Decision heuristics based on an Abstraction/Refinement model

(HaifaSat)

Ofer Strichman

Roman Gershman

Technion

SAT solving

- "Naïve" point of view:
  - Searches in the decision tree, prunes subspaces.
  - Creates "blocking clauses" that restrain the solver from choosing the same bad path again.

- This point of view fails to explain why
  - We can solve many formulas with 10^5 variables,
  - We cannot solve other formulas with 10^3 variables

A different point of view

- Modern solvers act as proof engines based on resolution, rather than as search engines, with structured problems.
- Evidence: adding the shortest conflict clauses is not the best strategy [R04].
- Furthermore: certain strategies resemble a proof by abstraction-refinement.

Abstraction of models and formulas

- Model $\widehat{\widehat{M}}$ is an (over approximating) abstraction of M if:

$$\forall a.\ a \models M \rightarrow \alpha \models \widehat{\widehat{M}}$$

A degenerated case:

- Formula $\widehat{\widehat{F}}$ is an (over-approximation) abstraction of F if:

$$\forall a.\ a \models F \rightarrow \alpha \models \widehat{\widehat{F}}$$

or simply: $\boxed{F \rightarrow \widehat{\widehat{F}}}$

Abstraction of formulas

- □ $F \rightarrow \widehat{\widehat{F}}$

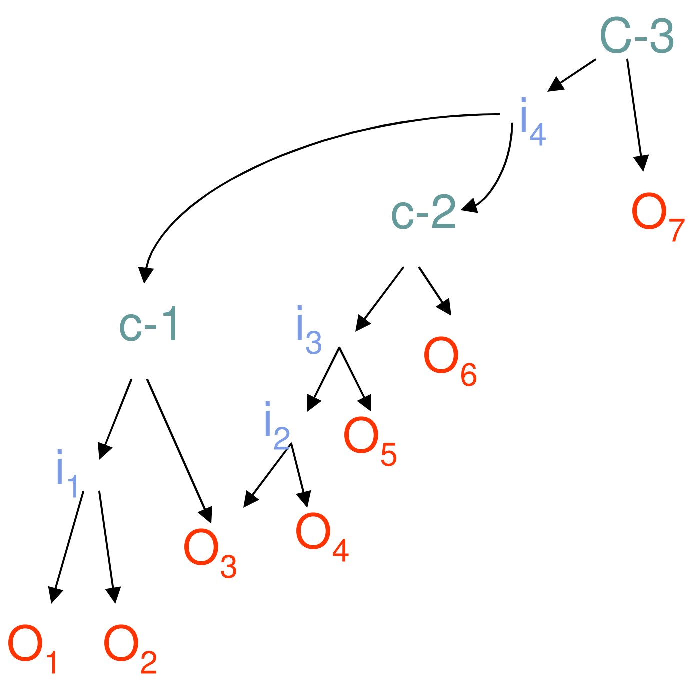
- □ Now consider Binary Resolution:

$$\underbrace{(A \vee x) \wedge (B \vee \neg x)} \quad \rightarrow \quad \underbrace{(A \vee B)}$$

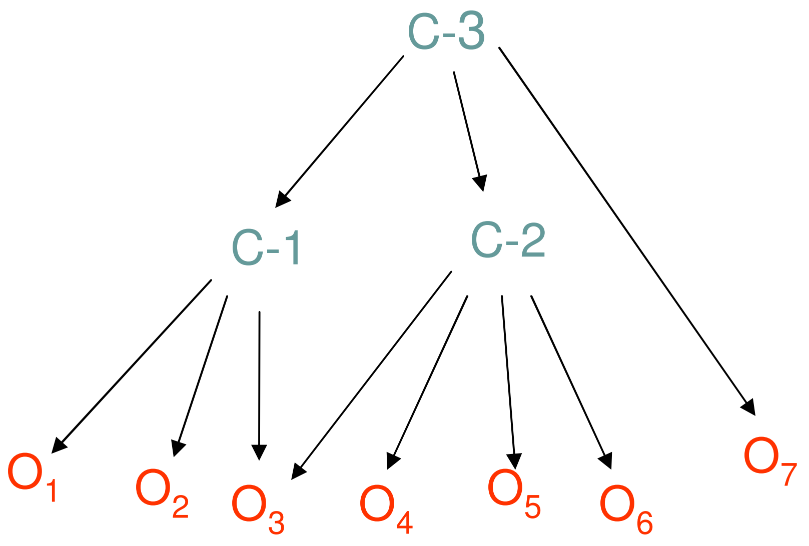
over-approximates

Resolution Graph

Binary DAG with intermediate and conflict clauses.



