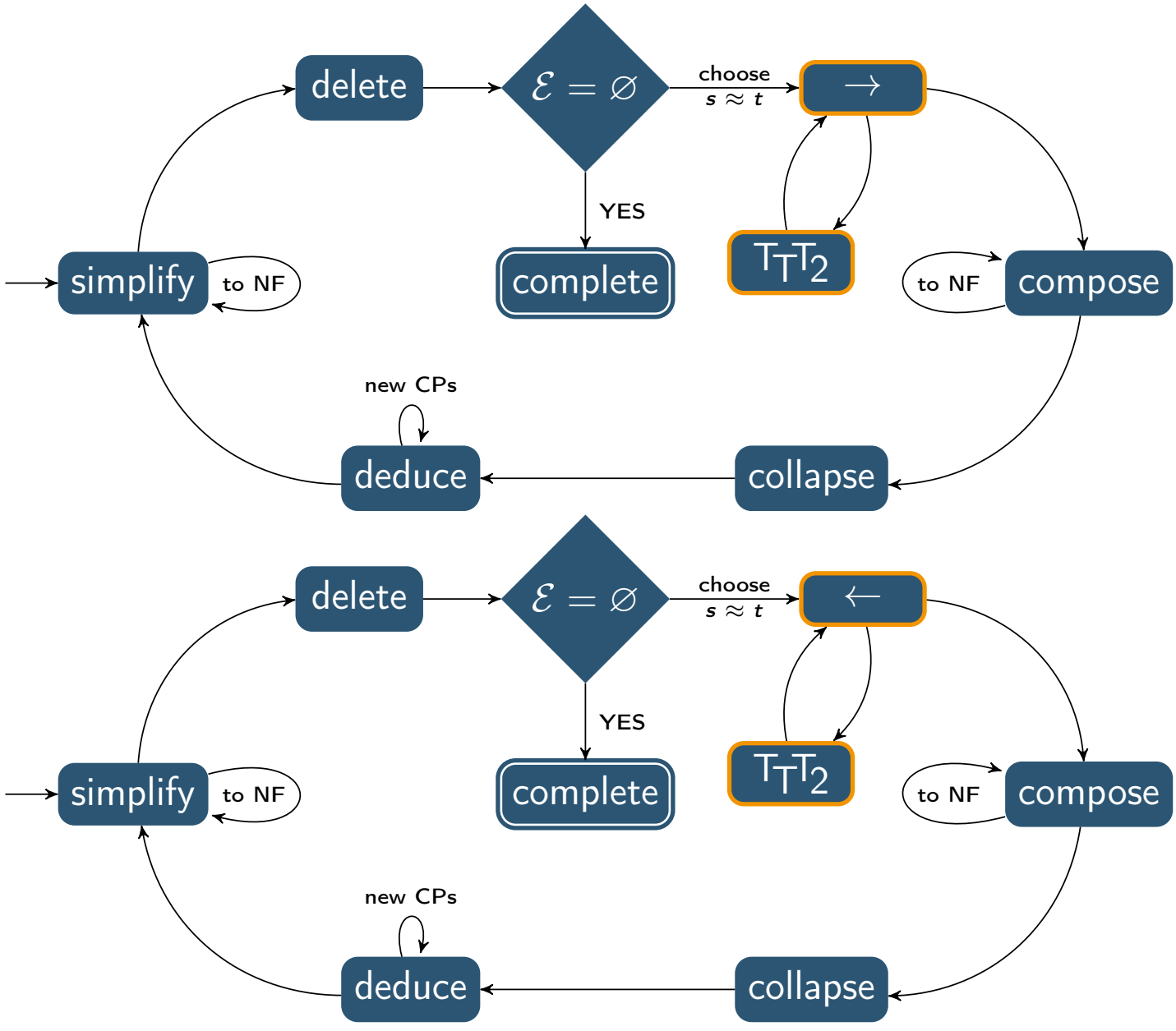
Optimization



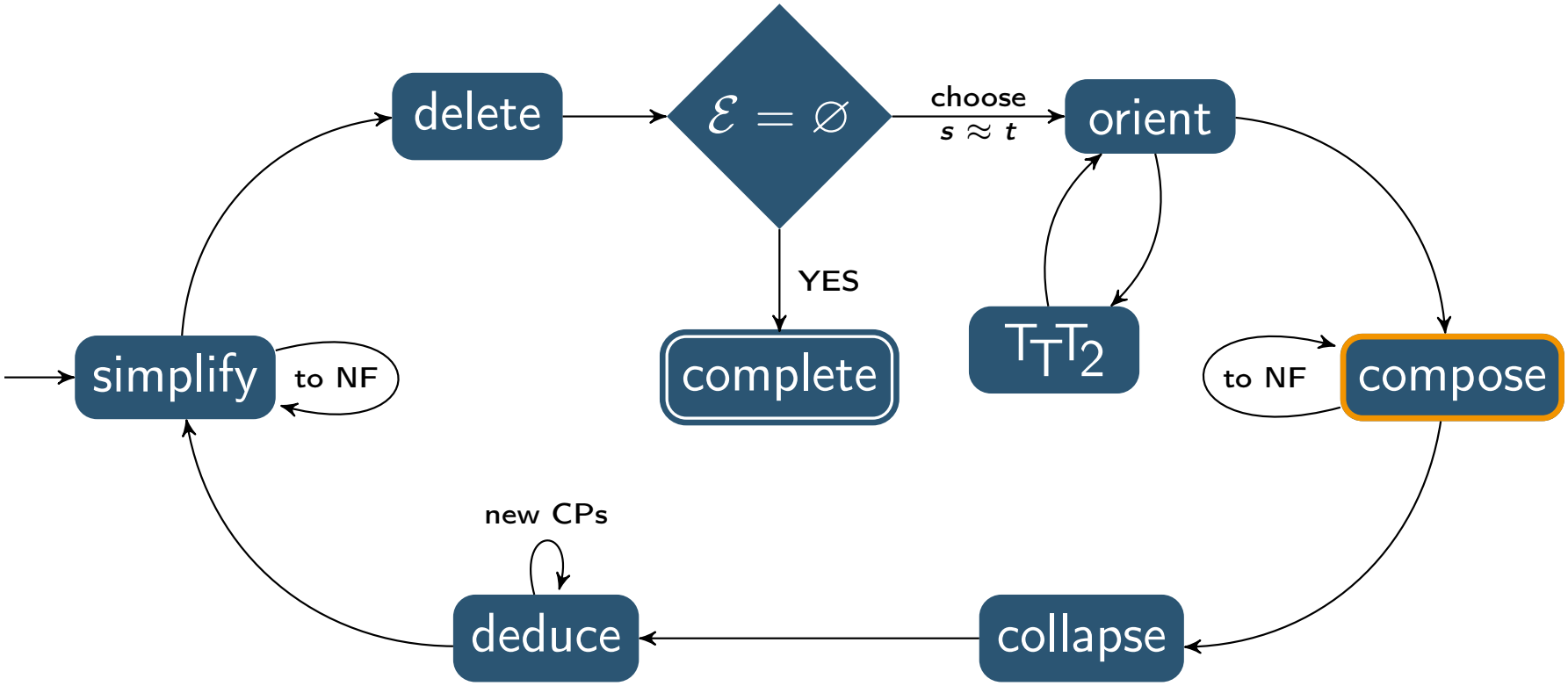
Optimization



