Make E Smart Again

Zarathustra Goertzel

Czech Technical University in Prague

IJCAR 2020

*Supported by the ERC Consolidator grant no. 649043 AI4REASON and by the
Czech project AI&Reasoning CZ.02.1.01/0.0/0.0/15 003/ 0000466 and the European Regional
Development Fund.

E Prover (a Saturation-based ATP)

- **Goal: Prove conjecture from premises.**
- **E has two sets of clauses:**
  - *Processed* clauses P (initially empty)
  - *Unprocessed* clauses U (Negated Conjecture and Premises)
- **Given Clause Loop:**
  - Select '*given clause*' g to add to P
  - Apply *inference rules* to g and all clauses in P
  - Process new clauses. Add non-trivial and non-redundant ones to U.
- **Proof search succeeds when empty clause is inferred.**
- **Proof consists of some of the given clauses.**

Given Clause Loop in E

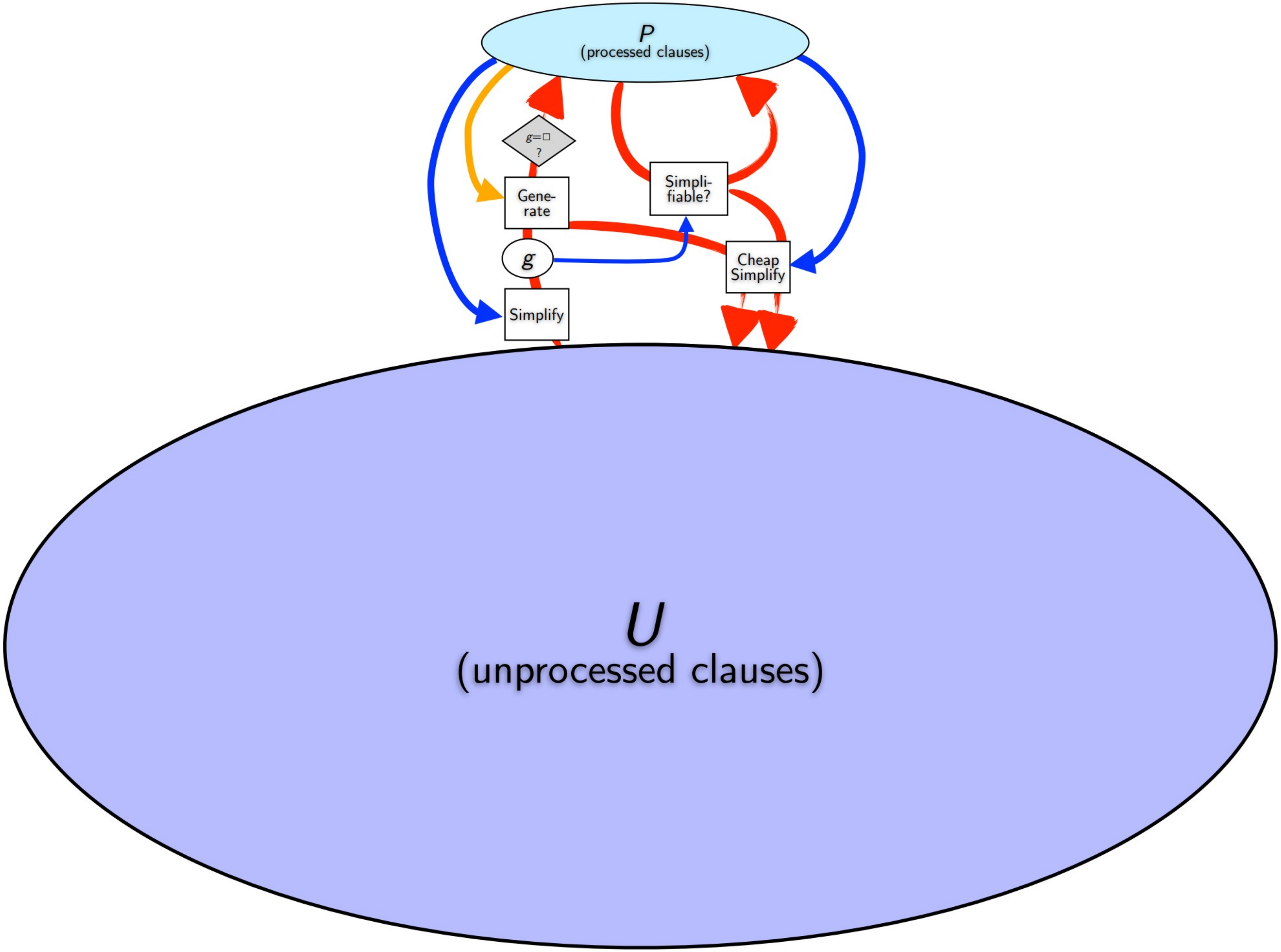


Image thanks to Stephan Schulz’s presentation on E

E Strategies

A strategy guides E’s proof search.

