IMPROVING SAT SOLVER WITH GRAPH NETWORKS AND RL

NVIDIA

UNIVERSITY OF OXFORD

AIMS EPSRC Centre for Doctoral Training in Autonomous Intelligent Machines & Systems

Vitaly Kurin, Saad Godil, Shimon Whiteson, Bryan Catanzaro

https://arxiv.org/abs/1909.11830

CAN RL IMPROVE AN EXISTING SAT SOLVER?

BOOLEAN SATISFIABILITY (SAT) PROBLEM

$$(x_1 \ OR \ x_2) \ AND \ (NOT \ x_2 \ OR \ x_3)$$

SAT IS IMPORTANT

- Theoretical computer science;
- Automatic theorem proving;
- Circuit design;

SOLVERS RELY ON HEURISTICS METICULOUSLY CRAFTED BY HUMANS

WHAT DO WE HAVE NOW?

- Graph-Q-SAT (GQSAT), a branching heuristic
- >2x iteration speed-up on random 3-SAT problems
- Generalization to problems 5x in size
- SAT -> unSAT

HOW DID WE ACHIEVE THAT?

- Injecting a model into an existing algorithm
- Graph Representation
- Graph Neural Networks
- Reinforcement Learning (DQN)

CDCL

```
def CDCL(formula):
    if trivially_satisfiable(formula):
        return True
    if trivially_unsatisfiable(formula):
        return False

    literal, value = pick_literal(formula)
    formula = propagate(formula, literal, value)
    return CDCL(formula)
```

Injecting a model into an existing algorithm

CONFLICT LEARNING

X_1 AND x_2 AND x_3 => unSAT?

Add `NOT (x_1 AND x_2 AND x_3)` to clauses

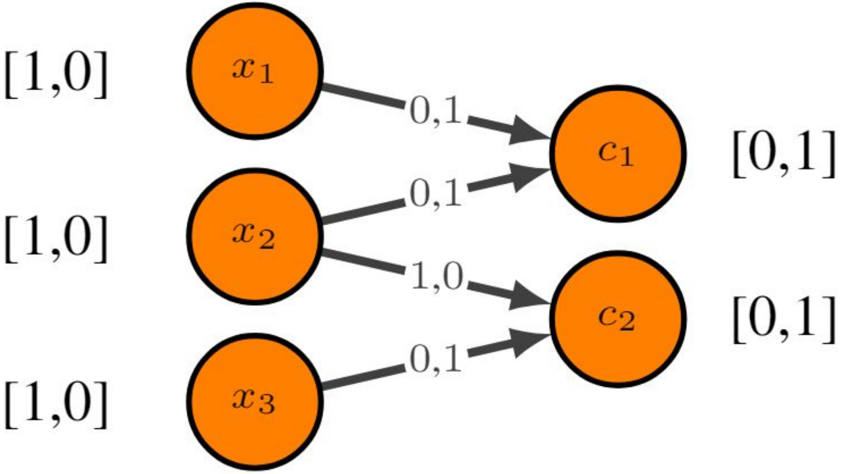
Injecting a model into an existing algorithm

