A strategy game based approach on solving logic formulas

Part 2: PNS and QCDCL

Yifan He

Supervised By Dr. Abdallah Saffidine

Thesis B seminar

2021 T3

Outline

- Thesis A recap
- Progress in thesis B
- Future work

Outline

- Thesis A recap
- Progress in thesis B
- Future work

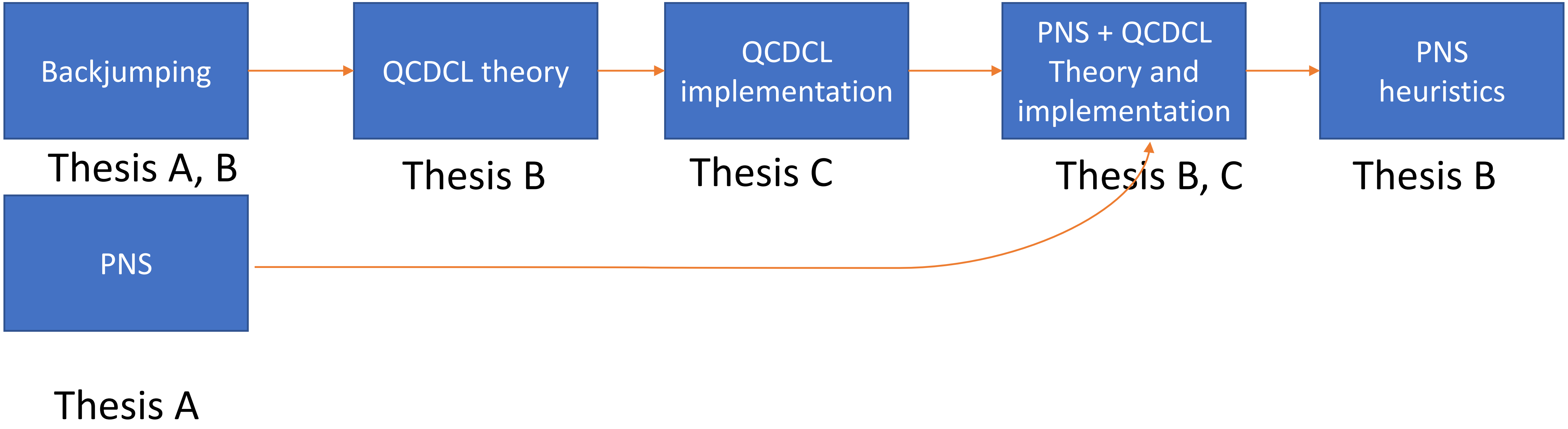
Thesis A Recap

- Literature review
  - Important definitions in QBF (semantic, unit propagation, QDLL procedure)
  - Backjumping (reason computation rules)
  - QCDCL
  - proof number search
- Define the final target
  - Combine proof number search with existing QBF solving techniques
- Progress made
  - DFS based Backjumping solver
  - PNS based Backjumping solver
  - Modified the heuristic formula for PNS solver
  - Compare the performance

Thesis A Recap (cont.)

- Original plan for thesis B:
  - Combine Backjumping with a SAT solver
  - Investigate QCDCL in greater depth (theory aspect)
  - Implement the data structure part of QCDCL
  - Investigate proof number search heuristics in the QBF world