The main components:

- **Clause Evaluation Functions**
- **Term ordering**
- **Literal selection**

E Strategies

- *Clause Evaluation Functions* consist of:
  - *Priority functions*: partition clauses into priority queues.
    - e.g., *ConstPrio, PreferUnit*
  - **Weight functions**: order clauses in queues based on a score.
    - e.g.: **Clauseweight**, **FIFOWeight**
- Weighted by frequency of use, for example, :

-H'(5*Clauseweight(ConstPrio,1,1,1),
1*FIFOWeight(ConstPrio))'

Strategy E1

--definitional-cnf=24 --split-aggressive --simul-paramod --forward-context-sr --destructive-er-aggressive --destructive-er --prefer-initial-clauses -tKBO -winvfreqrank -c1 -Ginvfreq -F1 --delete-bad-limit=150000000 -WSelectMaxLComplexAvoidPosPred -H(1*ConjectureTermPrefixWeight(DeferSOS,1,3,0.1,5,0,0.1,1,4),1*ConjectureTermPrefixWeight(DeferSOS,1,3,0.5,100,0,0.2,0.2,4),1*Refinedweight(PreferWatchlist,4,300,4,4,0.7),1*RelevanceLevelWeight2(PreferProcessed,0,1,2,1,1,1,200,200,2.5,9999.9,9999.9),1*StaggeredWeight(DeferSOS,1),1*SymbolTypeweight(DeferSOS,18,7,-2,5,9999.9,2,1.5),2*Clauseweight(PreferWatchlist,20,9999,4),2*ConjectureSymbolWeight(DeferSOS,9999,20,50,-1,50,3,3,0.5),2*StaggeredWeight(DeferSOS,2))

Strategy E2

--definitional-cnf=24 --split-aggressive --split-reuse-defs --simul-paramod --forward-context-sr --destructive-er-aggressive --destructive-er --prefer-initial-clauses -tKBO -winvfreqrank -c1 -Ginvfreq -F1 --delete-bad-limit=150000000 -WSelectMaxLComplexAvoidPosPred -H(3*ConjectureRelativeSymbolWeight(PreferUnitGroundGoals,0.1,100,100,50,100,0.3,1.5,1.5),4*FIFOWeight(PreferNonGoals),5*RelevanceLevelWeight2(ConstPrio,1,0,2,1,50,-2,-2,100,0.2,3,4))

E Strategies

- *Term orderings*:
  - Determine rewrite order
  - Contribute to completeness of proof search
  - Prevent cycles

E Strategies

- Term orderings:
  - Determine rewrite order
  - Contribute to completeness of proof search
  - Prevent cycles