VSIDS

x_1	x_2	x_3		x_1	x_2	x_3
0	0	0	⇨	4.2	3.1	2.7

$$(x_1\ OR\ x_2)\ AND\ (NOT\ x_2\ OR\ x_3)$$

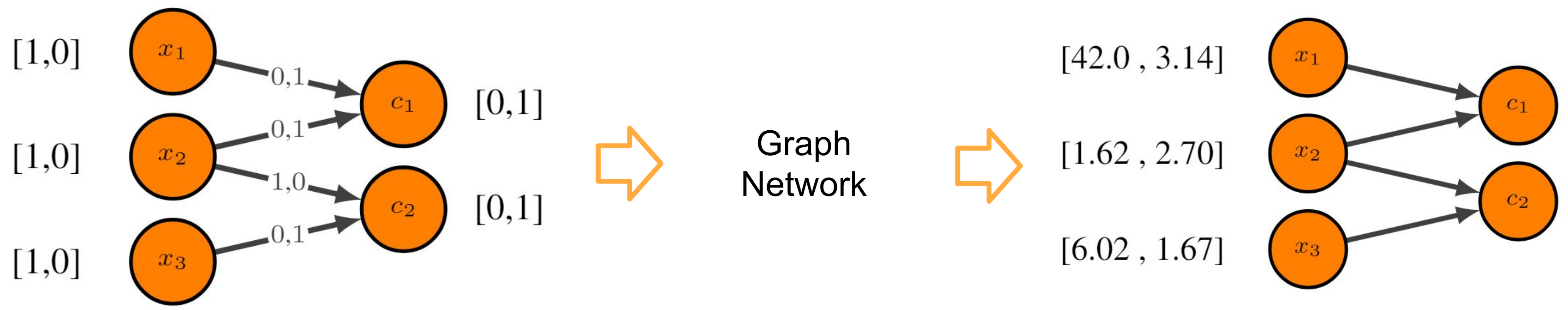
SAT AS A GRAPH



$$(x_1\ OR\ x_2)\ AND\ (NOT\ x_2\ OR\ x_3)$$

Graph Representation

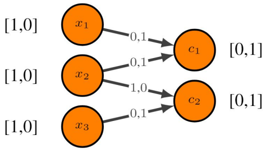
GRAPH NEURAL NETWORK



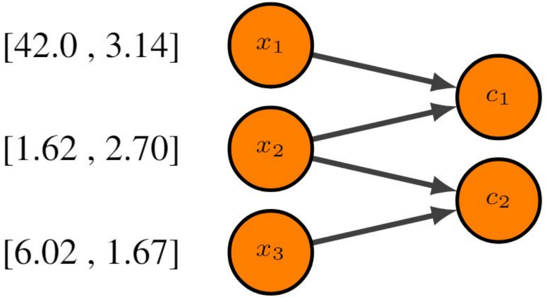
Graph Nets

DQN

Q(



) =



Reward is -0.1 for a non-terminal step.

TRAINING PIPELINE

- Train model on SAT 50-218 train data
- Evaluate every k-th epoch
- Pick the best
- Evaluate on the test set

METRIC OF SUCCESS

$$\left[\frac{Minisat\ Steps}{Our\ Steps}, \frac{Minisat\ Steps}{Our\ Steps}, \ldots, \frac{Minisat\ Steps}{Our\ Steps} \right]$$