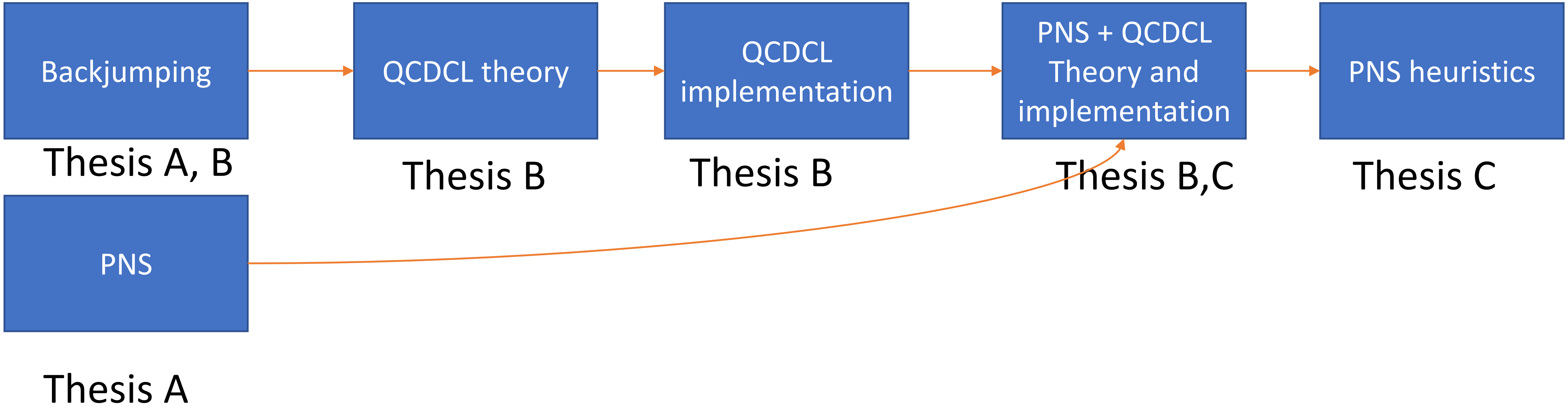
Dependency graph of the thesis tasks



Parameter tuning of PNS should happen at the last stage!

New plan in thesis B

- Backjumping with SAT solver enabled (minor priority)
- QCDCL and its implementation (DFS version)
- Combining QCDCL and PNS (theory)



Outline

- Thesis A recap
- Progress in thesis B
  - Backjumping with SAT solver enabled
  - QCDCL algorithm and implementation
  - Combining QCDCL with Proof number search
- Future work

Backjumping with SAT solver enabled

- Recap initial reason computation rule:
  - 1. If ϕ_μ contains an empty clause C. The reason for conflict is the clause C.
  - 2. If ϕ_μ has all clauses satisfied. The reason for solution can be obtained by repeatedly removing universal literals from μ such that all clauses are still satisfied.
  - 3. If the ϕ_μ contains no universal variables, the satisfiability of ϕ_μ can be determined by calling a SAT solver. The reason is μ. (weaken pruning, e.g. BLOCK family)
- Reason for internal nodes will be calculated by Q-resolution
- Update the third rule to:
  - If ϕ_μ contains no universal variables, the reason for unsatisfiability (resp. satisfiability) can be determined by repeatedly removing existential (resp. universal) literals from μ such that $\phi_{\mu'}$ is still UNSAT (resp. SAT). The reason is a minimal μ'.

Experimental results

- The computational overhead is not negligible for reason of UNSAT
- Significantly improve the performance of the solver on BLOCKS
- No significant improvement of the solver for other instances
- A stronger proof system is desired

Family	Total	#BJ-NOSAT	#BJ-OldSAT	#BJ-NewSAT
Counter (2004)	8	2	2	2
BLOCKS	13	3	0	**7**
Tree	14	11	11	11
k_d4_n	21	2	2	2
k_d4_p	21	5	6	6
k_ph_n	21	5	5	5
k_ph_p	21	4	4	**5**
k_lin_n	21	3	3	3
k_lin_p	21	5	5	5
k_dum_n	21	3	3	3
k_dum_p	21	6	6	6
k_t4p_n	21	1	1	1
k_t4p_p	21	1	1	1

Number of solved instances with a 900s time limit

Outline

- Thesis A recap
- Progress in thesis B
  - Backjumping with SAT solver enabled
  - QCDCL algorithm and implementation
  - Combining QCDCL with Proof number search
- Future work

QCDCL introduction

- Q: QBF, CDCL: Conflict Driven Clause Learning (1990s)
- Conflict driven clause learning + solution driven cube learning
- Improve Backjumping
- Proposed by l. Zhang (2002), R. Letz (2002), E. Giunchiglia (2003)

