

INTRODUCTION

- Conflict-Driven Clause Learning:

- are believed to be well-known:

- * They embed dynamic heuristics [6, 3], learning [7], restarts [4, 1] and lazy data structures [6].
 - * (efficient) SAT Solvers can be written from scratch in less than a thousand lines of code.
- #### - are not so well-known:
- * Who really understand the underlying mechanisms?
 - * No real experimental studies: Progresses have been made with extensive tests "only"
 - * What is a good learnt clause? When to forget? When to restart?

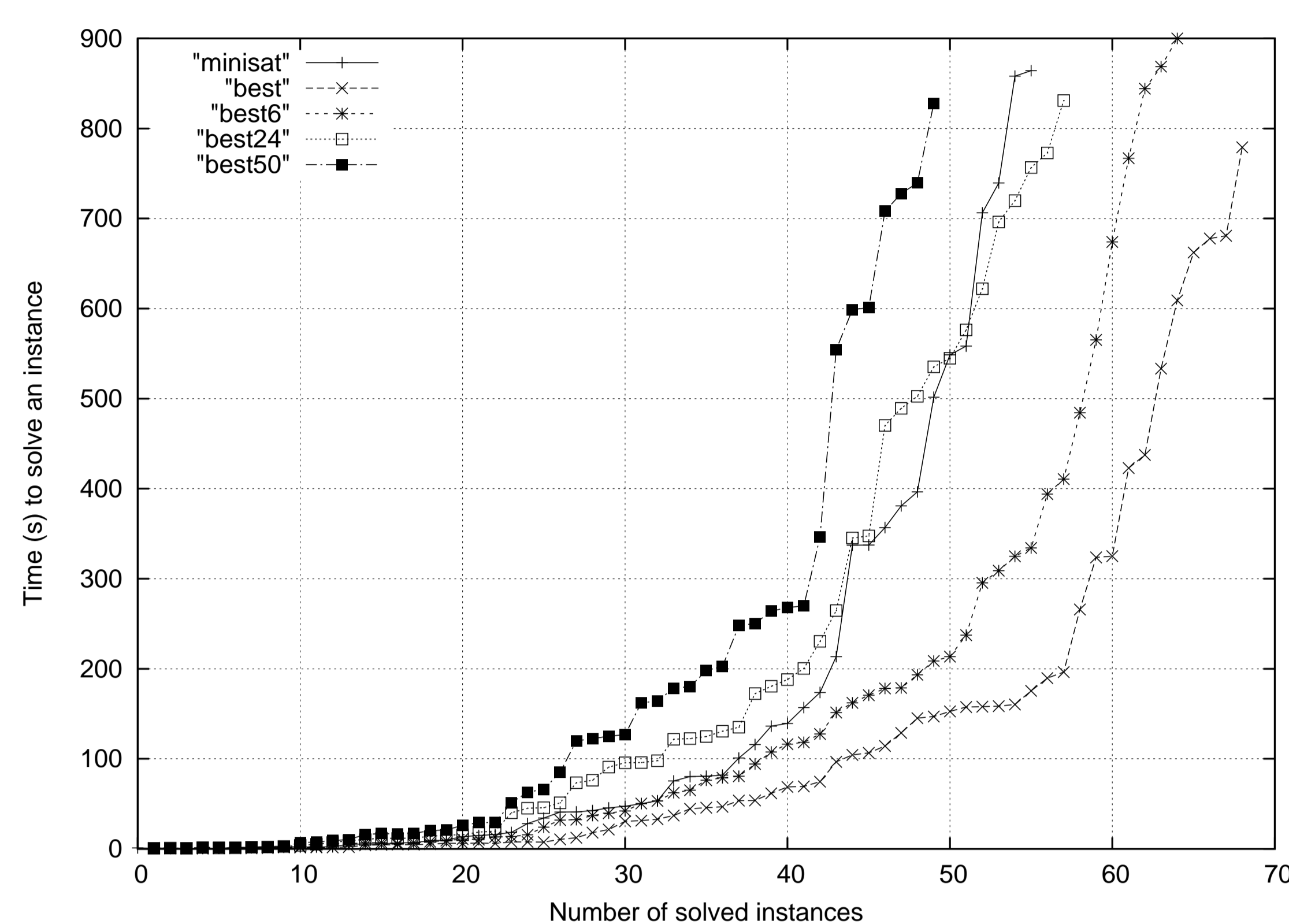
- Crucial for future progresses:

