The Complexity of Finding Tangles

Oksana Firman[1] Philipp Kindermann[1] Alexander Wolff[1] Johannes Zink[1]

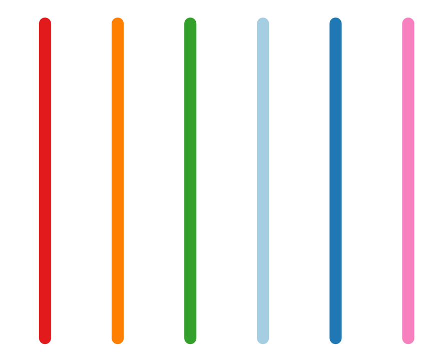
[1]Lehrstuhl für Informatik I, Universität Würzburg, Germany

Stefan Felsner[2] Alexander Ravsky[3]

[2]Institut für Mathematik, TU Berlin, Germany [3]IAPMM, National Academy of Sciences of Ukraine, Lviv

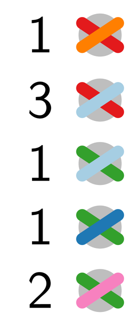
PROBLEM

Given a set of n y-monotone wires



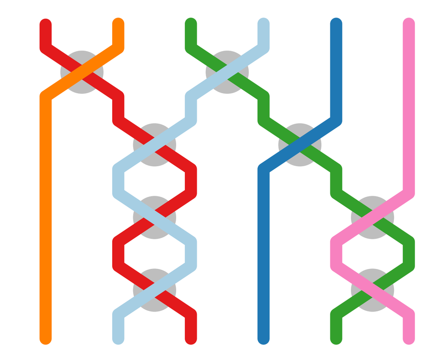
+

. . . and a *list* of swaps, i.e., a multiset of swaps between wires,



=

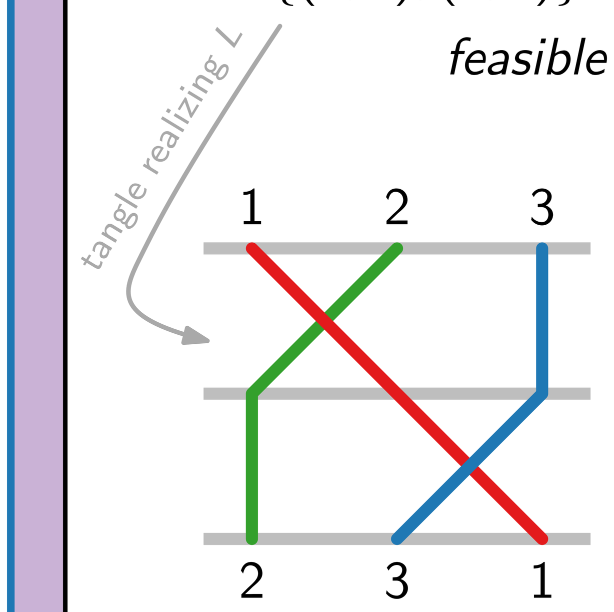
is there a tangle *realizing* these swaps?



EXAMPLE

$L = \{(1,2),(1,3)\}$

feasible ✓



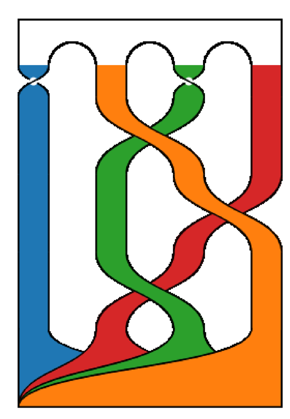
$L' = \{(1,2),(1,2),(1,3)\}$

not feasible ✗

PREV. WORK

Olszewski et al., Visualizing the template of a chaotic attractor GD 2018:

Exp.-time algorithm for finding height-optimal tangles



Firman et al., The complexity of finding tangles GD 2019:

Faster exp.-time algorithm for finding height-optimal tangles

Tangle-Height Minimization is NP-hard.