Collapsed DAG with multi-degree nodes



Each node in the graph is an abstraction of its descendants

Refinement of models and formulas

- An intermediate model $\widehat{M}$ is a refinement of $\widehat{\widehat{M}}$ if:

$$\begin{aligned}\forall a. \quad & a \models M \rightarrow \alpha \models \widehat{M} \wedge \\ & a \models \widehat{M} \rightarrow \alpha \models \widehat{\widehat{M}}\end{aligned}$$

- An intermediate formula $\widehat{F}$ is a refinement of $\widehat{\widehat{F}}$ if:

$$\begin{aligned}\forall a. \quad & a \models F \rightarrow \alpha \models \widehat{F} \wedge \\ & a \models \widehat{F} \rightarrow \alpha \models \widehat{\widehat{F}}\end{aligned}$$

or simply: $F \rightarrow \widehat{F}\,,\, \widehat{F} \rightarrow \widehat{\widehat{F}}$

Why all this theory? …

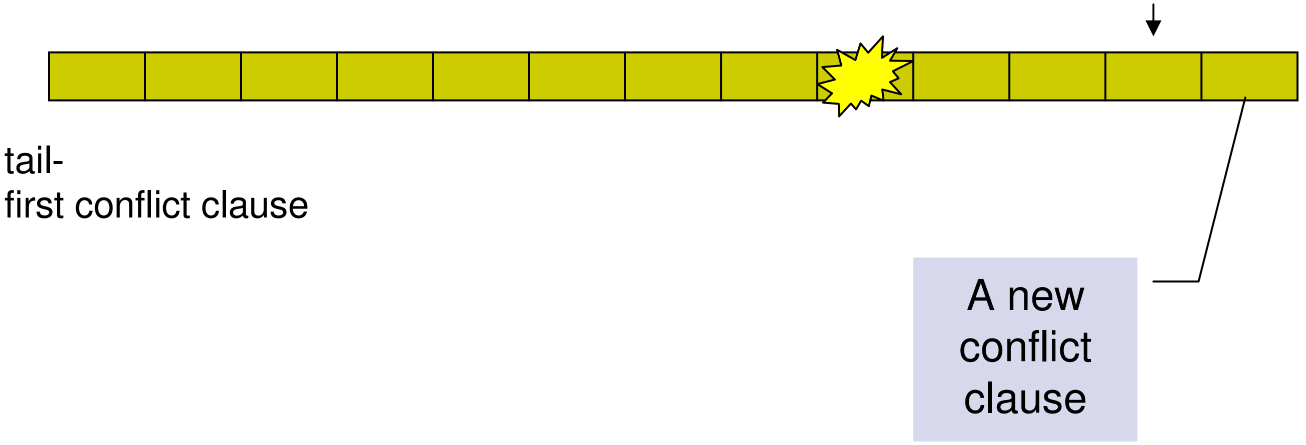
- Because Conflict Clauses are derived through a process of resolution.
- Several modern Decision Heuristics are guided by the Conflict Clauses (e.g. Berkmin)

- Hence, we can analyze them with the Abstraction/Refinement model.

Berkmin’s heuristic

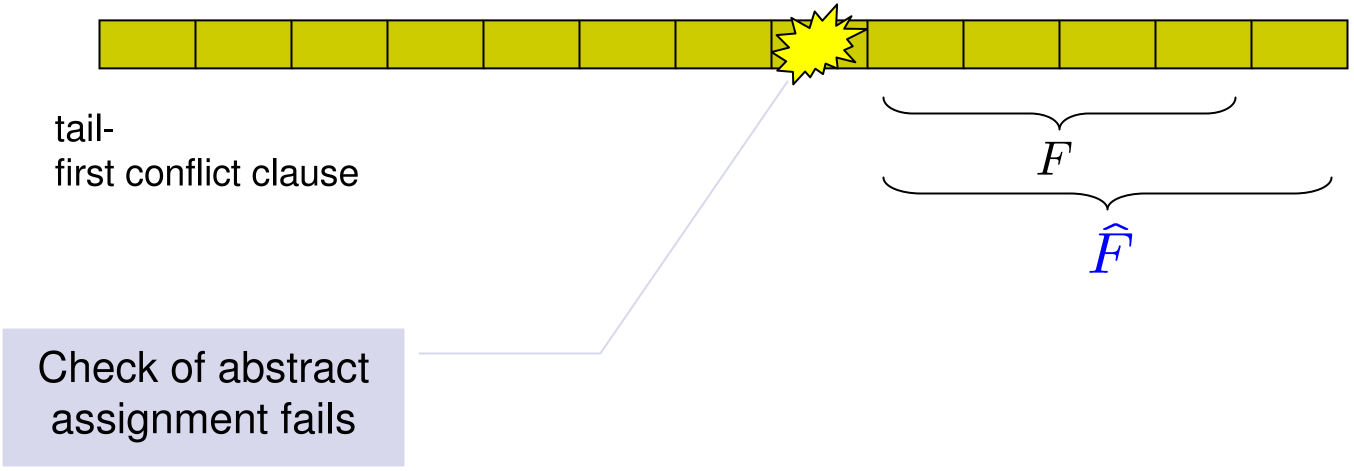
- Push conflict clauses to a stack.
- Find the first unsatisfied clause and choose a variable from this clause.
- If all conflict clauses are satisfied, choose a variable according to the VSIDS (Zchaff) heuristic.