- In-depth study rather than immediate improvements
- Understanding each component of any CDCL solvers

- Questions addressed here:

- How small changes in Minisat [2] change results?
- Impact of data structures v.s. "good" ideas

LISA SYNDROME REVISITED



- Principle

- Shuffling instances is admitted to be "bad" for industrial benchmarks. Is it really true?? How much it is bad?
- We shuffled instances [5] and performed 50 runs by instance

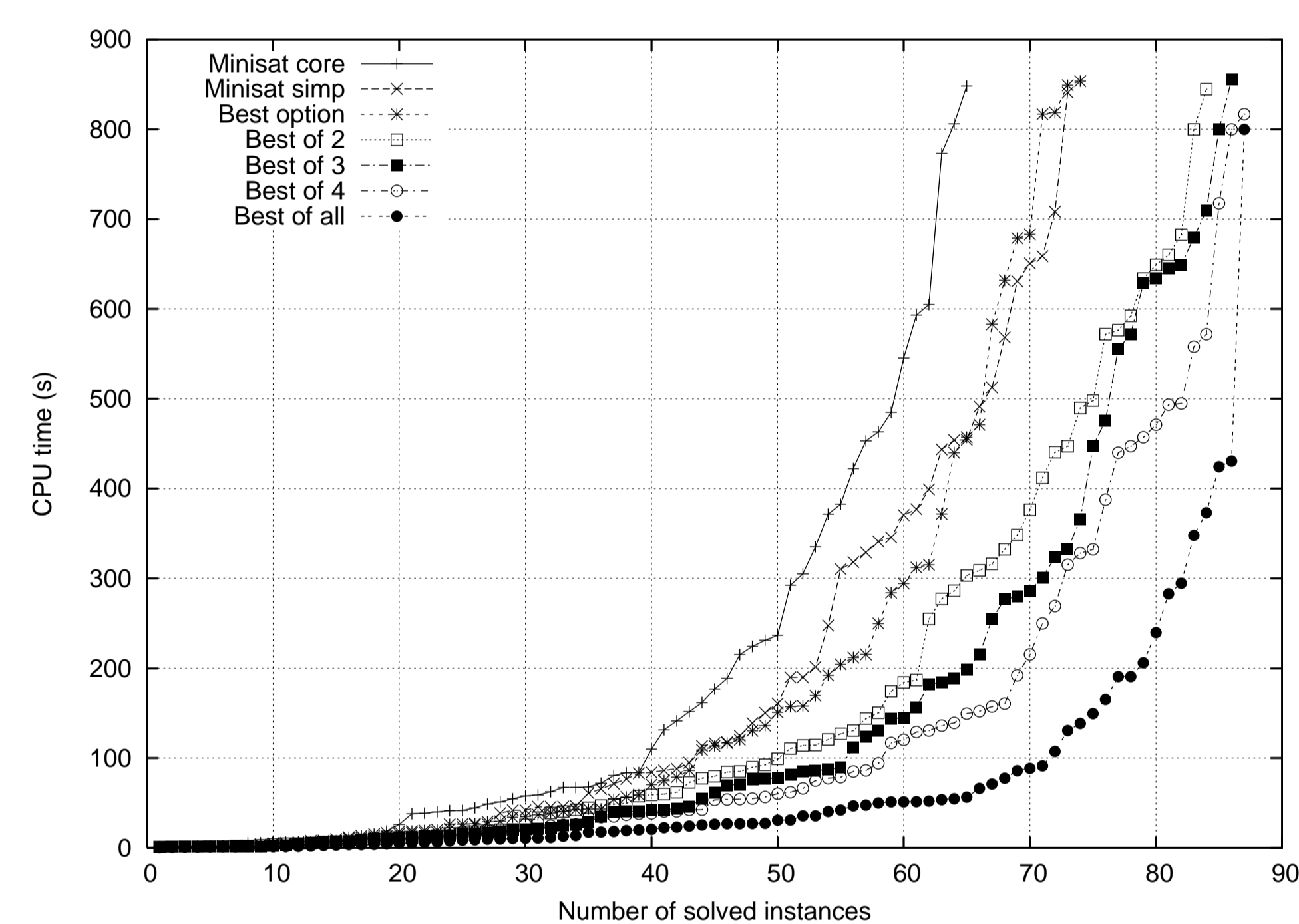
- Observations

- Curves "best" (resp. "best_i") plot the result of virtual solvers which would have the best CPU time on all shuffled runs (resp. the *i*th percentile)