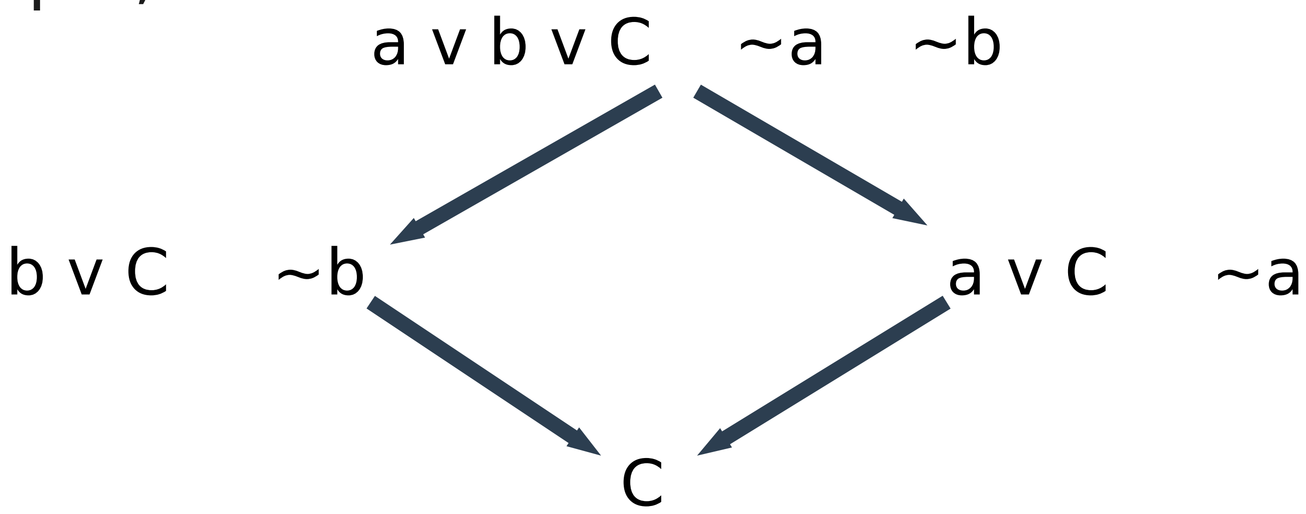
- For example, given $x+0=0$,
  - $x+0 \rightarrow x$ is valid
  - $x \rightarrow x+0$ is invalid

E Strategies

- Term orderings:
  - Determine rewrite order
  - Contribute to completeness of proof search
  - Prevent cycles
- For example, given *x+0=0*,
  - x+0 → x is valid
  - x → x+0 is invalid
- Our strategies use KBO: Kunth-Bendix ordering

E Strategies

- Literal selection limits resolution.
  - Limits redundancy
  - Contributes to completeness.
- For example,



ENIGMA

- **Statistical Learning**
  - XGBoost
- **Learns from given-clauses**
- ***Positive* and *Negative***
- **Maps clauses to vectors**
- **Weight function**
- **Ranks all clauses**

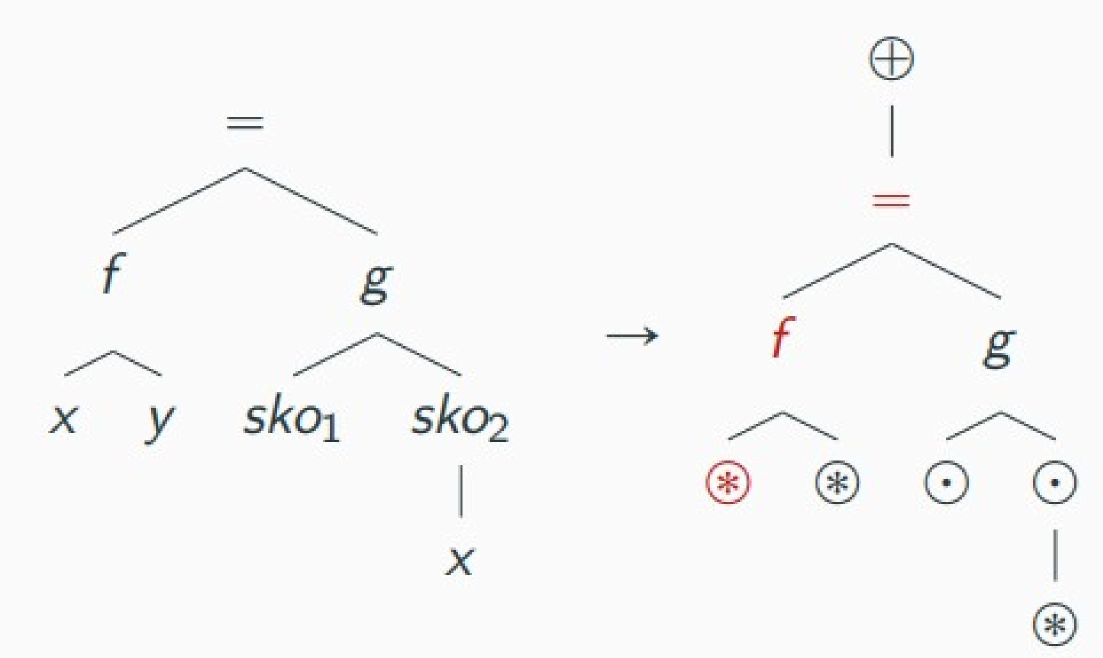
- Input:
  - Positive examples + conjecture features
  - Negative examples + conjecture features

- Output:
  - Fast model to predict whether (clause, conjecture) pairs are *positive* or *negative*.