RESULT

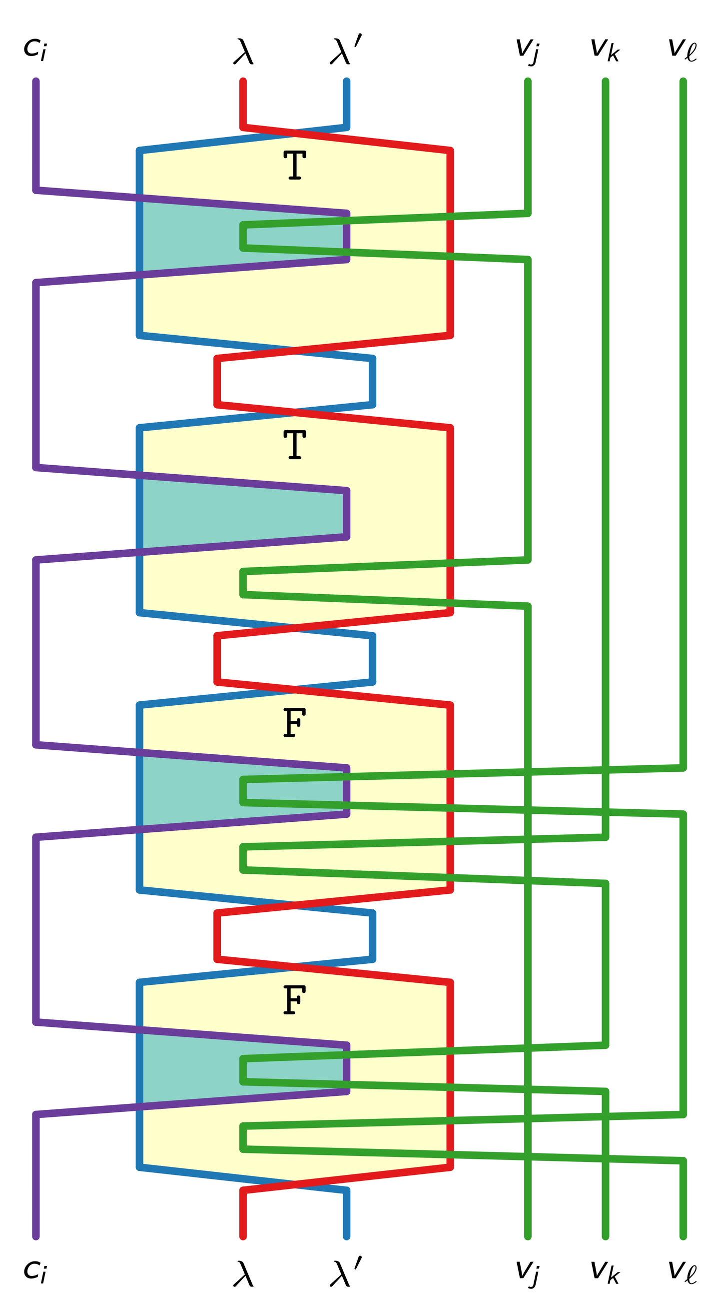
List-Feasibility is NP-hard

Reduction from Positive Not-All-Equal 3-SAT Diff