- MINISAT solves 55 problems, "best" 69.
- 25% of chances to obtain better results by shuffling instances!

- Conclusion

- Shuffling instance may improve results. Many shuffling shows that MINISAT on original instance is not the best run. Improvements may be obtained here.
- Shuffling is not so bad. If it would have been so crucial to keep original benchmarks, we would have expected worst results.

PARAMETERS EFFECTS



- Principle

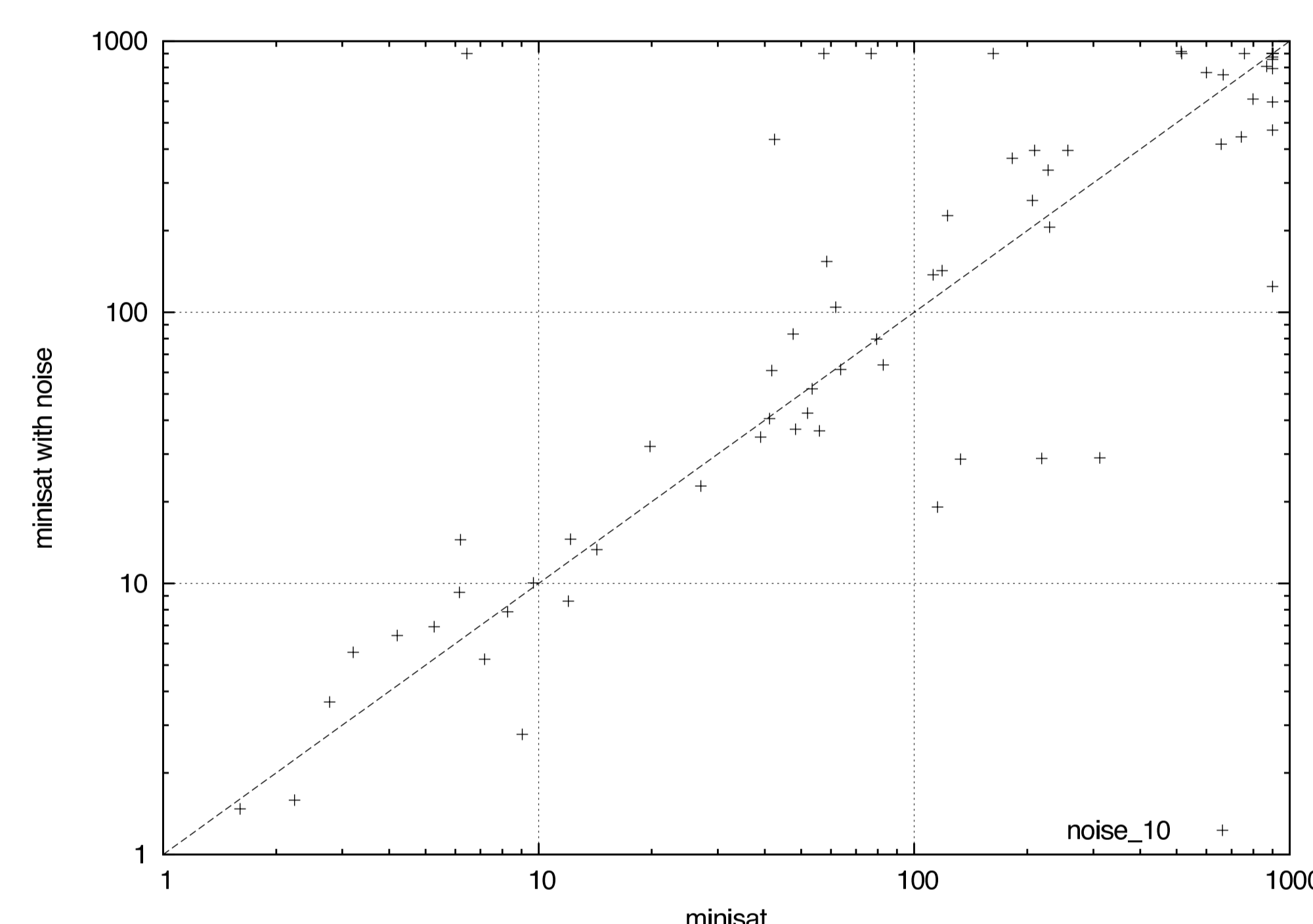
- Solvers need parameters. How to select them? Is changing parameters as strong as changing solvers?
- We took 10 magic values of MINISAT and built 126 solvers, each 1-parameter away from the original one.