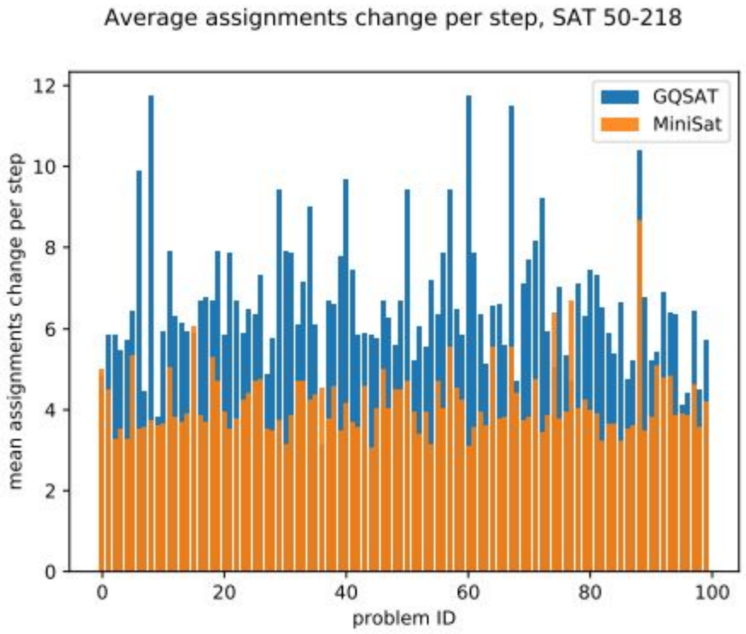
problem 1   problem 2   problem 100

PROBLEM SIZE/TYPE GENERALIZATION

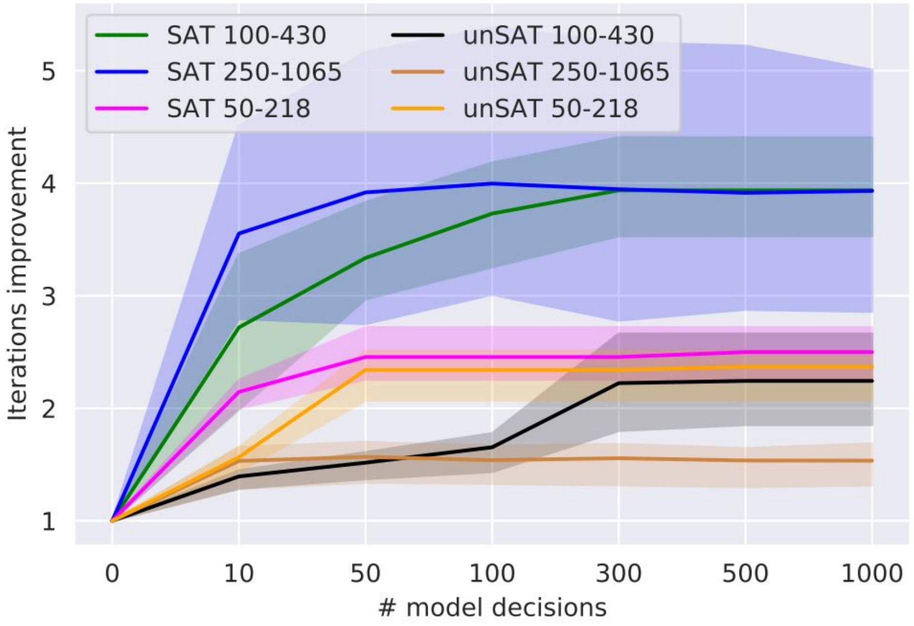
Table 2: MRIR for GQSAT trained on SAT-50-218. Evaluation for SAT-50-218 is on a separate test data not seen during training.

dataset	mean	min	max
SAT 50-218	2.46	2.26	2.72
SAT 100-430	3.94	3.53	4.41
SAT 250-1065	3.91	2.88	5.22
unSAT 50-128	2.34	2.07	2.51
unSAT 100-430	2.24	1.85	2.66
unSAT 250-1065	1.54	1.30	1.64

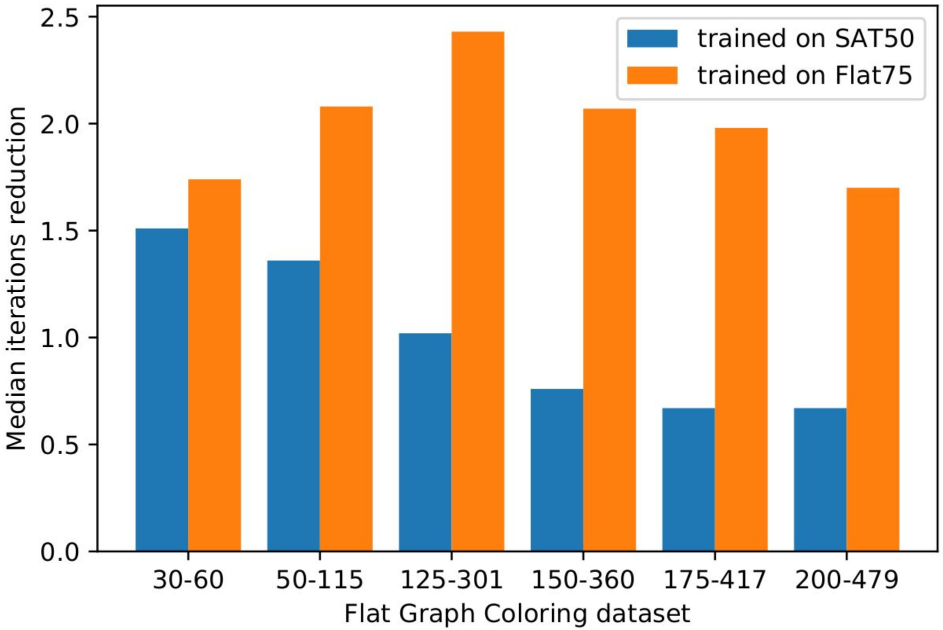
WHY IS GQSAT EFFICIENT?