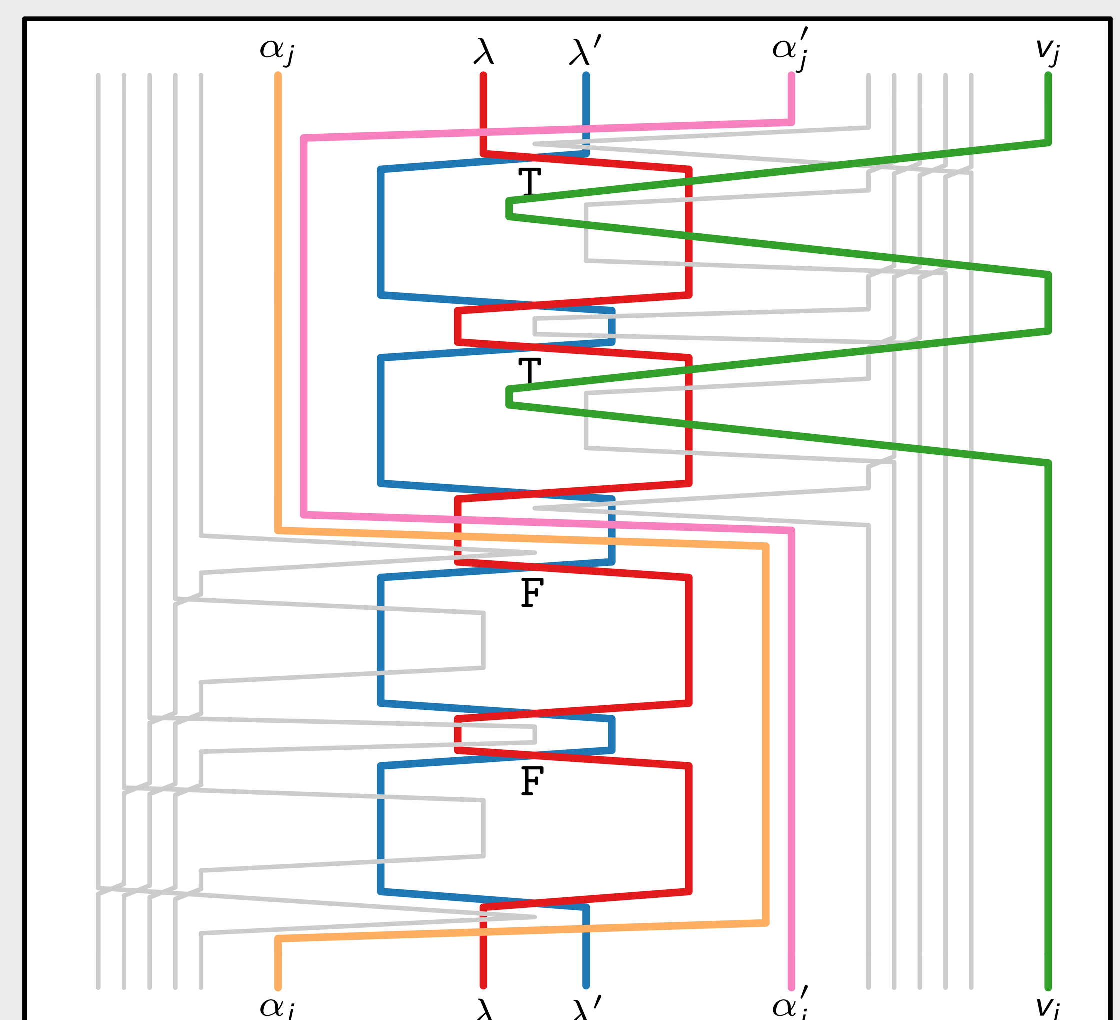
~~negative literals~~ ~~(T ∨ T ∨ T)~~ ~~(F ∨ F ∨ F)~~ ~~one variable twice in a clause~~