Clauses → Vectors

- Treat clauses as trees. Abstract vars and skolem symbols

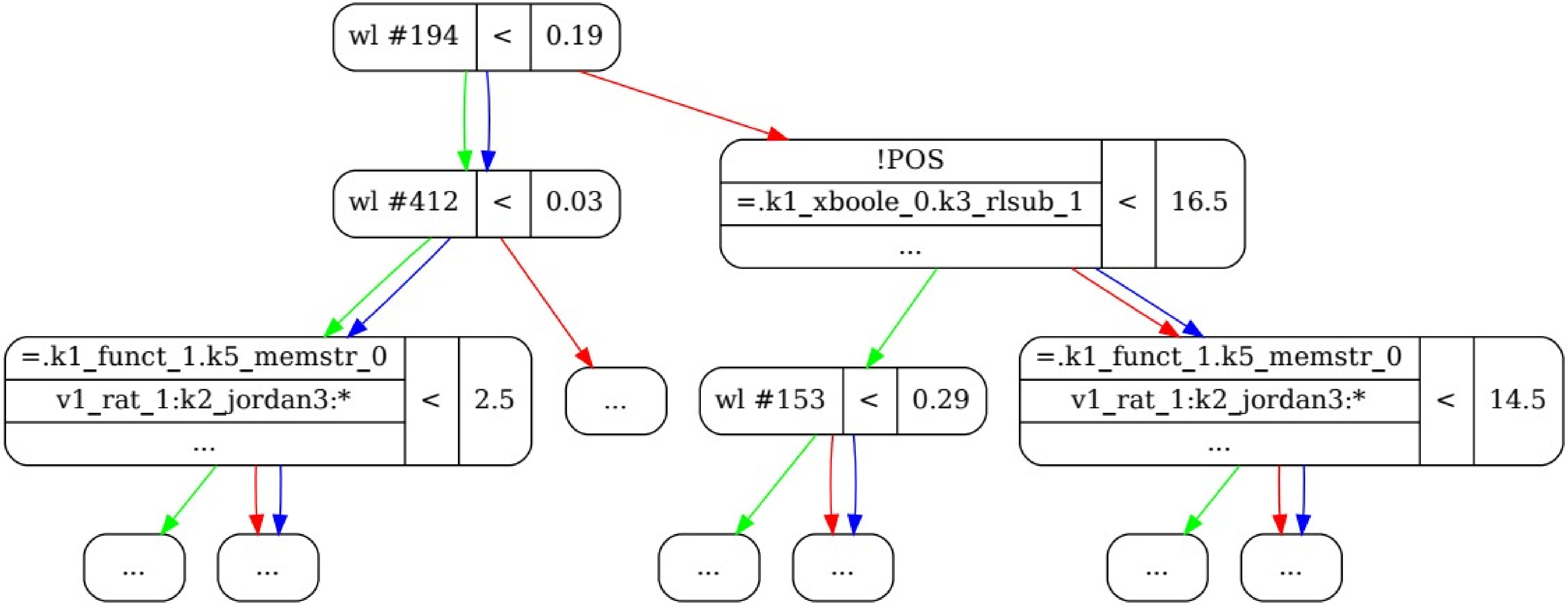
For example: $f(x, y) = g(\mathrm{sko}_1, \mathrm{sko}_2(x))$



- Enumerate features:
  Count features in a clause for its vector

#	feature	count
1	(⊕,=,a)	0
⋮	⋮	⋮
11	(⊕,=,f)	1
12	(⊕,=,g)	1
13	(=,f,⊛)	2
14	(=,g,⊙)	2
15	(g,⊙,⊛)	1
⋮	⋮	⋮

XGBoost Example Tree



Dataset

- Mizar Mathematical Library (MML) contains 1148 articles and 57880 theorems:
  - including *Bolzano-Weierstrass* and *Gödel's completeness theorem*

- An interactive theorem proving system

- Premises are already selected.

Previous ENIGMA results:

On all 57880 Mizar problems:

- *E1*: 14526
- *E2:* 12778
- *ENIGMA:* 25562 (+76%)

Research Question

- **Can ENIGMA learn to guide E without its strategies?**

Method**:**

1) Make E stupid.

2) Try to learn!