Berkmin heuristic



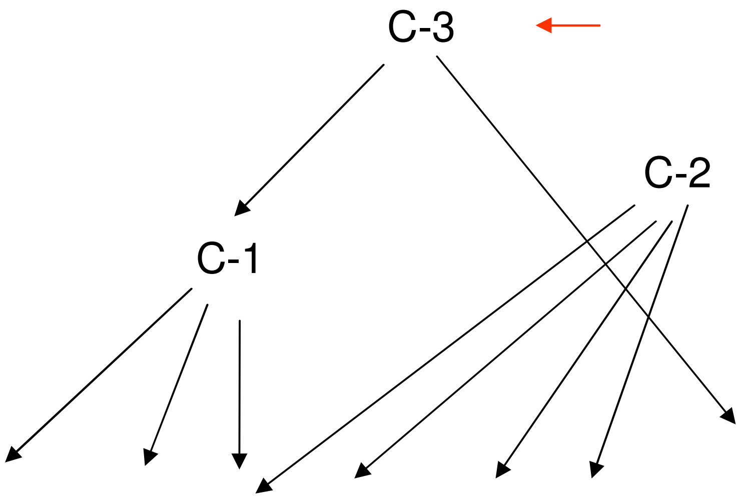
Berkmin heuristic

- Let φ denote the original formula
- F abstracts φ $(\varphi \rightarrow F)$
- $\widehat{F}$ is a refinement of F with respect to φ $(\varphi \rightarrow \widehat{F},\ \widehat{F} \rightarrow F)$



Berkmin heuristic

- Does not focus on a specific Abstraction/Refinement path.



- Generally: hundreds of clauses can be between a clause and its resolving clauses.

Progressing on the resolve graph

- Progress with “Best-First” according to some criterion.
- Must store the whole resolve graph in memory – this is frequently infeasible.
- HaifaSat’s strategy:
  - Do not store graph
  - Be more abstraction-focused than Berkmin

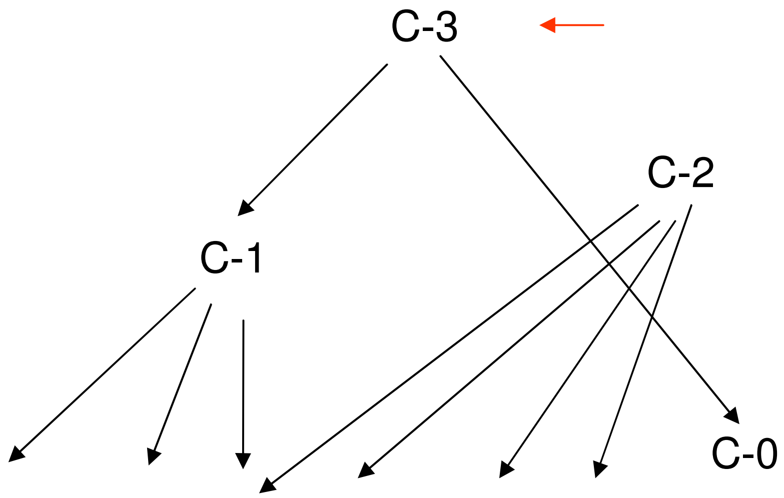
The CMTF heuristic

- Position conflict clauses together with their resolving clauses in the end of a list.

- Find the first unsatisfied clause and choose a variable from this clause.
- If all conflict clauses are satisfied, choose a variable according to the VMTF (Siege) heuristic.

Gives us the 'first-layer approximation' of the graph.