Implementation details

- Efficient QBF data structure supports:
  - Assign literal
  - Unassign literal
  - Unit clause detection
  - Conflict clause detection
  - Solution detection
  - Add new clauses
- Two watched literal data structure:
  - Efficient lazy data structure for SAT (Chaff 2001)
  - First used by QUBE (2004) and Quaffle (2006) in the QBF world
  - Reduce the number of clause iteration per variable assignment
  - Nothing is done to the data structure during backtracking

2-WL vs. Data structure used in thesis A

Instance	Time-BJOld (s)	Time-2wl-BJ (s)	Clause/ass-BJOld	Clause/ass-2wl-BJ
chain23v24	> 900	**654.12**	6.26	**1.02**
L*BWL*B1	> 900	**714.16**	235.19	**1.17**
toilet_a10.01.15	> 900	**74.75**	67.24	**1.13**
L*BWL*A1	> 900	**47.54**	153.13	**1.12**
k_path_n-4	> 900	**23.87**	8.03	**0.93**
TOILET7.1.iv.13	478.03	**232.50**	6.77	**0.98**
k_ph_p-5	123.20	**29.46**	10.58	**1.02**
k_lin_p-5	415.82	**62.22**	11.96	**0.94**
k_t4p_p-1	3.12	**1.55**	7.13	**0.88**
k_lin_n-3	282.45	**13.95**	13.95	**0.93**

Time and number of clause iterated per assignment for different Backjumping solvers

Important definitions

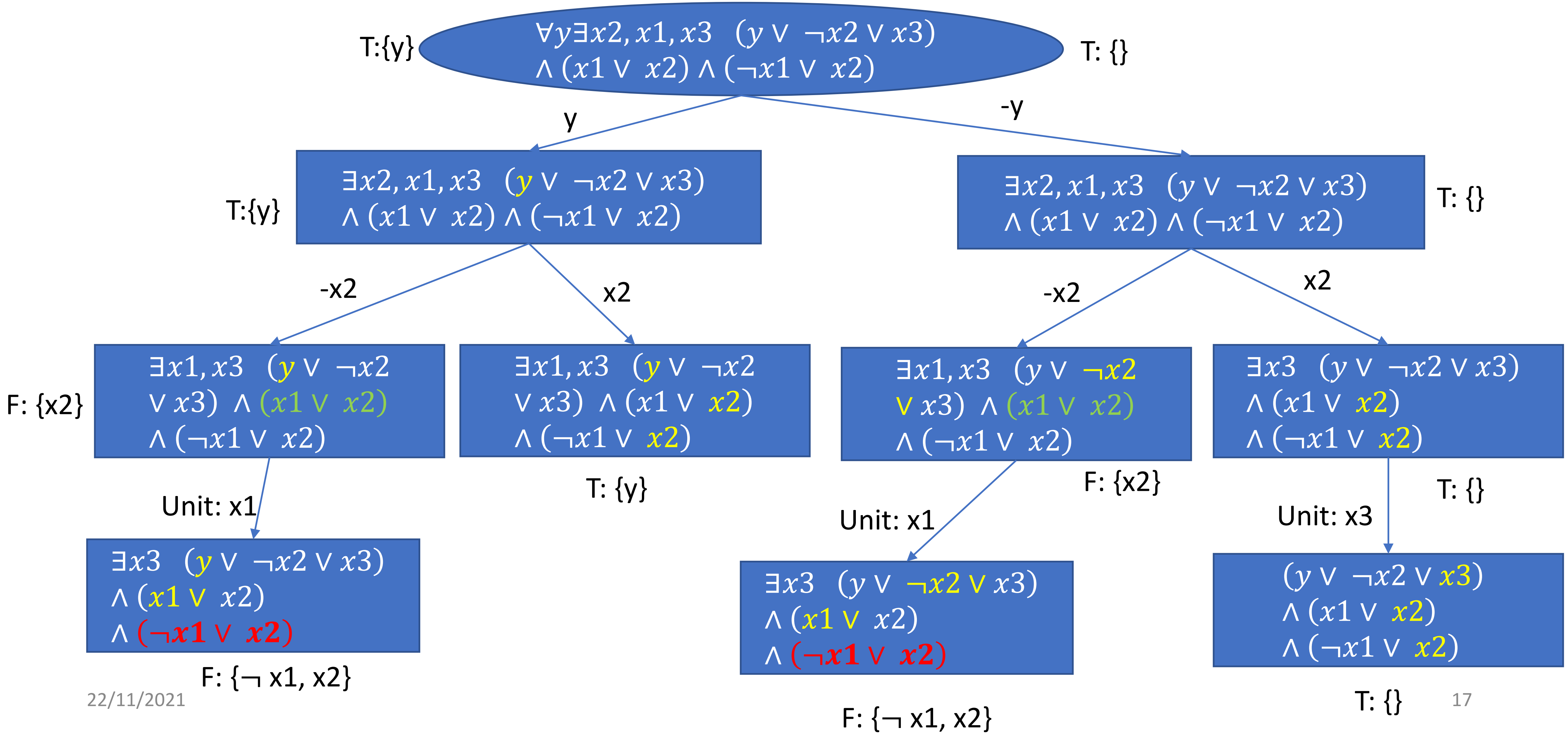
- Minimal form: A clause C is minimal if the literals in C with maximum level are existential. Universal reduction: C := min(C) is sound.
  - $\exists x \forall y \exists z \forall w \exists p \ (y \vee z \vee w)$ has equivalent minimal form $\exists x \forall y \exists z \forall w \exists p \ (y \vee z)$
- A cube T is minimal if the literals in T with maximum level are universal. Existential reduction: T := min(T) is sound.
- Q-resolution:
  - $Q\_res\_c(C1, C2) = \dfrac{C1(l) \qquad\qquad C2(\neg l)}{\min(C1 \cup C2 \ \setminus \{l, \neg l\})}$
  - $Q\_res\_t(T1, T2) = \dfrac{T1(l) \qquad\qquad T2(\neg l)}{\min(T1 \cup T2 \ \setminus \{l, \neg l\})}$

