On the Hierarchical Community Structure of Practical Boolean Formulas

Chunxiao Li, Jonathan Chung, Soham Mukherjee, Marc Vinyals, Noah Fleming, Antonina Kolokolova, Alice Mu, and Vijay Ganesh

Why are CDCL SAT Solvers Efficient?

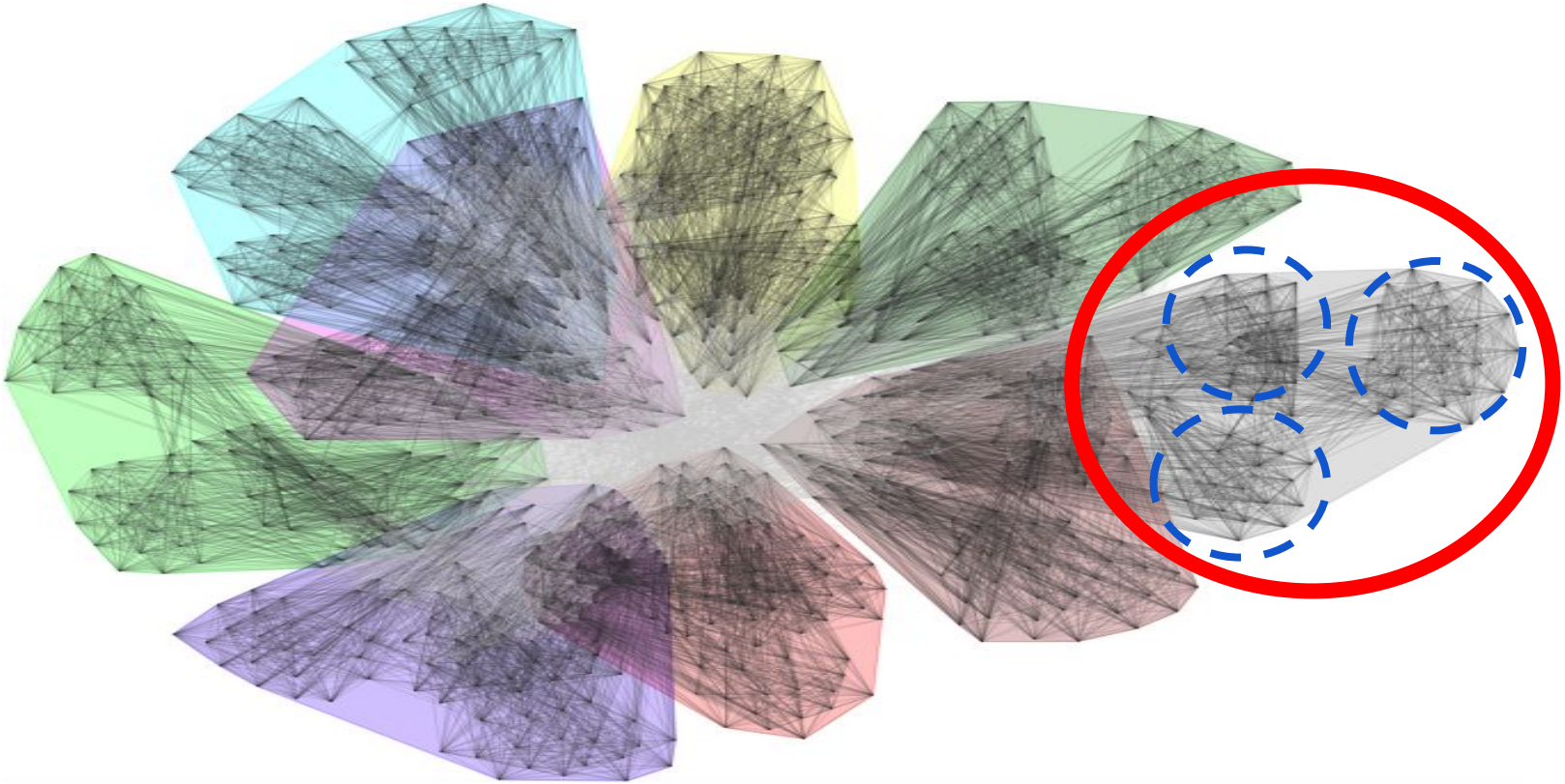
Problem Statement:

- SAT solvers can solve large verification instances with millions of variables/clauses in them, despite the fact that the SAT problem is NP-complete

- **Bridging this gap between theory and practice is one of the central research directions in SAT solver research**
  - Via structure of industrial instances and contrasting with random/crypto/crafted
  - This problem has proven to be very challenging. Proposed theoretical parameters don't seem to work in practice (e.g., tree-width) and vice-versa (e.g., community structure)

- **Goal: Find parameters that both**
  - Explain empirical success of SAT solvers over industrial instances
  - Enable us to prove complexity-theoretic upper bounds on proof size and proof search

Hierarchical community structure (HCS)



	Meaning of Result	Result
Empirical	- HCS cleanly differentiates between verification and random instances	- 99% accuracy over 10869 instances from SAT competition and other benchmarks. - Verification instances have “good” HCS parameters and random instances don’t
	- HCS predicts solver runtime	- Empirical hardness model: R^2 of 0.83
	- When we scale HCS, we see an expected scaling in solver runtime	- Hardness of instances scales with HCS parameter values
Theoretical	- The better the HCS parameters, the larger the class of expanders that are ruled out	- As the values of the HCS parameters become worse, the size of the largest embeddable expanders increases
	- The parameters are “necessary”	- If even one HCS parameter is “bad”, can construct a hard formula
Future Work	- Proving parameterized upper bounds with “good” HCS parameters and other constraints	