CMTF

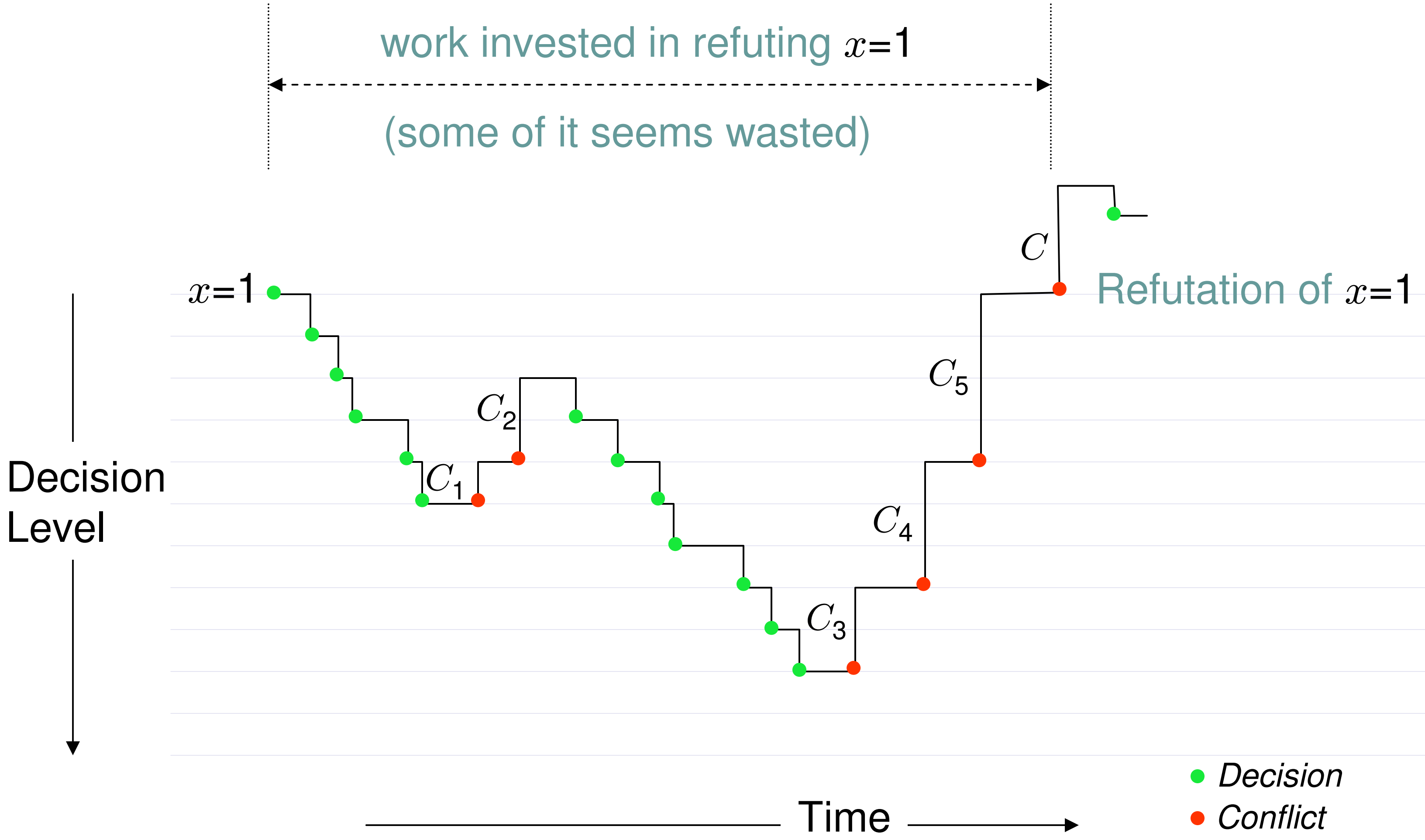


- When C-3 is created, C-0, C-1 are moved to the head of the list together with C-3.
- C-2 is left in place.

Given a clause: choose a variable.

- The Activity of a variable v:
  - Activity score of a variable increases when it is a resolution variable, but…
  - only when the clause it helped resolving is currently relevant, and…
  - it happened recently
- A recursive computation embedded in the First-UIP scheme.