    - Example: $\dfrac{x \ \vee w \vee p \qquad\qquad\qquad z \vee \neg p}{x \vee z}$
- Q-resolution is sound and complete
- QBF in CNF is SAT (resp. UNSAT) iff empty clause cannot (resp. can) be derived by Q-resolution
- QBF in DNF is SAT (resp. UNSAT) iff empty term can (resp. cannot) be derived by Q-resolution

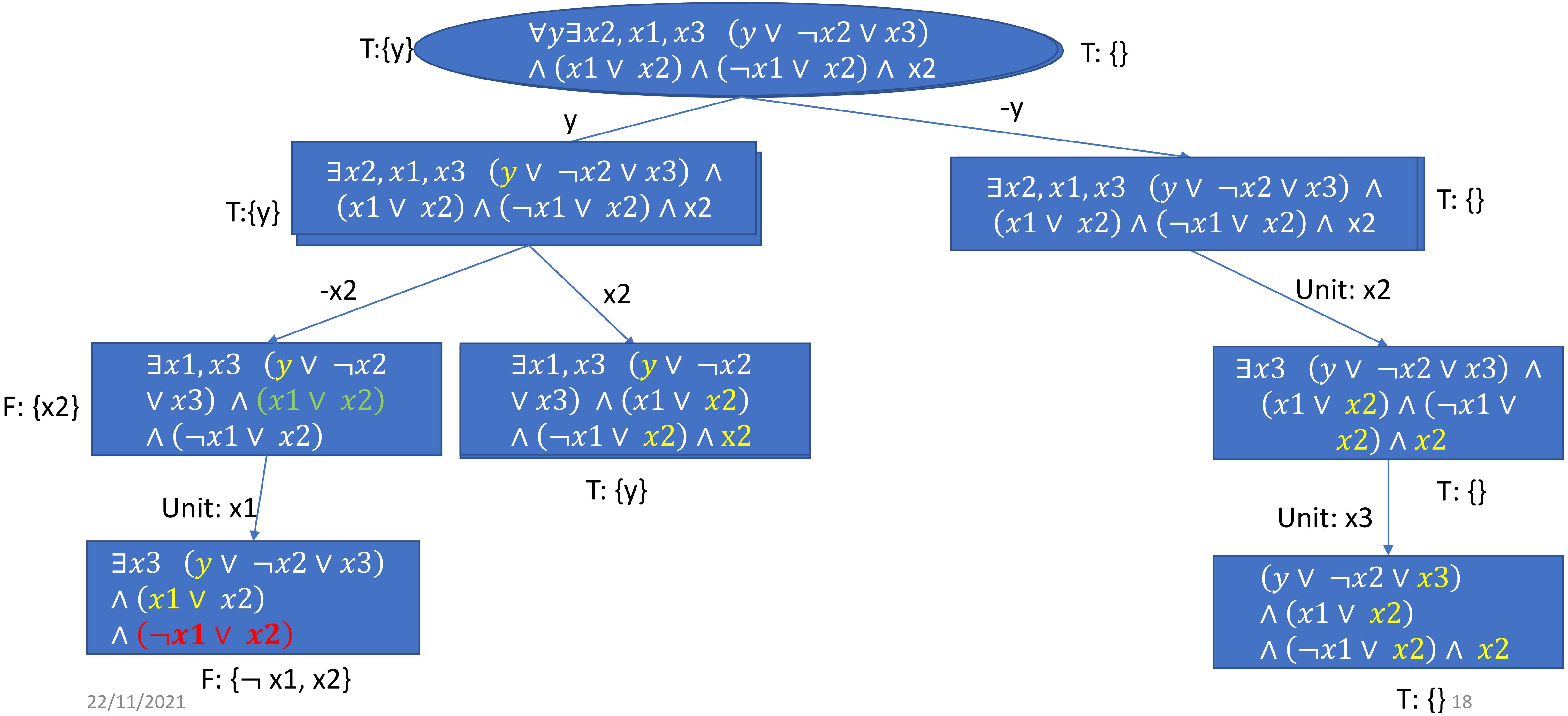
QCDCL introduction (cont.)

- Combine Q-resolution with QDLL
  - When we reach a conflict, we compute the reason for unsatisfiability
  - When we reach a solution, we compute the reason for satisfiability
  - Add Q-resolved conflicts to the formula conjunctively (i.e. clause learning)
  - Add Q-resolved solutions to the formula disjunctively (i.e. cube learning, dual to clause learning)
  - Produce more unit propagation
- Degenerates to CDCL (zChaff, Minisat) if no universal variables exist
- Used by state of art search based QBF solvers after 2006
- To make testing easier, only conflict driven clause learning and solution driven backjumping was implemented in thesis B

QCDCL example



QCDCL example (cont.)



QCDCL vs Backjumping

Family	Total	# 2wl-BJ	# 2wl-QCDCLSBJ
Counter (2004)	8	3	**4**
BLOCKS	13	**3**	2
Tree	14	11	**14**
k_d4_n	21	3	**5**
k_d4_p	21	21	21
k_ph_n	21	6	**13**
k_ph_p	21	5	**6**
k_lin_n	21	3	**8**
k_lin_p	21	5	**21**
k_dum_n	21	4	**7**
k_dum_p	21	8	**21**
k_t4p_n	21	1	**2**
k_t4p_p	21	2	**5**

Number of solved instances for pure Backjumping and CDCL + solution driven backjumping solvers

Outline

- Thesis A recap
- Progress in thesis B
  - Backjumping with SAT solver enabled
  - QCDCL algorithm and implementation
  - Combining QCDCL with Proof number search
- Future work