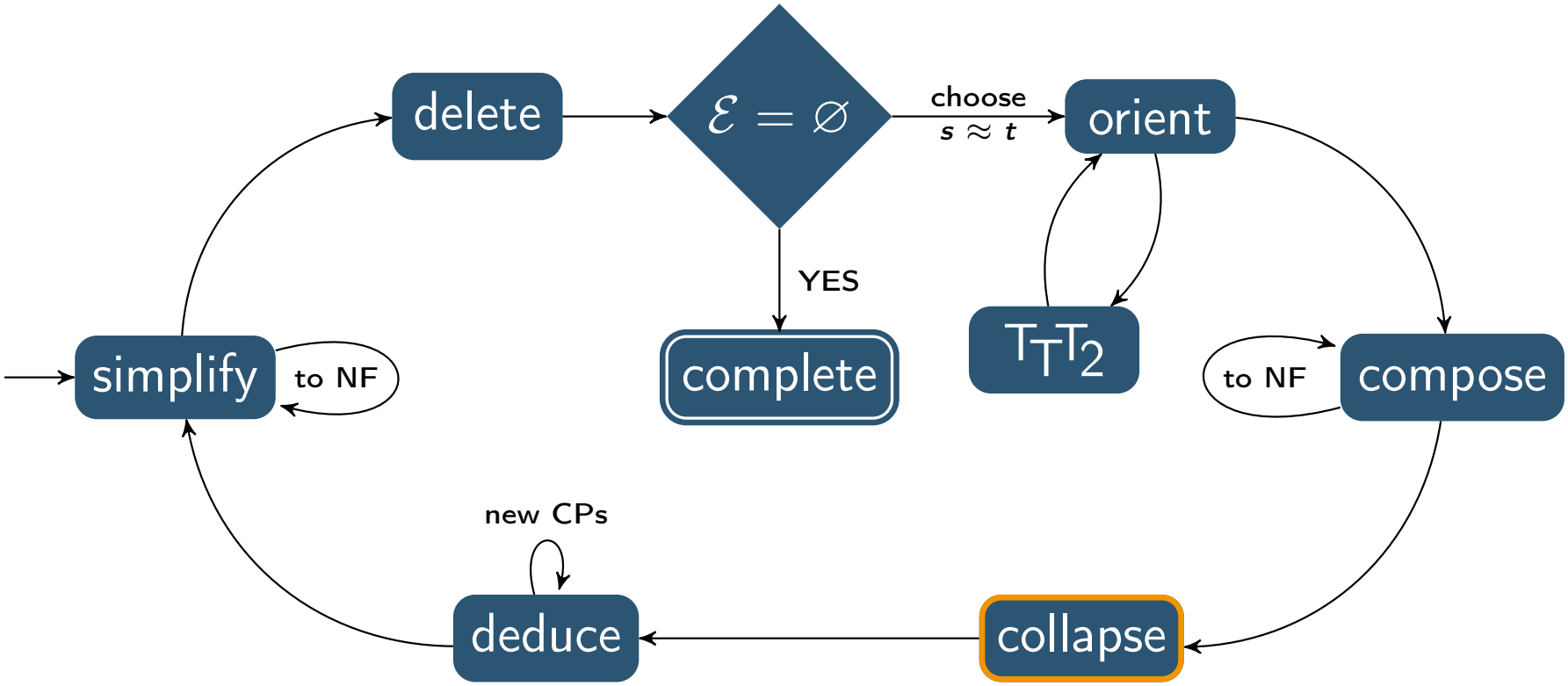
Optimization



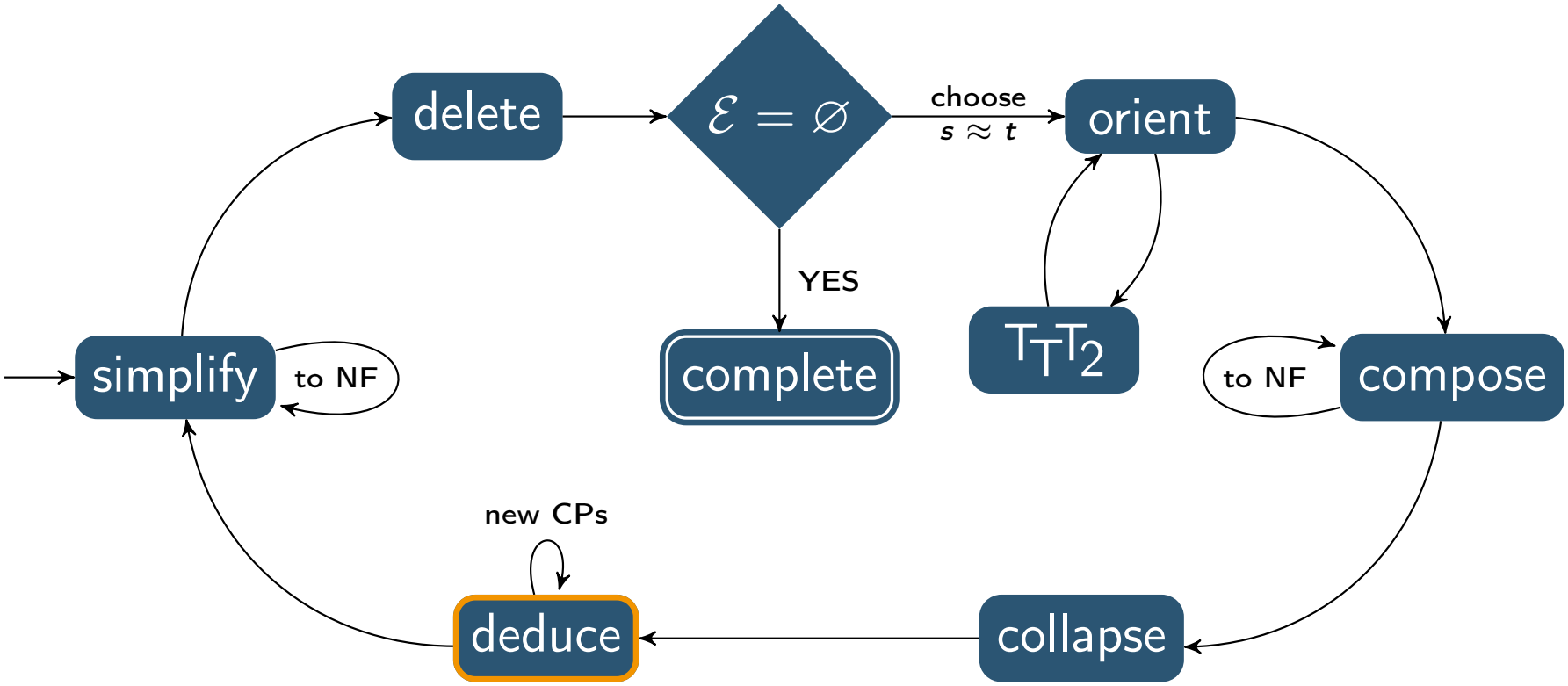
Optimization



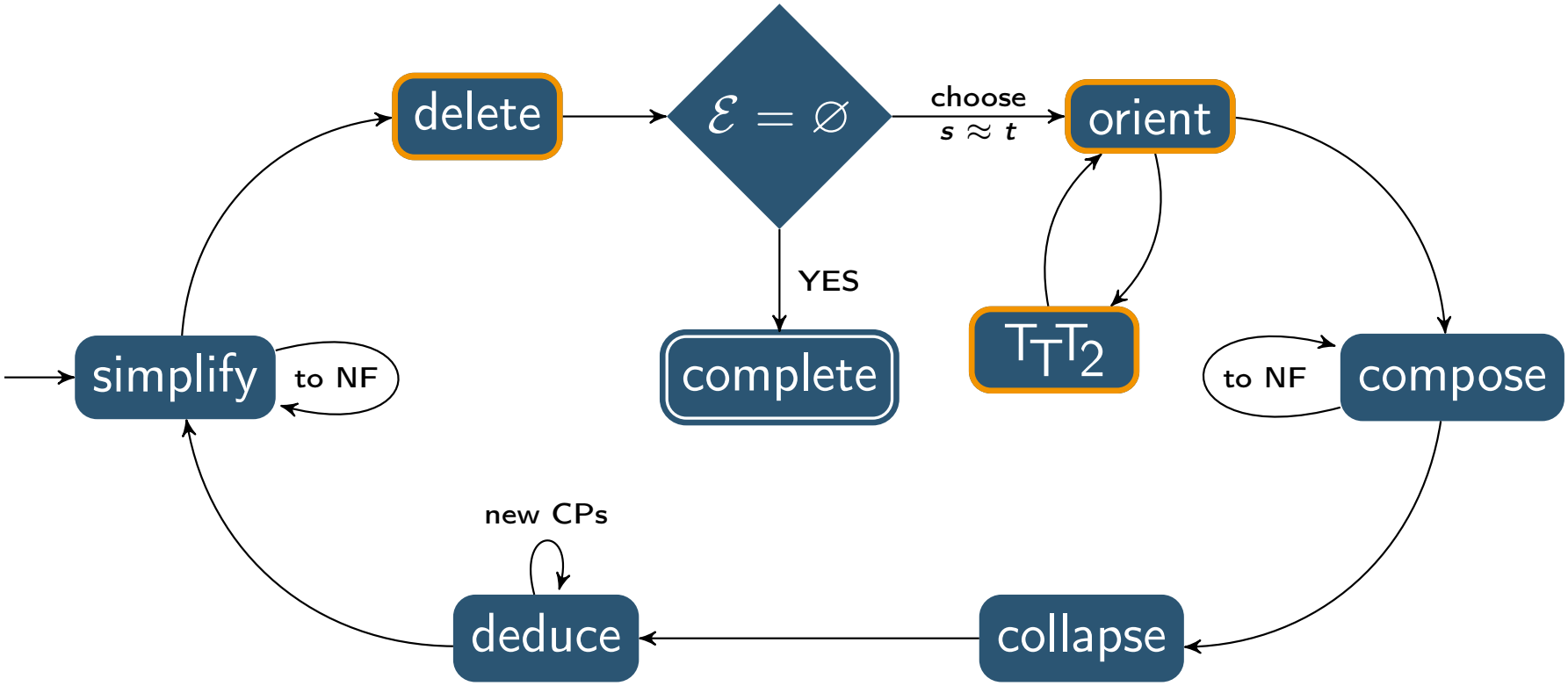