Activity Score

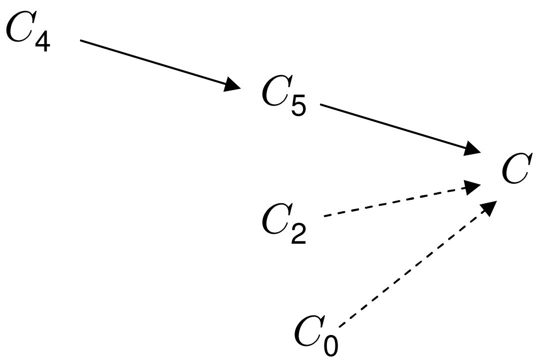


work invested in refuting x=1

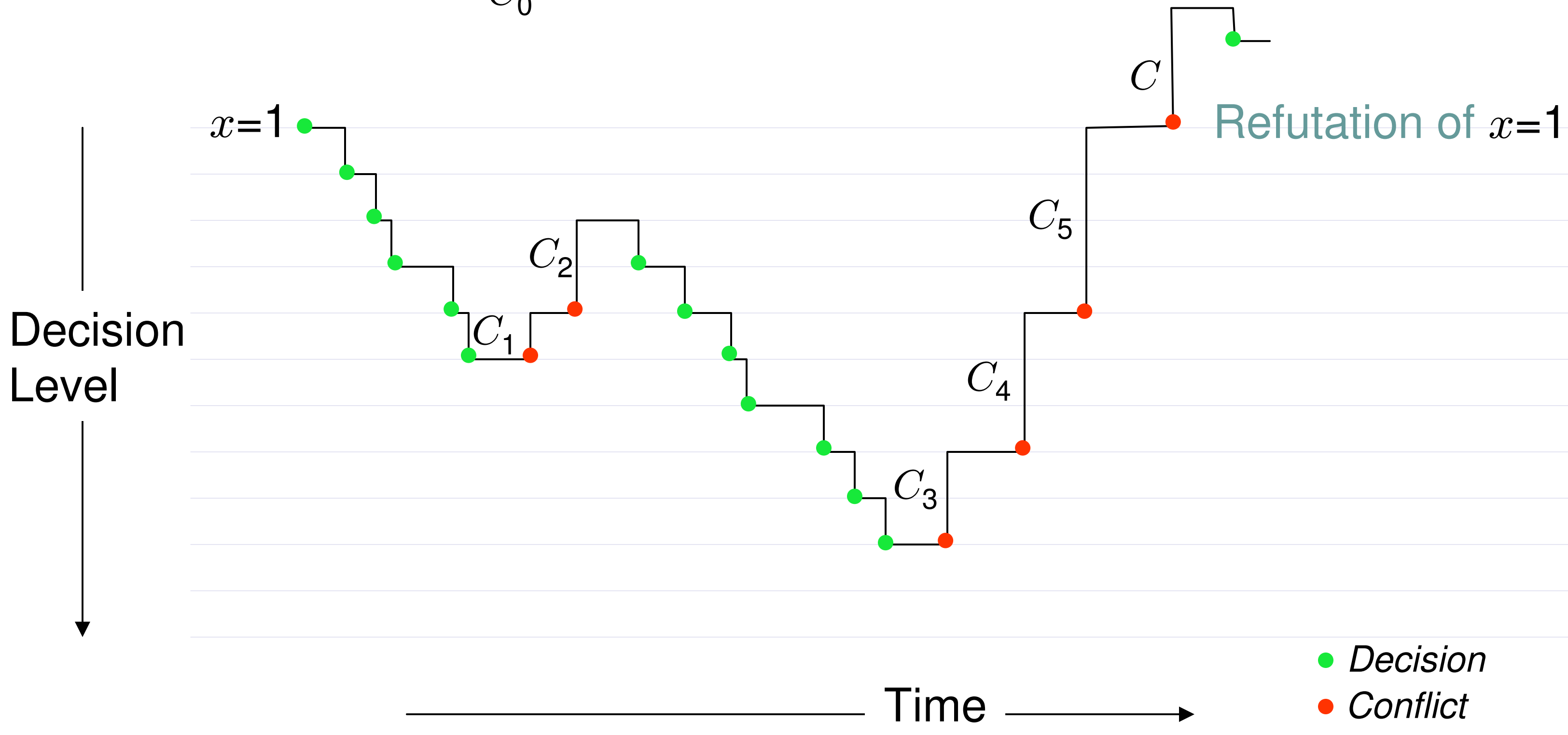
(some of it seems wasted)

Refutation of x=1