Making E Stupid: E0

1) **Replace KBO with structural identity relation**
   - (Which disables term rewriting!)

2) **Disable literal selection**

3) **Use only the basic strategy:**

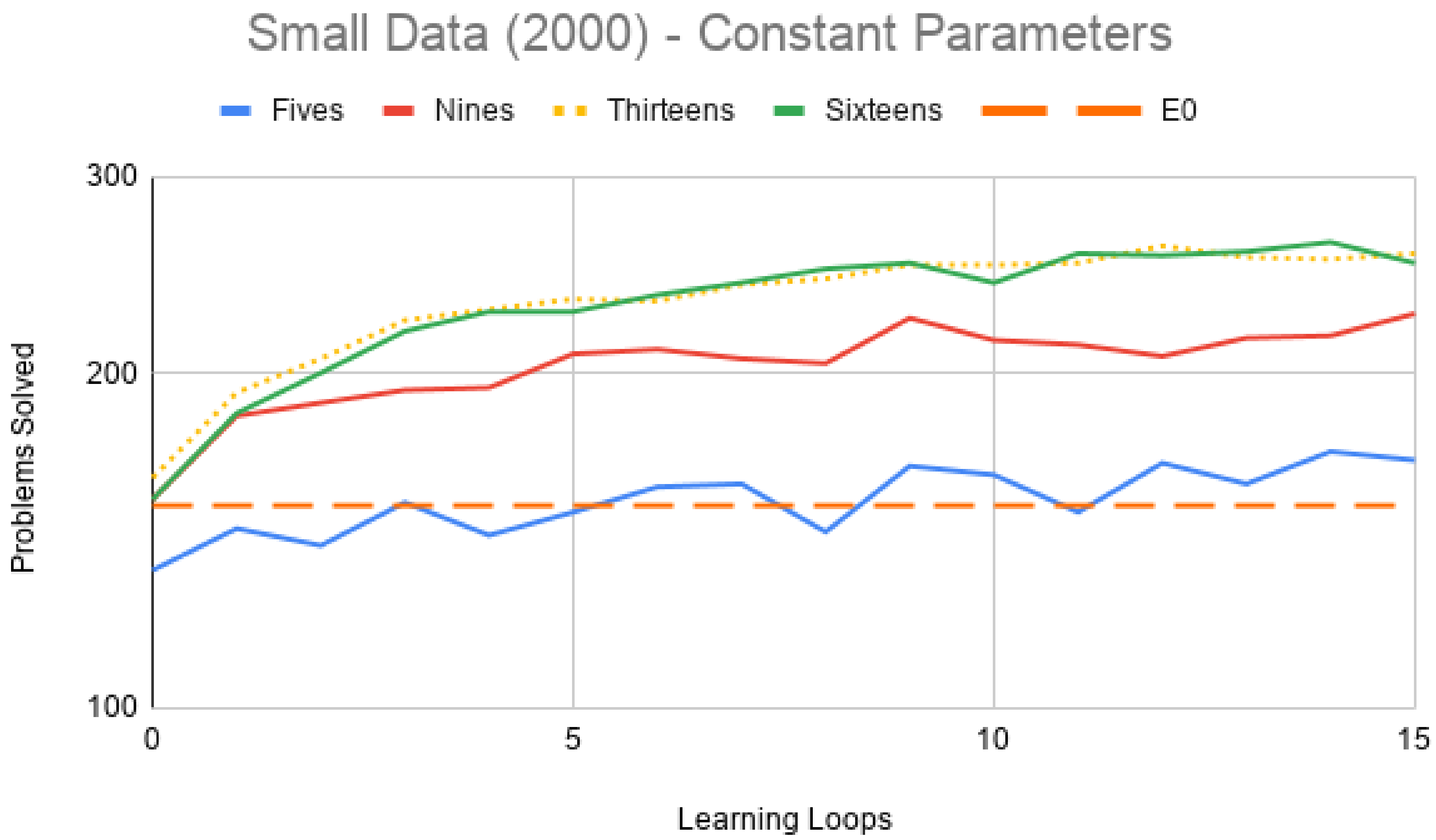
--definitional-cnf=24 --prefer-initial-clauses
-tIDEN –restrict-literal-comparisons
-H’(5*Clauseweight(ConstPrio,1,1,1),
1*FIFOWeight(ConstPrio))’