Main idea of the proof

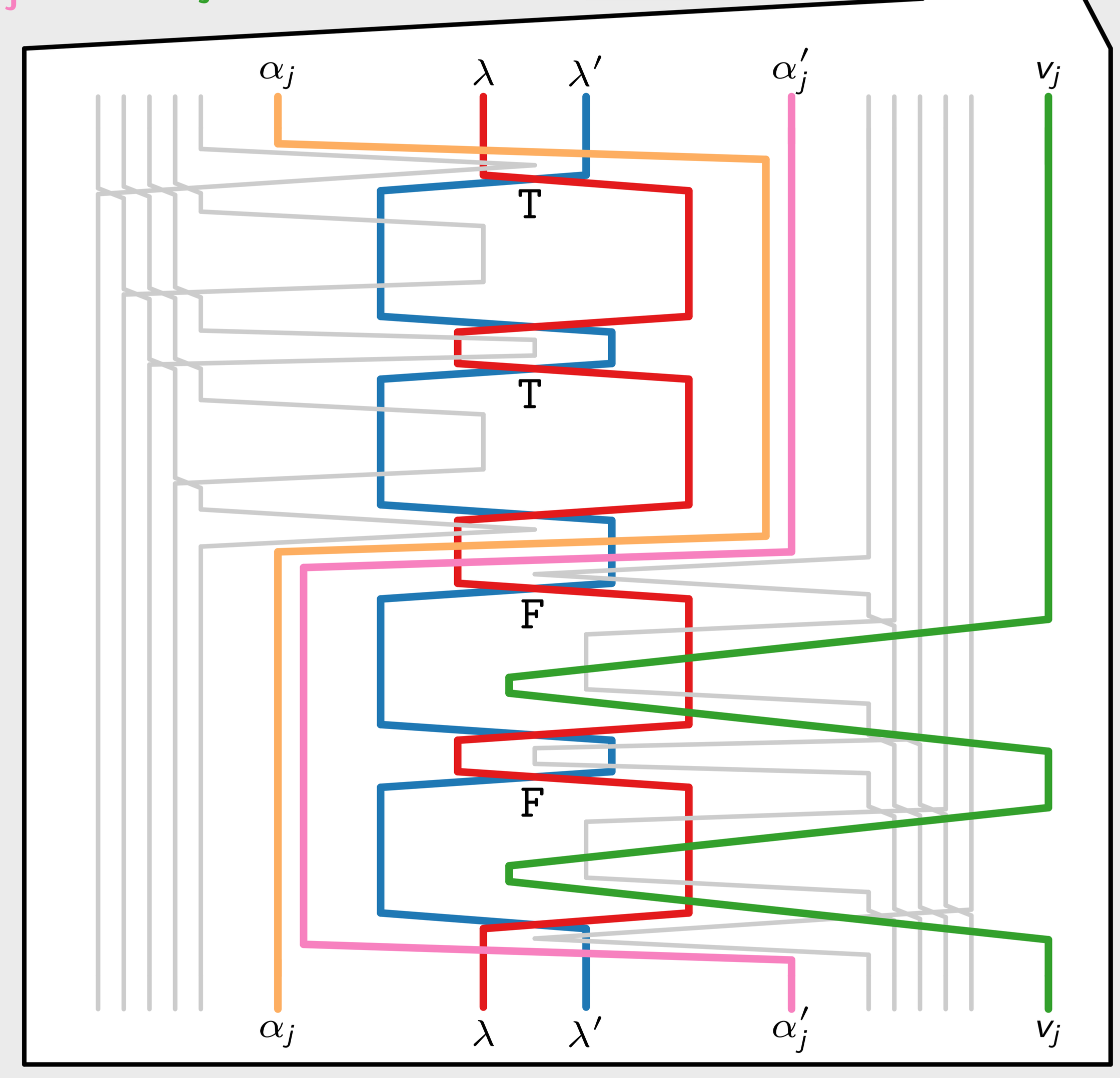
- Two wires build 4 *loops*.
- Two loops represent `True`, the other two `False`.
- For each clause, there is a **wire** with an *arm* in each of the 4 loops.
- For each variable, there is a **wire** entering either only `True` or only `False` loops.
- Each clause wire meets precisely its three corresponding variable wires – each one in a different loop.
- Exactly 2 `True` loops and 2 `False` loops ⇒ clause wires meet all their variable wires iff Positive Not-All-Equal 3-SAT Diff formula is satisfiable.