Activity Score



Weight is given to variables resolved-on in the process of resolving C



Refutation of x=1

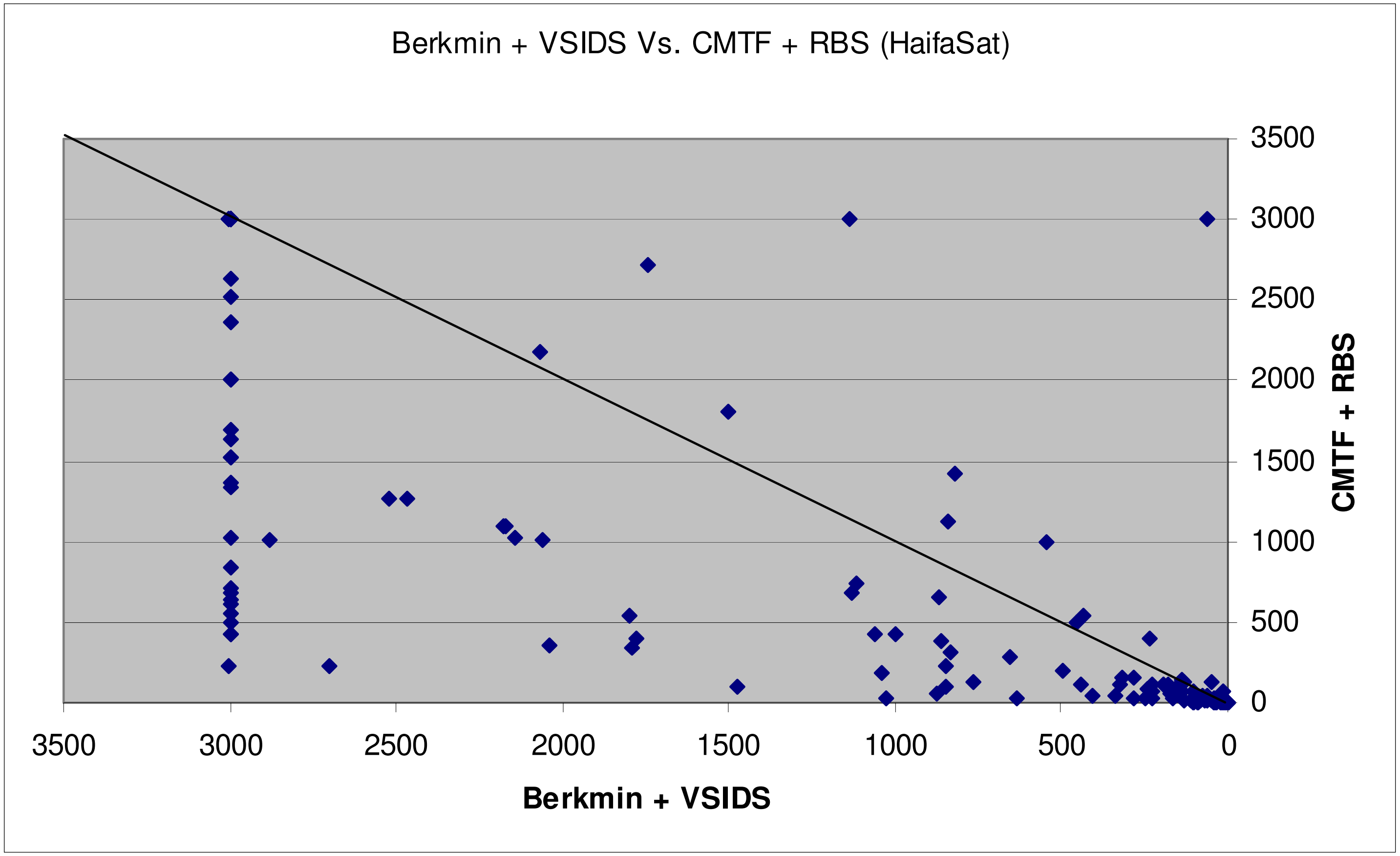
Results (sec., average)

