# TaSSAT: Transfer and Share SAT

Md Solimul Chowdhury, Cayden R. Codel, and **Marijn J.H. Heule**

Carnegie Mellon University

30th International Conference on Tools and Algorithms
for the Construction and Analysis of Systems

Luxembourg April 8, 2024

# Local Search and DDFW Overview

Dozens of local search algorithms for SAT

- On various problems much faster than CDCL
- Most algorithms use local flips (to be prob. complete)
- We studied weight transfer algorithms (with global flips)

# Local Search and DDFW Overview

Dozens of local search algorithms for SAT

- On various problems much faster than CDCL
- Most algorithms use local flips (to be prob. complete)
- We studied weight transfer algorithms (with global flips)

Arguably the best weight transfer algorithm is DDFW

- *Divide and Distribute Fixed Weights*
- Original solver by Ishtaiwi et al. (2005) was never released
- Tompkins reverse engineered the details for UBCSAT
- Various papers mention effectiveness of DDFW in UBCSAT

# Weight Transfer Heuristics

Key heuristic: transfer weight from neighboring clauses

- Clauses are neighboring if they share a literal
- Transfer weight from satisfied to falsified clauses
- Transfer from highest weight satisfied neighboring clause

# Weight Transfer Heuristics

Key heuristic: transfer weight from neighboring clauses

- Clauses are neighboring if they share a literal
- Transfer weight from satisfied to falsified clauses
- Transfer from highest weight satisfied neighboring clause

Divide and Distribute Fixed Weights (DDFW) heuristics

- Weight initialization $W(C) = w_0 = 8$
- Transfer weights if no weight-reducing variable to flip
- Transfer a weight of 1 if $W(C_{\text{satisfied}}) = w_0$
- Transfer a weight of 2 if $W(C_{\text{satisfied}}) > w_0$

# New Weight Transfer Heuristics

Divide and Distribute Fixed Weights (DDFW) heuristics

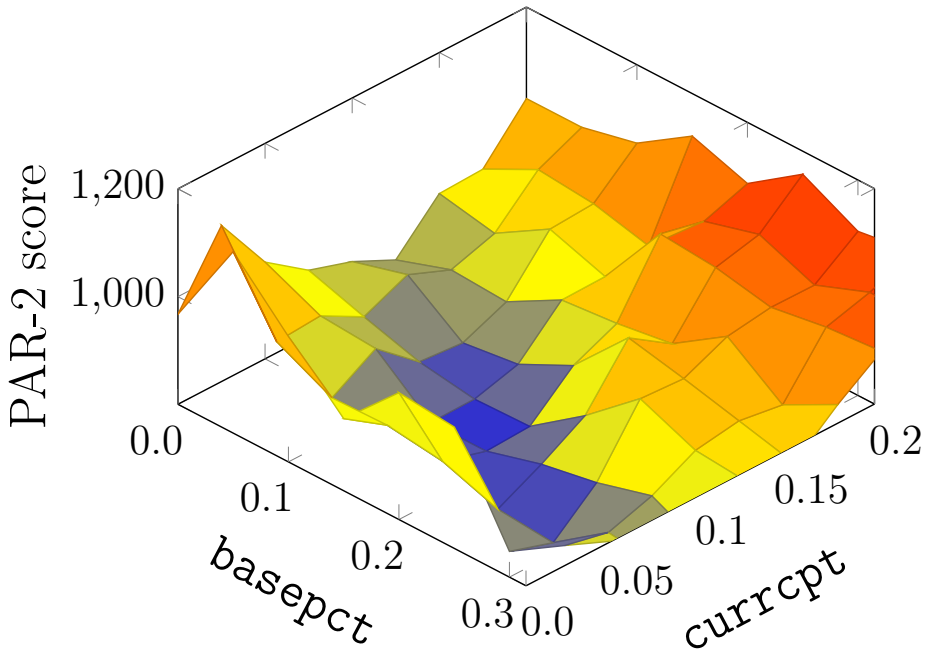
- Weight initialization $W(C) = w_0 = 8$ (int)
- Transfer weights if no weight-reducing variable to flip
- Transfer a weight of 1 if $W(C_{\text{satisfied}}) = w_0$
- Transfer a weight of 2 if $W(C_{\text{satisfied}}) > w_0$