Optimization



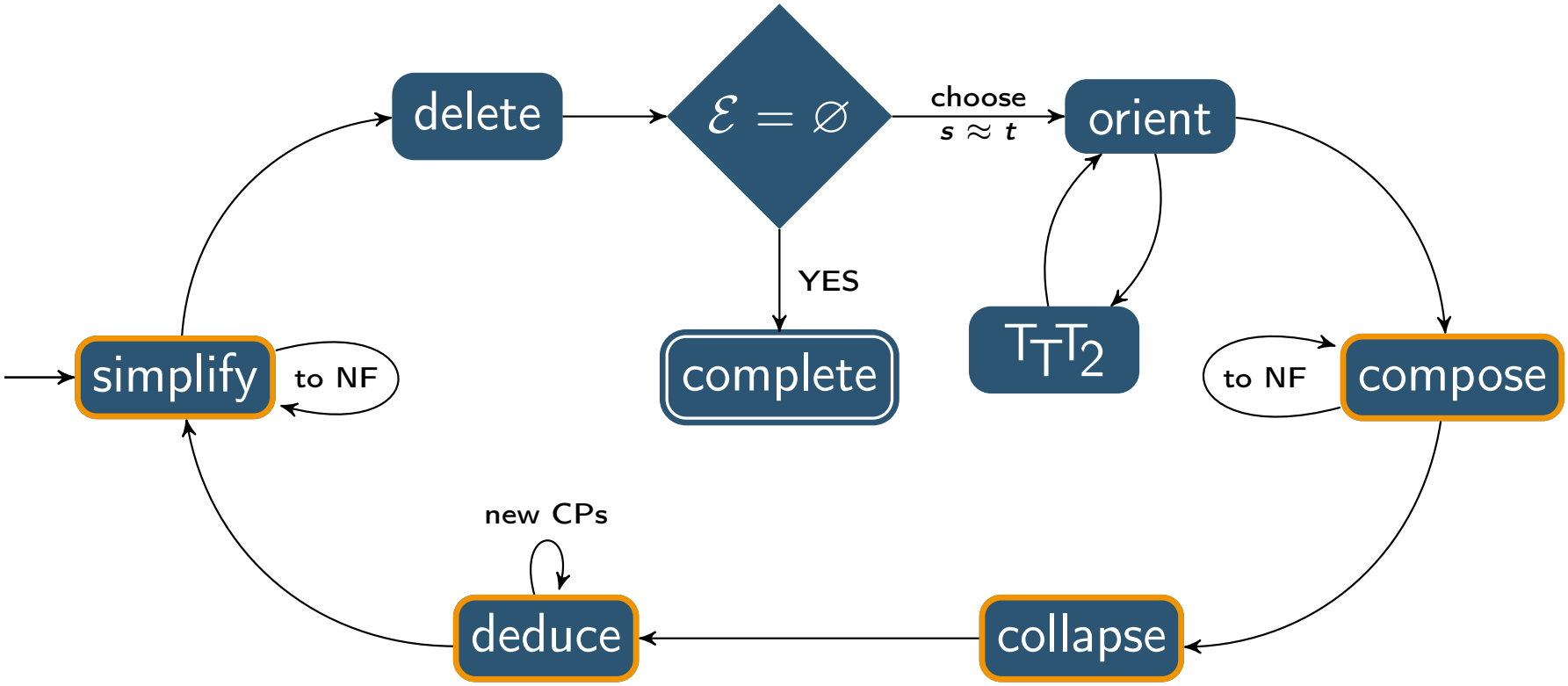
Optimization



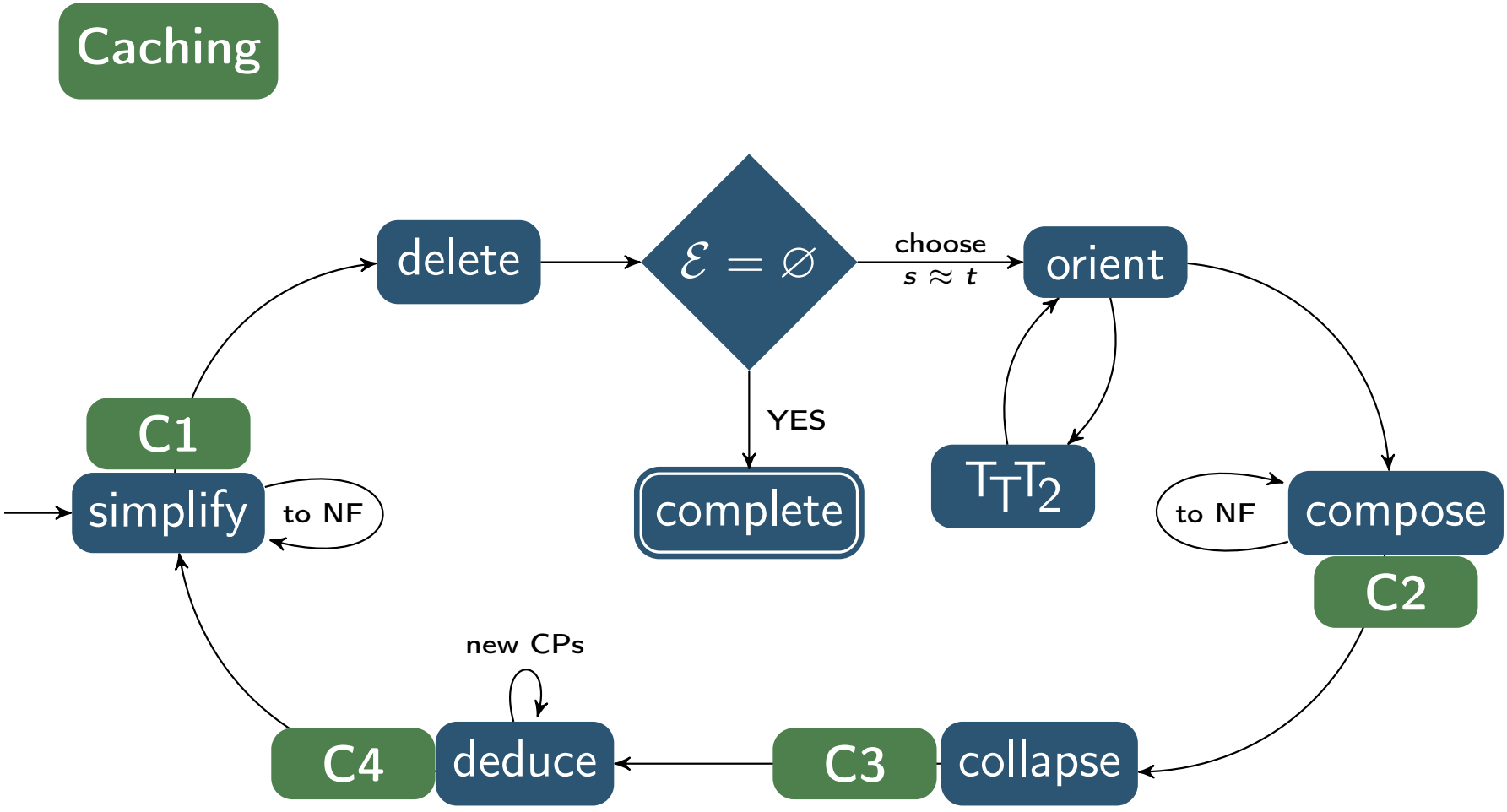
Optimization