Technion

Benchmark (#)	**Berkmin+VSIDS**	**CMTF+RBS**
Hanoi (5)	530	130
IP (4)	395	203
Hanoi03 (4)	1342	426
Check-int (4)	3323	681
Bmc2 (6)	1030	1261
Fifo8 (4)	3944	1832
Fvp2 (22)	8638	1995
W08 (3)	5347	2680
Ibm02 (9)	9710	3875
01_rule (20)	33642	19171
11_rule_2 (20)	34006	22974

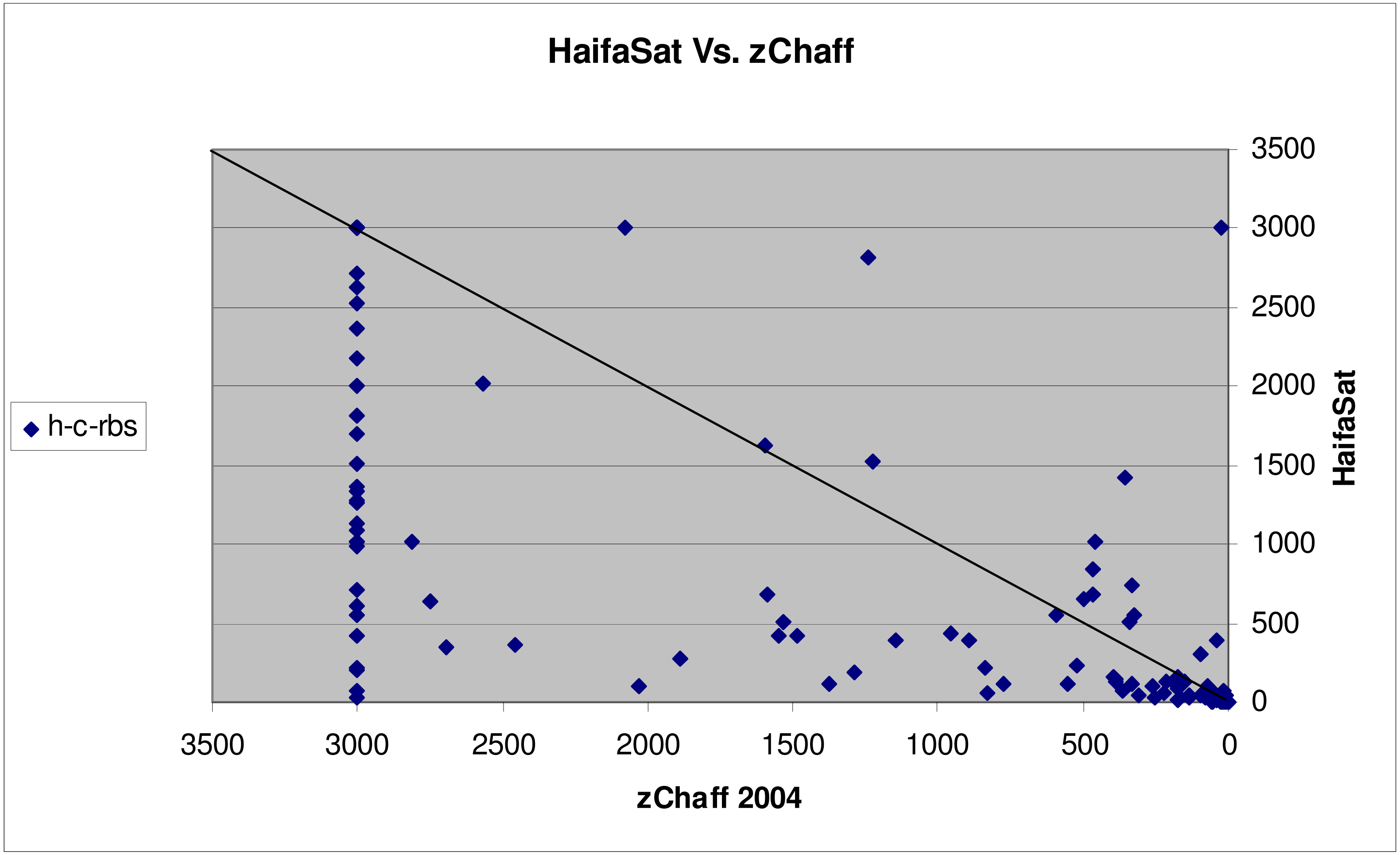
(CMTF + RBS) Vs. Berkmin (both implemented inside HaifaSat)