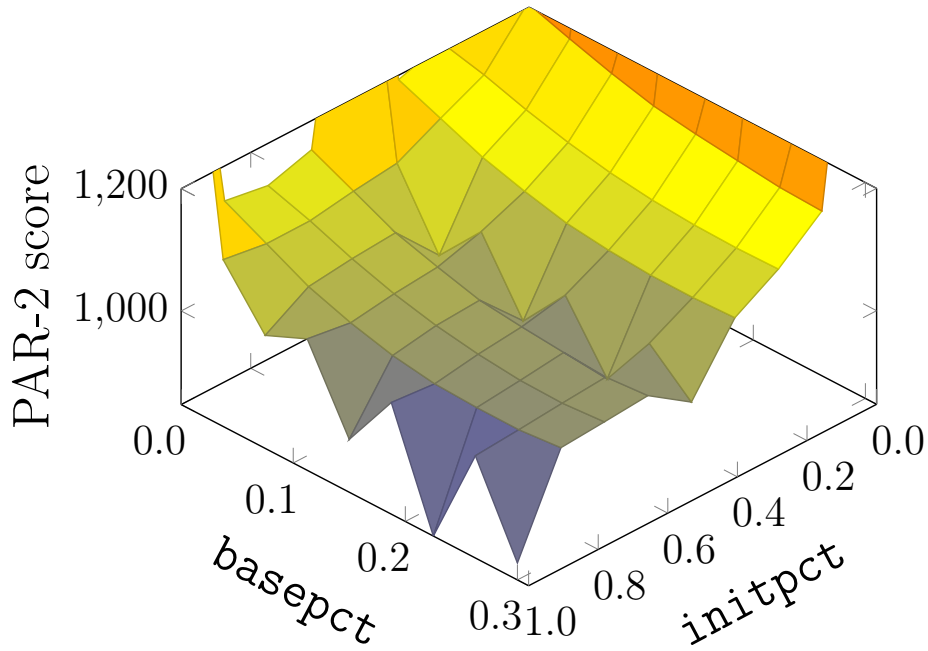
Linear Weight Transfer heuristics [NFM 2023]

- Weight initialization $W(C) = w_0$ (float)
- Transfer weights if no weight-reducing variable to flip
- Transfer a weight of $p_{\text{init}} \times w_0$ if $W(C_{\text{satisfied}}) = w_0$
- Otherwise a weight of $p_{\text{base}} \times w_0 + p_{\text{curr}} \times W(C_{\text{satisfied}})$

# Optimizing the Parameters

PAR-2: average runtime with timeout counted as 2× timeout




Observations:

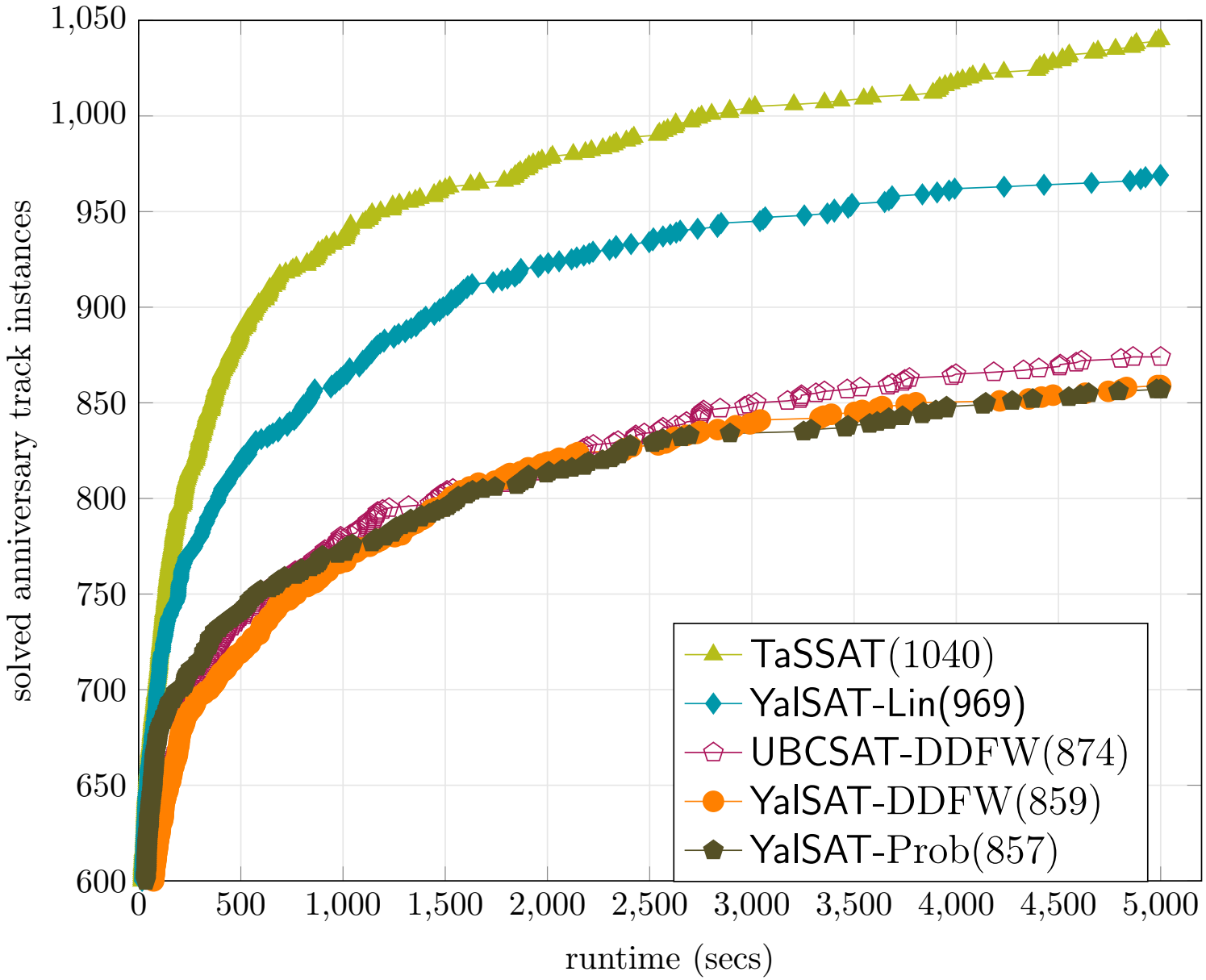
- Combining $p_{base}$ (`basepct`) and $p_{curr}$ (`currpct`) is best
- Max $p_{init}$ (`initpct`), i.e., taking all weight, is best

# Solver Comparison

Solvers used for runtime comparison

- TaSSAT: The solver presented in this talk/paper
- YalSAT-Lin: Weight transfer with NFM'23 paper heuristics
- YalSAT-DDFW: Weight transfer with DDFW heuristics
- YalSAT-ProbSAT: Default YalSAT
- UBCSAT-DDFW: Only public implementation of DDFW

# Results on SAT Competition Benchmarks



# Data-Structure Sharing

PalSAT:

- Each tread reads / stores / preprocesses formula
- Redundant computation
- Large memory footprint

PaSSAT:

- Master thread reads / stores / preprocesses formula
- Shared clause database and lookup table
- Large memory reduction when using many cores

# Results on van der Waerden Numbers

Color the numbers 1 to $n$ red and blue without

- arithmetic progress of length 3 in red
- arithmetic progress of length $k$ in blue

Best known results by Ahmed et al. using parallel SAT

- used DDFW in UBCSAT
- some bounds obtained by enforcing symmetries

| result \ k | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
|---|---|---|---|---|---|---|---|---|---|
| Known | 930 | 1006 | 1063 | 1143 | 1204 | 1257 | 1338 | 1378 | 1418 |
| PaSSAT | **953** | **1011** | **1071** | **1145** | **1208** | **1260** | **1341** | **1380** | **1419** |

# Conclusions

TaSSAT: Arguably the best SAT-based local search solver

- open source: https://github.com/solimul/tassat
- best SLS performance on SAT Competition benchmarks
- improved many van der Waerden lower bounds
- PaSSAT has reduced memory footprint

Future work

- Communication between threads (e.g. sharing assignments)
- Combining TaSSAT with CDCL
- Further improve heuristics