- Observations

- Released MINISAT is well-tuned. Taking 2 versions improves performances, but more than 3 don't.
- Satellite is not always important (best of 2 contains a core version with fast restarts).
- Is MINISAT hard limit 74 or 85? What performances are awaited for a "better" solver than MINISAT?

- Conclusion

- High discrepancy is observed when changing restart policies. Fast restart is efficient with MINISAT core.
- Small changes can really change solvers. Ideas for multicore processors?



- Principle

- When new solvers are "better" than older ones? New ideas may hide noise effects on parameters!
- Side-effect over parameters are simulated by adding only 10% noise each time a constant is requested.

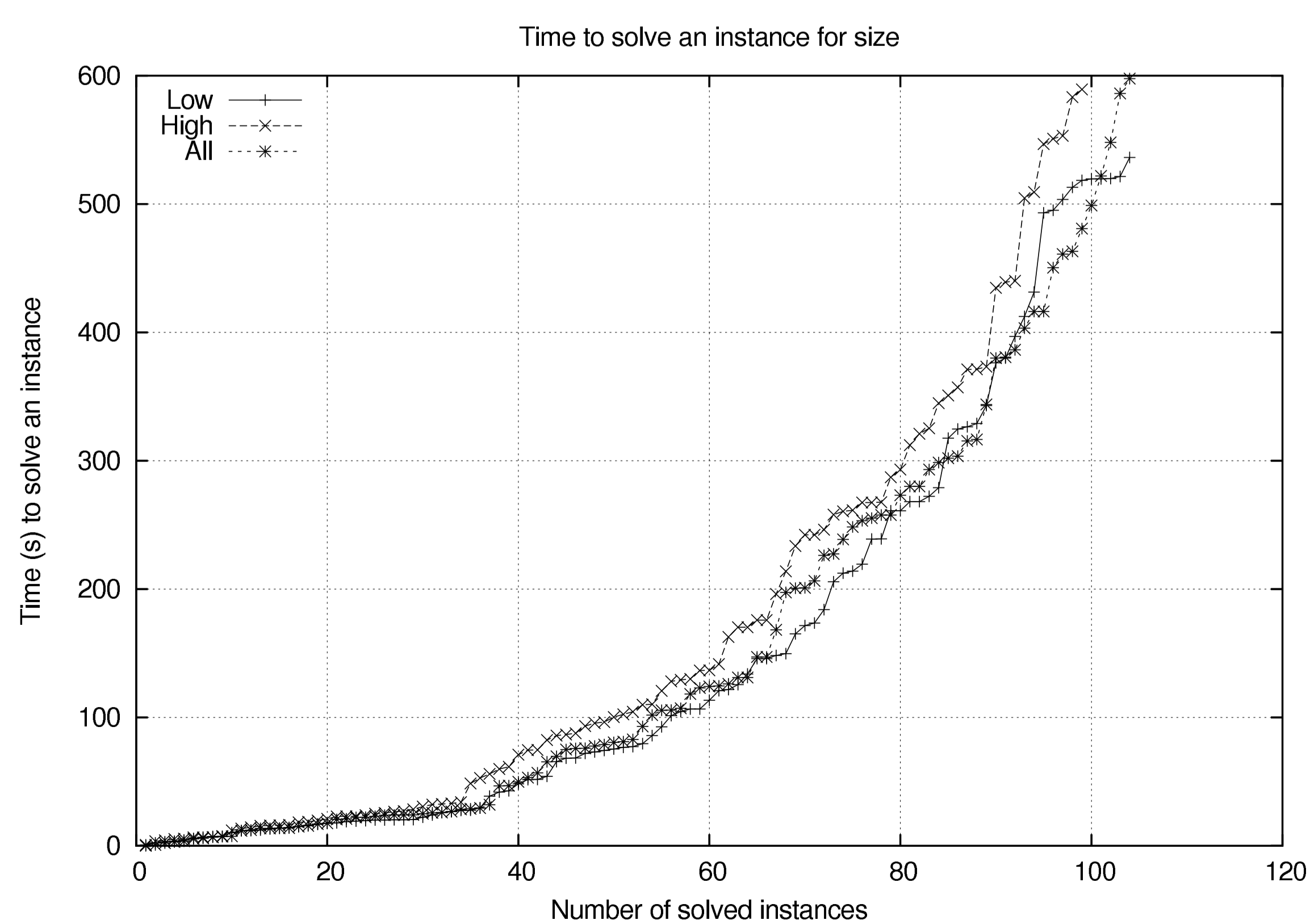
- Observations

- Noisy MINISAT behaves like a real different solver. Hard to say if one is better than another.

- Conclusion

- When observing this kind of plot, nothing must be drawn from it, relatively to new ideas. This figure may only report a noisy side-effect of any data-structure.

IDENTIFYING GOOD LEARNT CLAUSES ?



- Principle

- CDCL solvers are very memory consuming. Bad clauses must be forgotten.
- Is it possible to detect "good" clauses? It is well admitted that short clauses are better. Is this simple assumption already true?
- We deleted at random 50% of short (resp. large) clauses and compare it with deleting 25% of clauses.

- Observations

- No differences are observed! Deleting short or large clauses at random is the same.