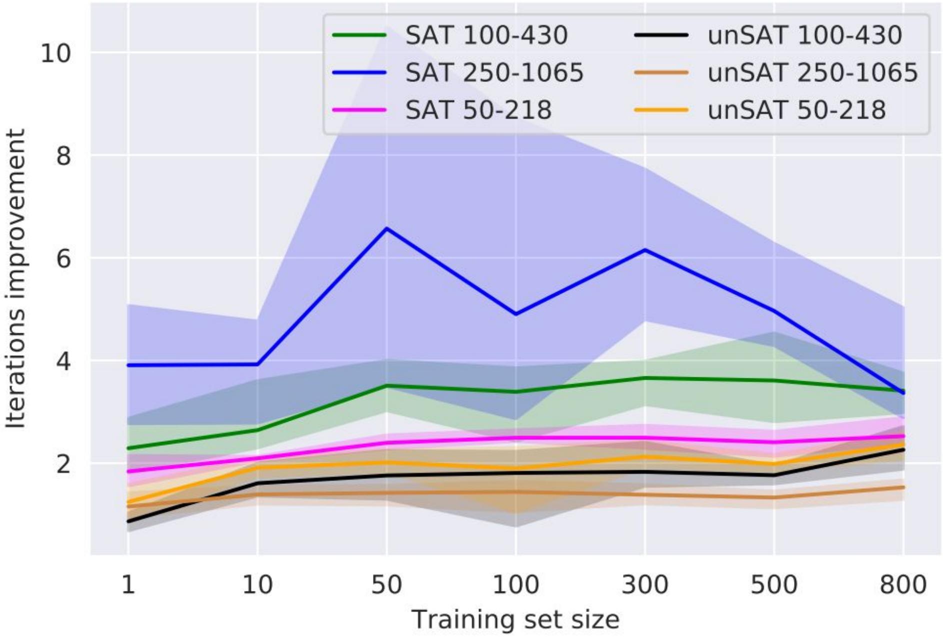
WARMING UP THE EXISTING ALGORITHM



PROBLEM STRUCTURE GENERALIZATION



DATA EFFICIENCY



FURTHER WORK

- Training on problems with larger horizon.
- Scaling to larger problems.
- From reducing number of iterations to wallclock time speedup.

ACKNOWLEDGEMENTS

- ***Bryan Catanzaro*** for inviting me, giving a problem to solve and constant feedback.
- ***Saad Godil*** for endless everyday discussions, feedback, coding and experiments.
- ***Andrew Tao*** and ***Guy Peled*** for computing support.
- ***Rajarshi Roy****,* ***Robert Kirby****,* ***Yogesh Mahajan****,* ***Alex Aiken****,* ***Mohammad Shoeybi****,* ***Rafael Valle****,* ***Sungwon Kim, Ryan Prenger*** and the rest of the **ADLR** team @NVIDIA for great discussion and being amazing people.
- ***University team*** and ***ERC*** **@**NVIDIA for making this a great experience.
- ***Shimon Whiteson*** @Oxford for supervising me.
- ***Samsung R&D UK*** for funding my DPhil.
- ***Wendy Poole*** **@**Oxford for organizing everything related to CDT.

@y0b1byte vitaliykurin@gmail.com https://yobibyte.github.io