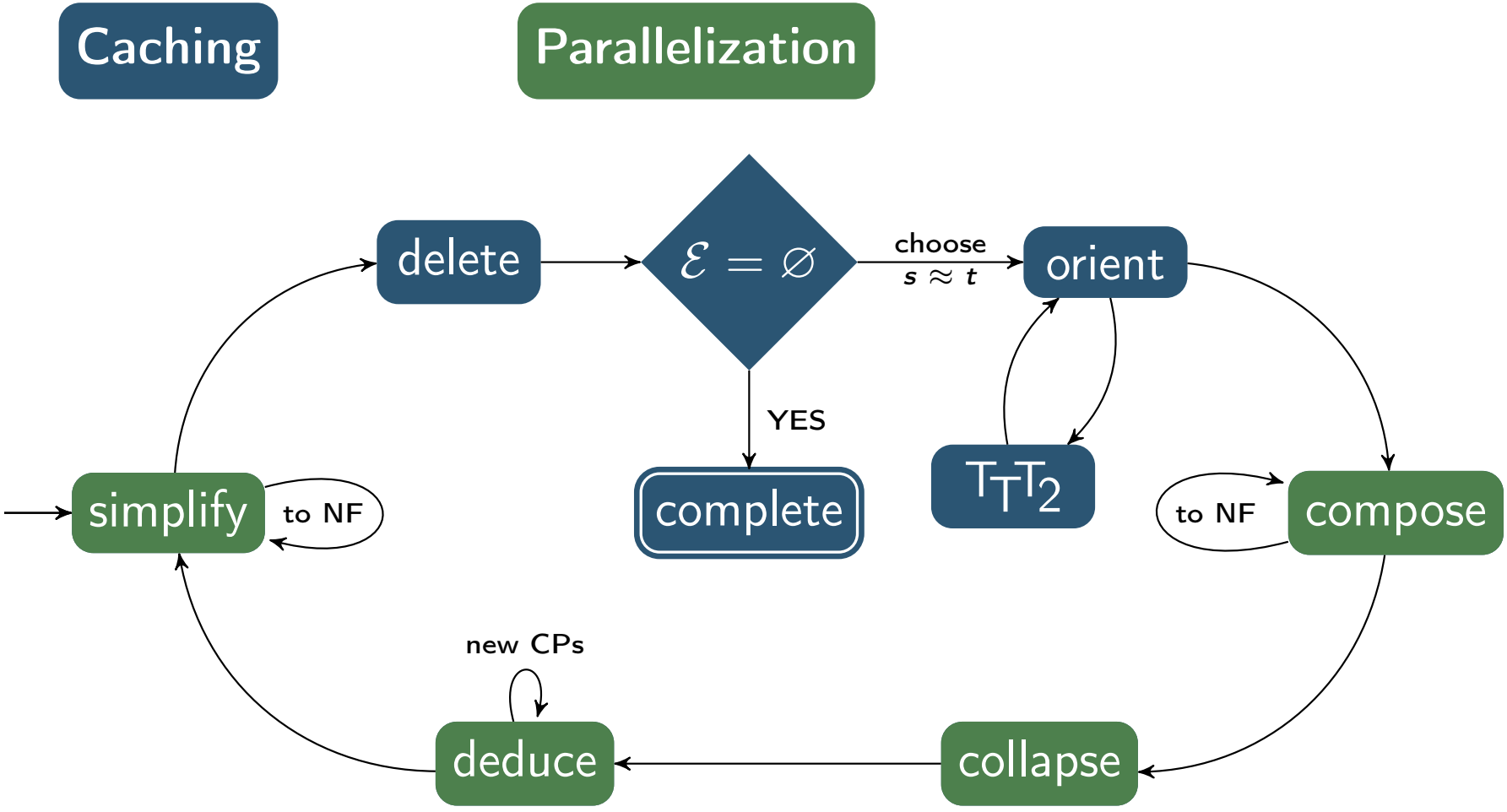
Optimization



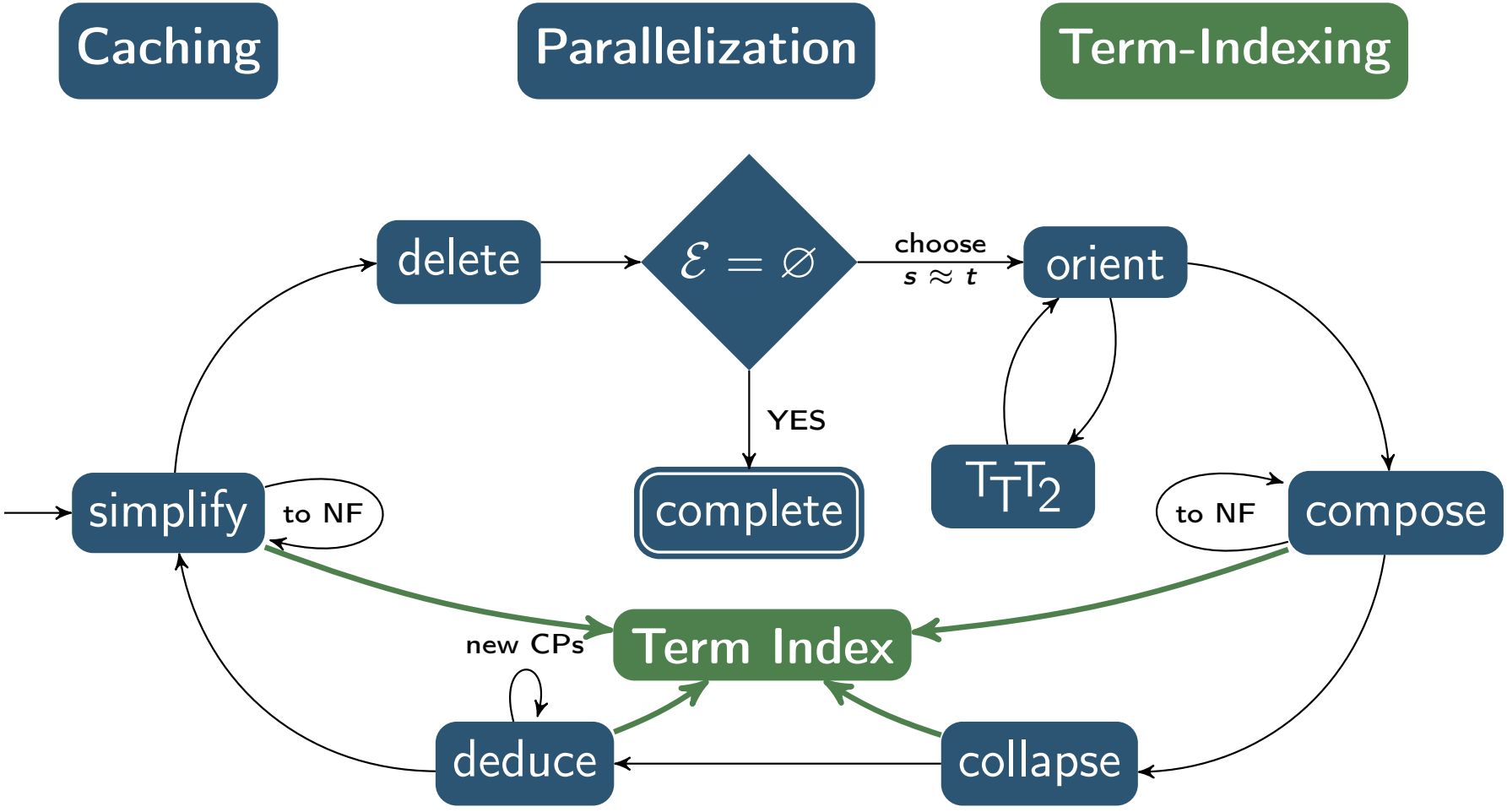
Optimization



Optimization