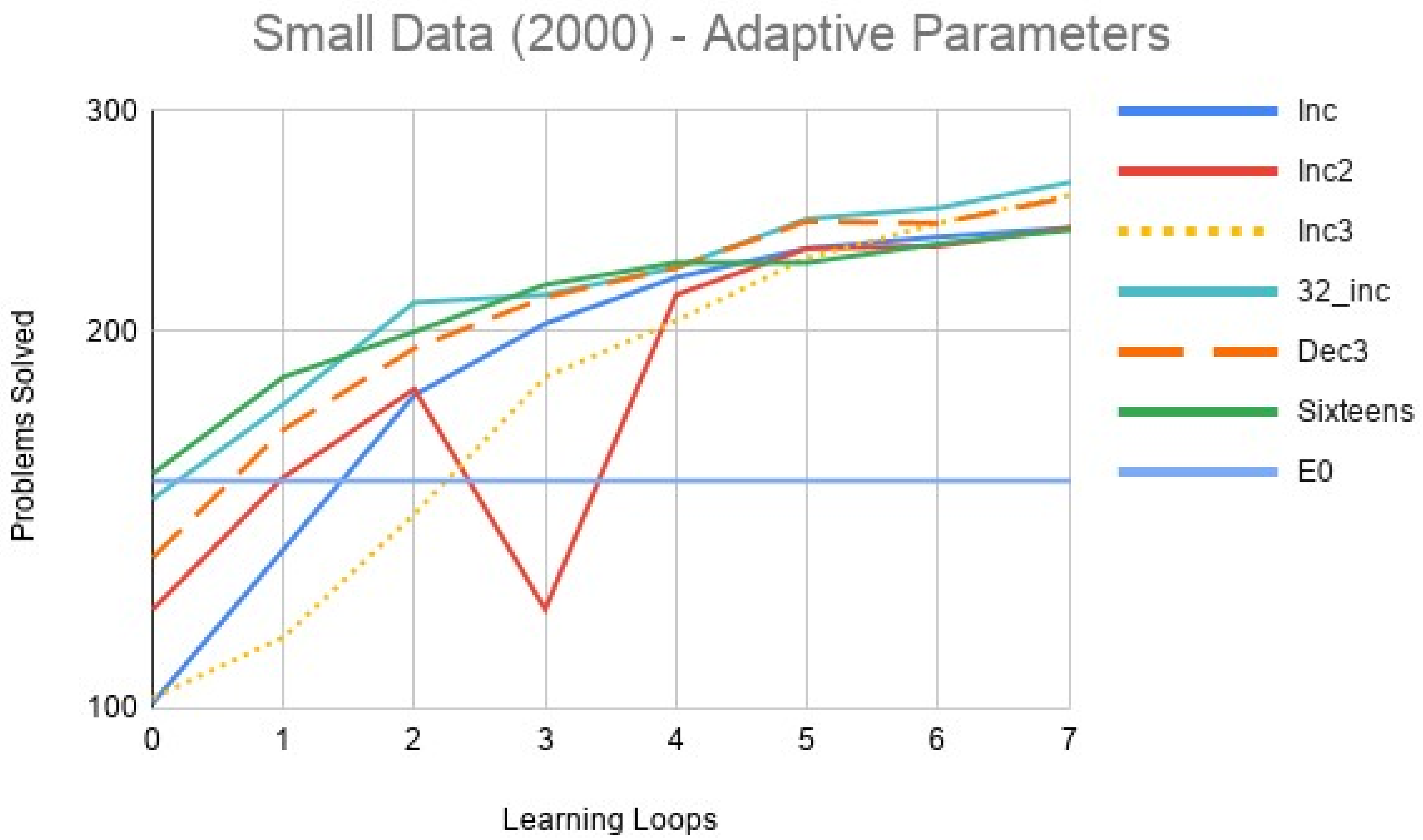
ENIGMA Training

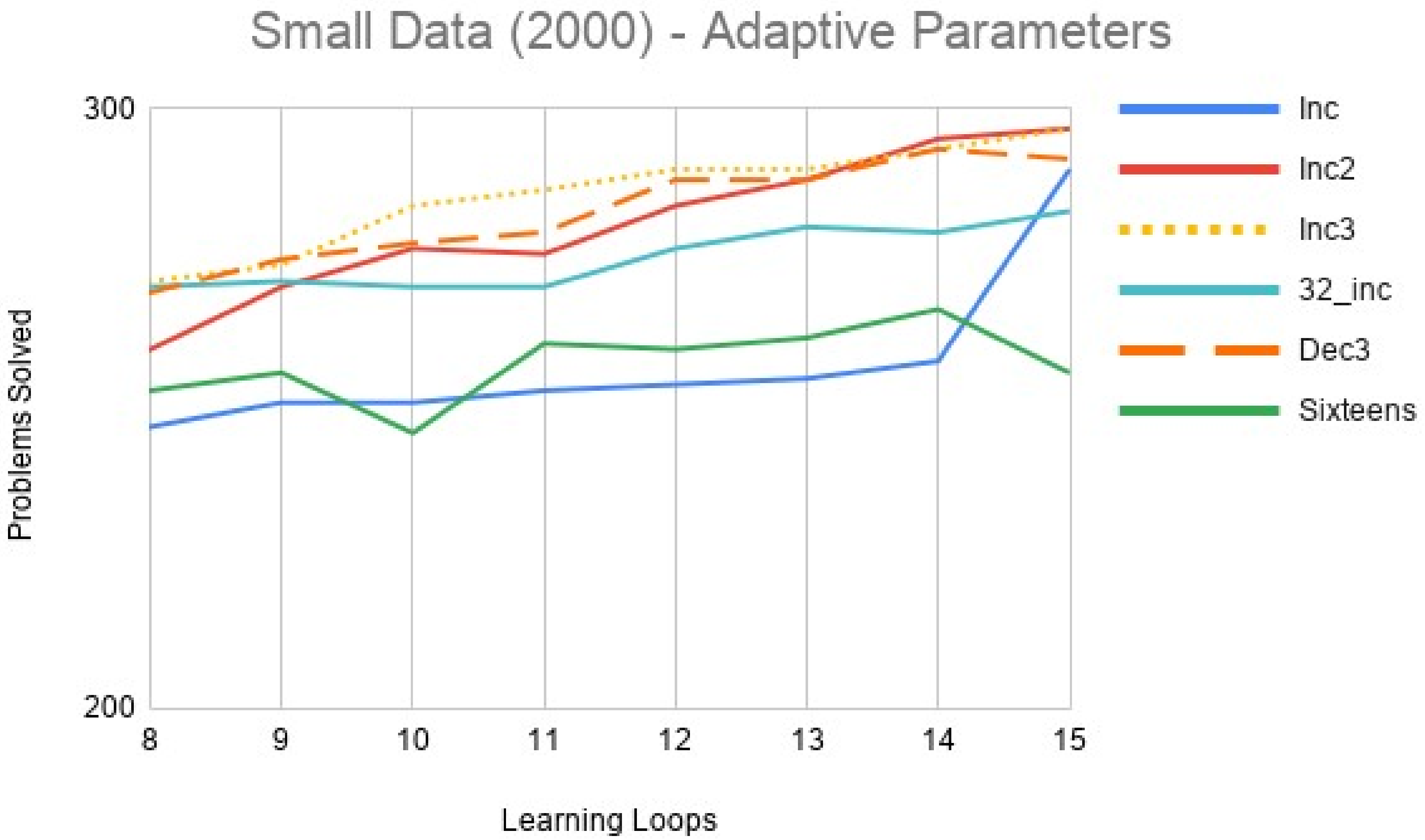
1) Run E0
2) Train a model.
3) Run E0 with the model (loop 0).
4) Repeat.