Technion

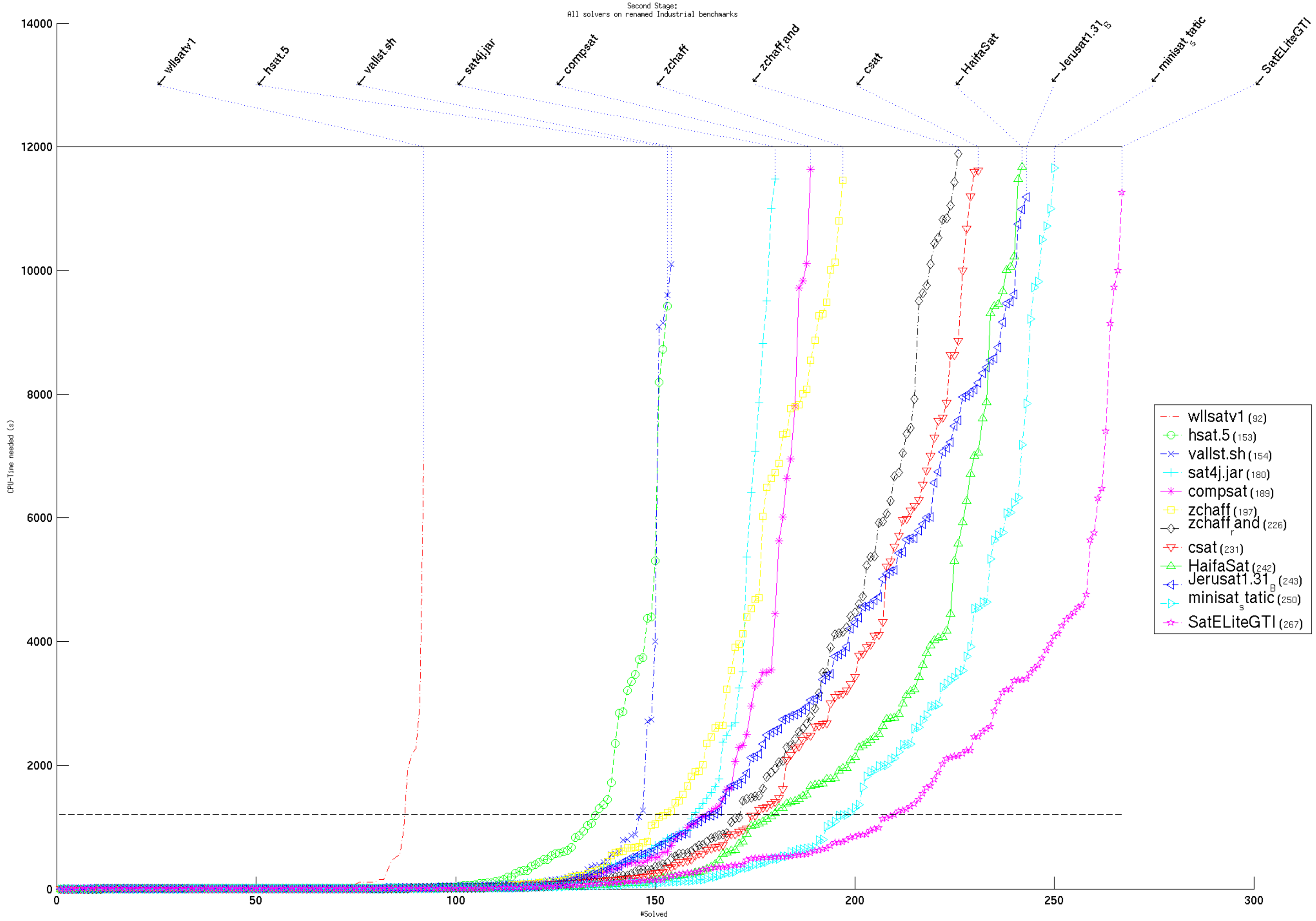


HaifaSat Vs. zChaff 2004

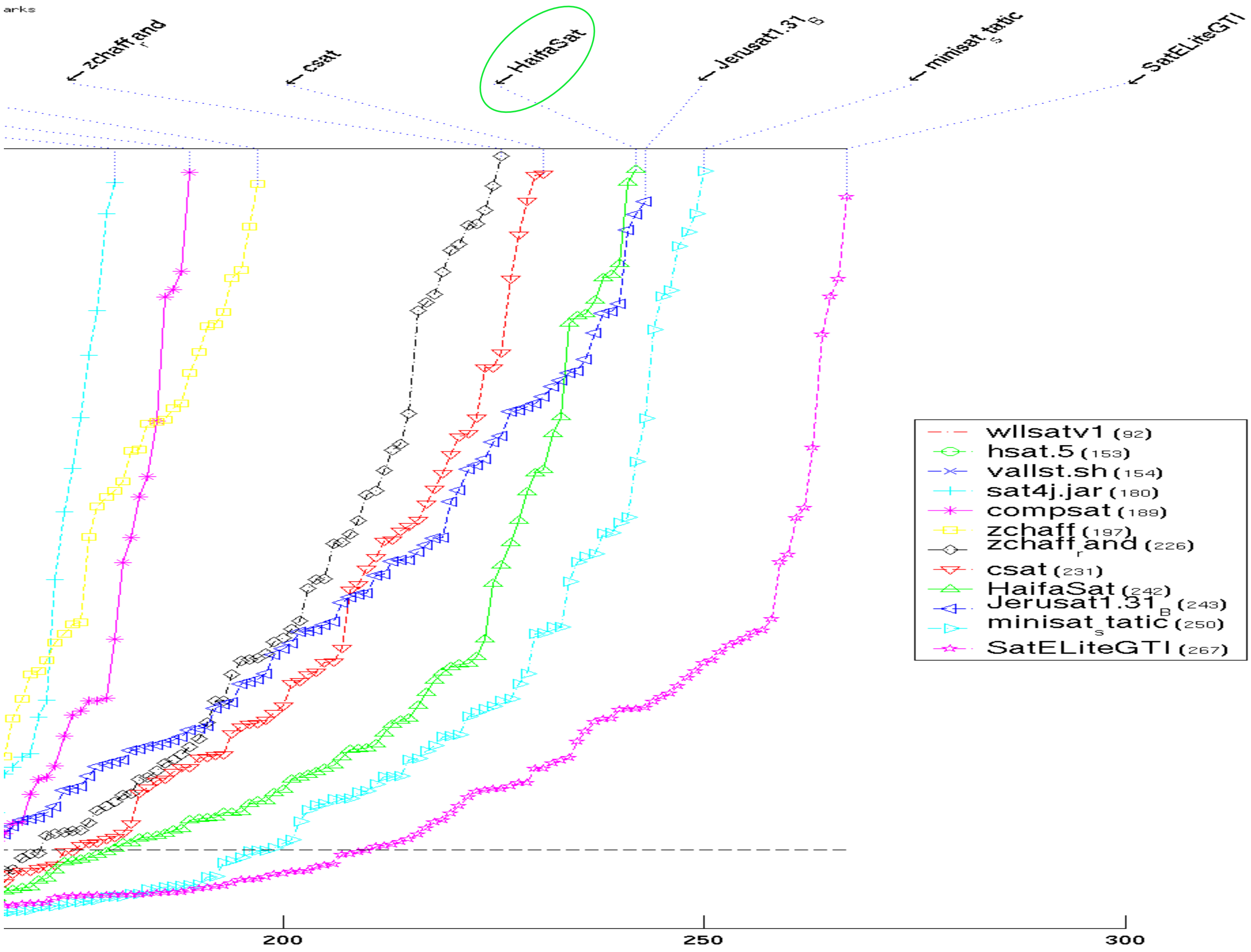
Technion



Results –SAT05 (Industrial)



Results –SAT05 (Industrial)



General Heuristic

1. Mark all roots.
2. Choose an unresolved marked clause V

   (If there are none - exit)
3. Decide a variable from V until it is satisfied.
4. Mark V's children

The Clause-Move-To-Front (CMTF) heuristic

- Is an instantiation of the general heuristic
- Does not need to store the whole graph.
- More focused than Berkmin.