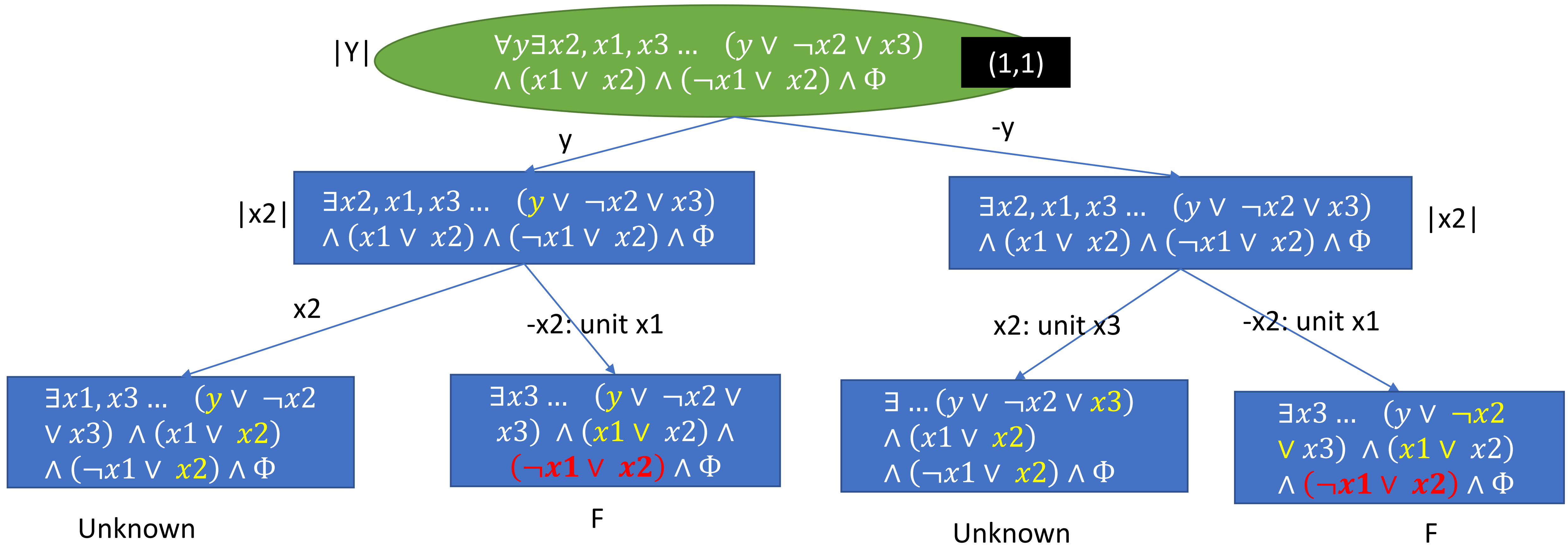
PNS and QCDCL: Motivation

- PNS + Backjumping outperforms DFS + Backjumping on gttt4x4 instances
- QCDCL outperforms Backjumping on most instances
- Combine PNS and QCDCL will be interesting