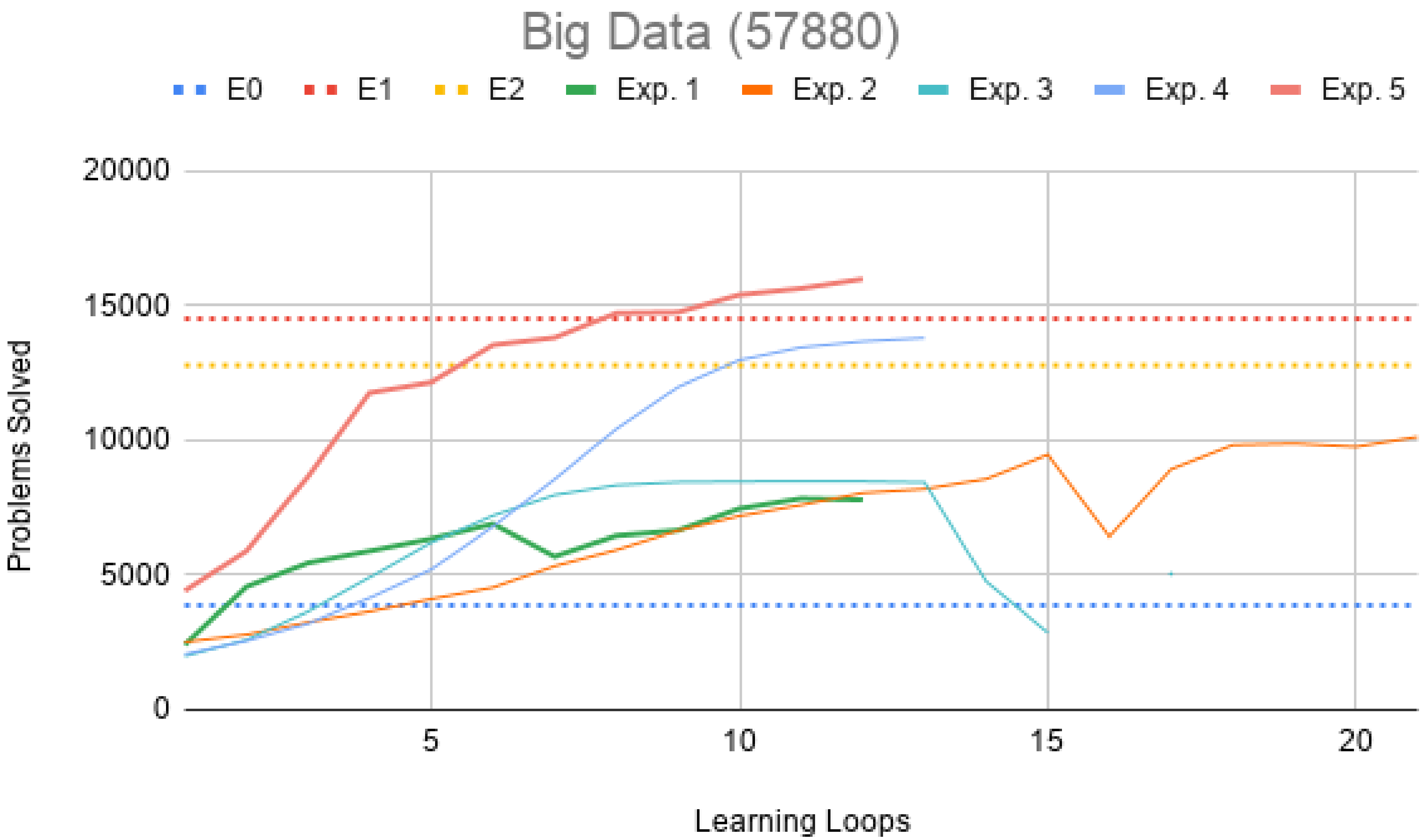
Hyper-parameters:

- Number of loops
- Number of trees per loop
- Depth of trees


Small Data (2000) - Constant Parameters
Fives
Nines
Thirteens
Sixteens
E0
300
200
100
Problems Solved
0
5
10
15
Learning Loops


Small Data (2000) - Adaptive Parameters
Problems Solved
300
200
100
0
1
2
3
4
5
6
7
Learning Loops
Inc
Inc2
Inc3
32_inc
Dec3
Sixteens
E0


Small Data (2000) - Adaptive Parameters
300
200
Problems Solved
8
9
10
11
12
13
14
15
Learning Loops
Inc
Inc2
Inc3
32_inc
Dec3
Sixteens



Big Data (57880)
E0
E1
E2
Exp. 1
Exp. 2
Exp. 3
Exp. 4
Exp. 5
Problems Solved
20000
15000
10000
5000
0
5
10
15
20
Learning Loops

Conclusion

- Can ENIGMA learn to guide E without its strategies?
  - Yes!
  - 256% increase w/ additional training data.
  - 156% increase trained on E0 data alone.
- Term ordering and literal selection are still helpful.

- Can the gap be bridged?
  - Perhaps by ENIGMA-NG..
- Will relaxing term ordering help combine strategies?

References

- **Stephan Schulz' slides on E with good graphics: http://aitp-conference.org/2016/slides/StSGuidance.pdf**