Optimization



C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

Experiments

C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

Experiments

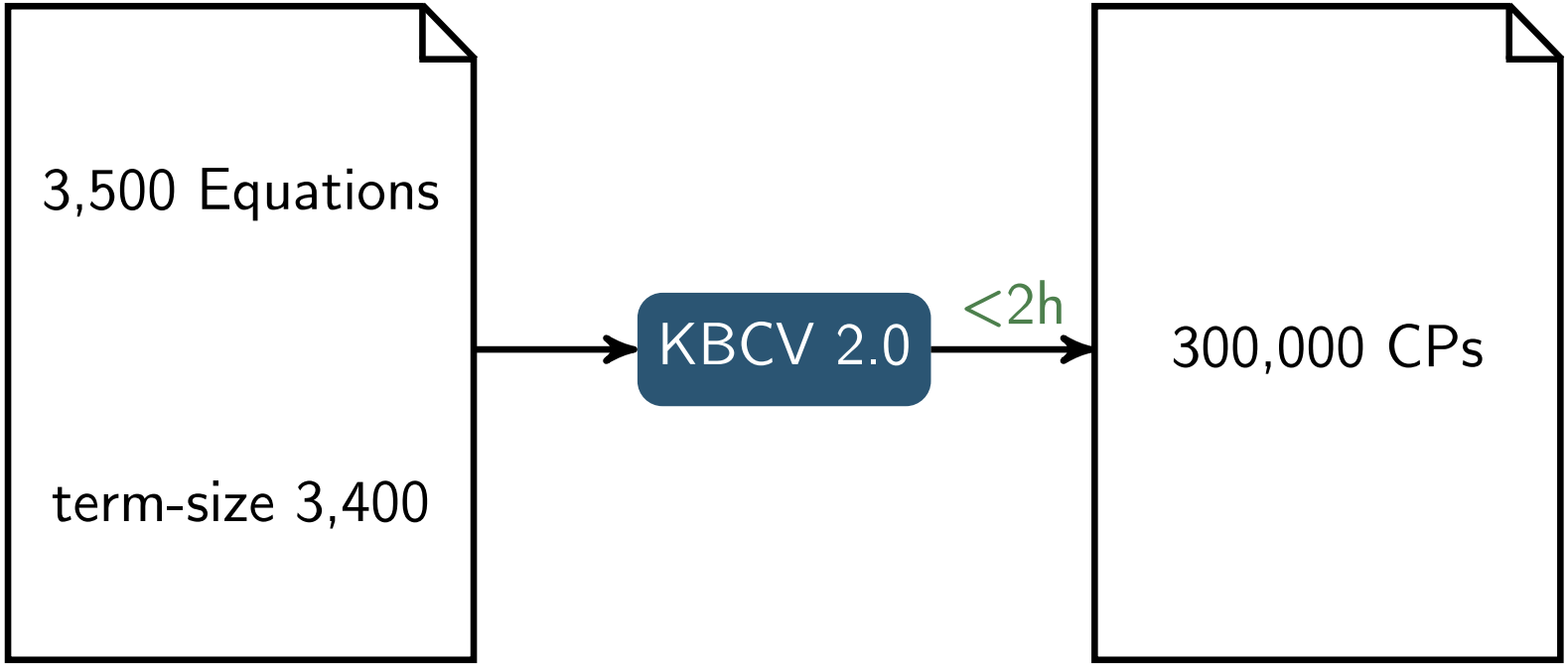
C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

C	*P*	*T*	*completed*	*avg. time*	*speed-up*
			85	13.4	1.0
		×	85	5.9	2.3
	×		85	4.5	3.0
	×	×	85	3.8	3.5
×			87	5.9	2.9
×		×	89	13.1	3.5
×	×		90	14.7	4.0
×	×	×	90	12.4	4.5

Table : 48 processors, 115 systems, timeout 10 min.

Experiments



Thank you for your attention!
Any questions?