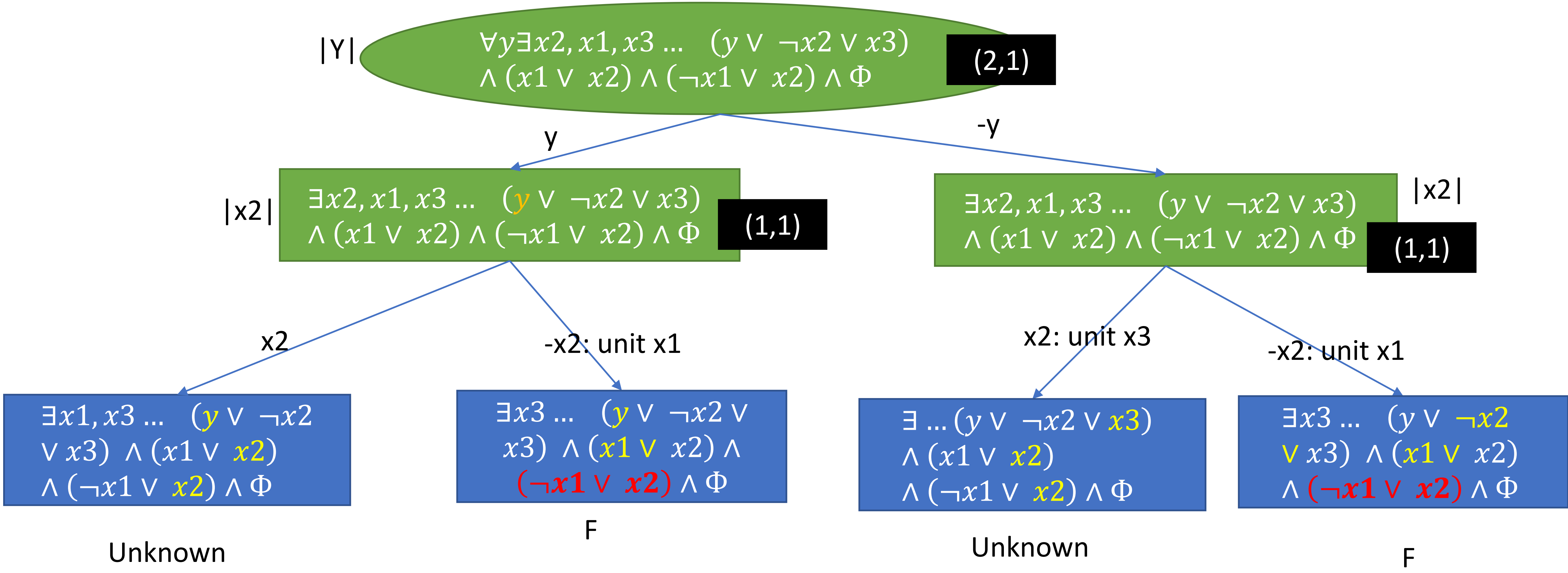
PNS and QCDCL: Difficulty

- Recall PNS + QBF: each node in the tree corresponds to a branching variable
- Searching space might change after clause learning
- Branching variable becomes unit