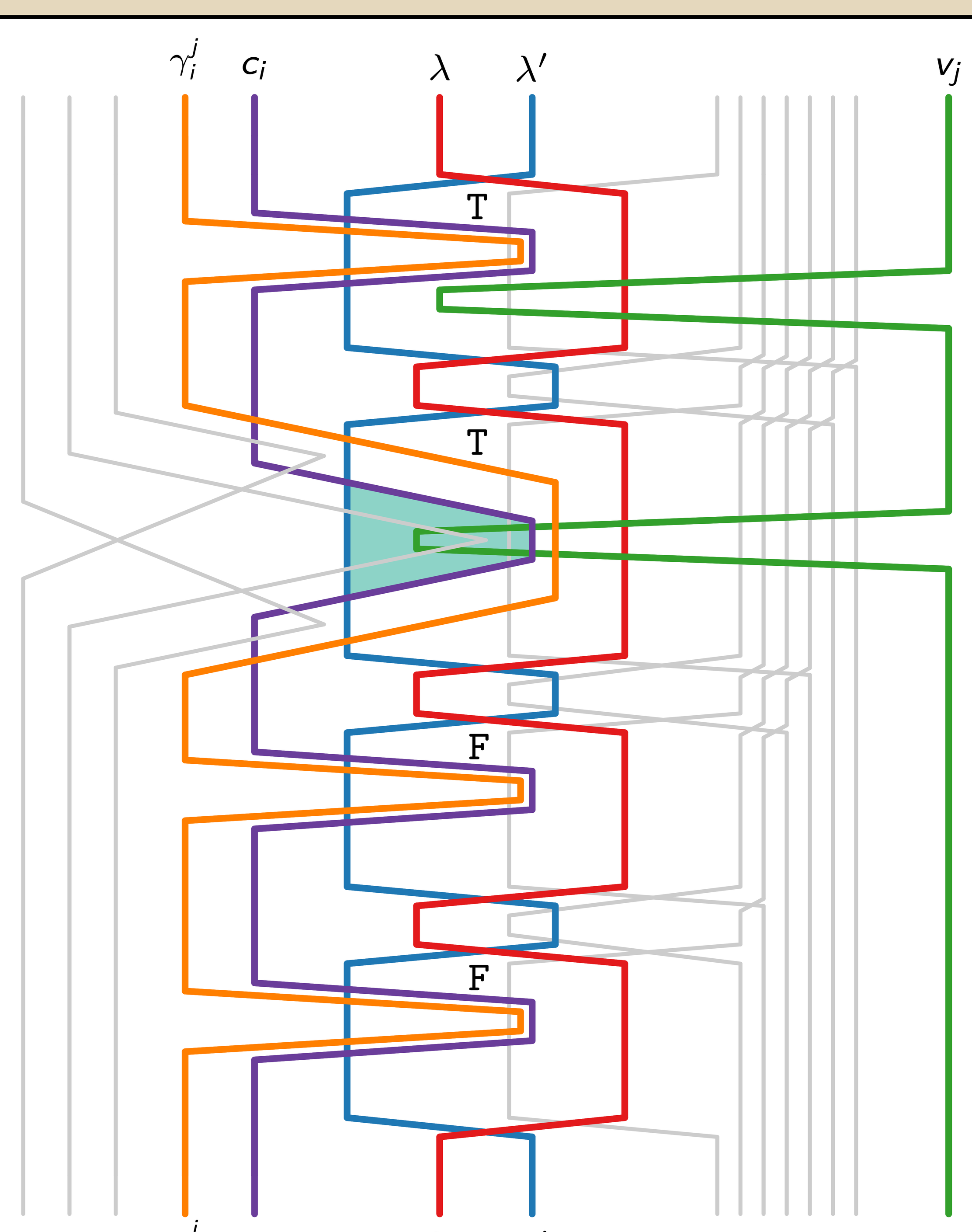
Variable gadget



α_j, α'_j make v_j appear only in `True` or in `False` loops



Clause gadget

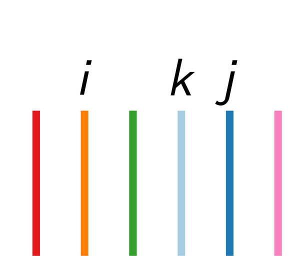


In variable and clause gadgets, the gray wires have the same rigid structure that helps to complete the proof.

Wire γ_i^j protects the arm of c_i that intersects v_j from other variable wires.

FUTURE WORK

For lists where all entries are 0 or 1, we can find a tangle that has height at most OPT+ 1 in polynomial time. Can we also always find a tangle of height OPT efficiently?



A list is *non-separable* if for any three wires $i < k < j$: ik and kj don't swap implies that ij doesn't swap.

This condition is necessary for feasibility.

For lists where all entries are even, is this sufficient?