- Conclusion

- Large clauses are important for UNSAT proofs (already known).
- Large clauses may have a local - but important - impact on the proof, when it is reduced in the search tree.
- The simple assumption that shorter clauses are better is false. We need a way to identify good clauses, and experimentally validate this.

CONCLUSION

- Experimenting?

- Experimentation is the traditional companion section of papers. Competition are organized [5] to "rank" solvers.
- This is more "testing" than "experimenting".

- Poster Lessons

- A lot of work is still pending to really understand the behavior of CDCL solvers.
- Not easy to prove that a solver is "better" than another. What is "better"? Hard question, even on a fixed set of benchmarks.
- We need to build a real experimental science of CDCL solvers. This is possible, the SAT area is one of the pioneers in this direction.

References

- [1] A. Biere. Adaptive restart strategies for conflict driven SAT solvers. *SAT'08*.
- [2] N. Een and N. Sorensson. An extensible sat-solver. *SAT'03*.
- [3] E. Goldberg and Y. Novikov. BerkMin: A fast and robust SAT-solver. *DAM'07*.
- [4] J. Huang. The effect of restarts on the efficiency of clause learning. *IJCAI'07*.
- [5] D. Le Berre and L. Simon. Essentials of the SAT'03 competition. *SAT'03*.
- [6] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient SAT solver. *DAC'01*.
- [7] J. Marques-Silva and K. Sakallah. Grasp - a new search algorithm for satisfiability. *ICCAD'96*.