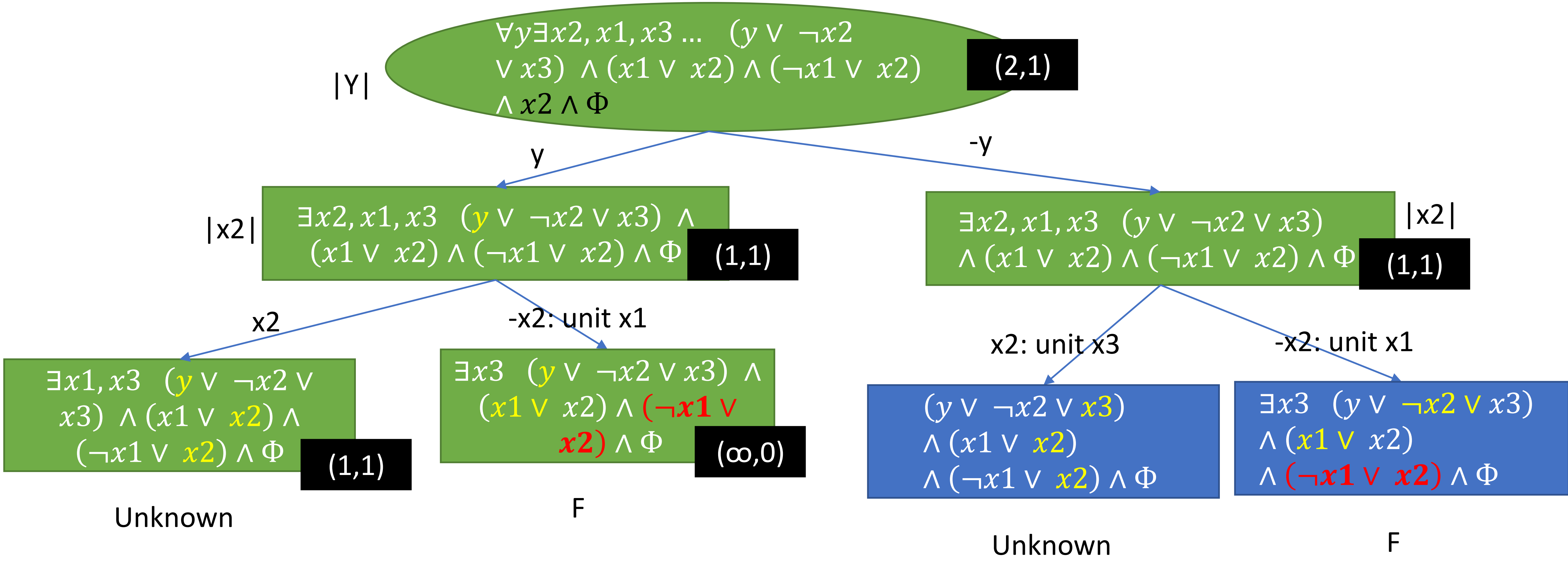
PNS + QCDCL example



PNS + QCDCL example

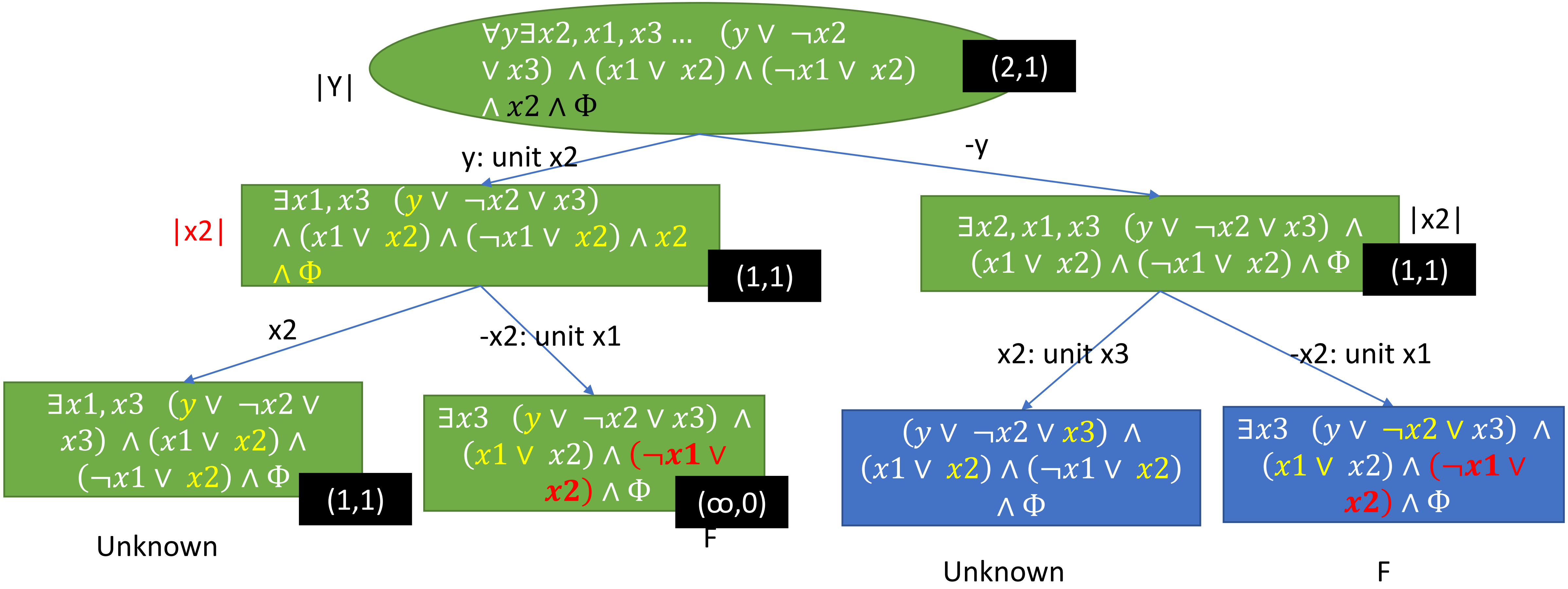


PNS + QCDCL example



learn unit literal x2, add x2 to the formula at the root after backpropagation

PNS + QCDCL example



X2 has been assigned as unit! The searching space changed, we can no longer assign x2!

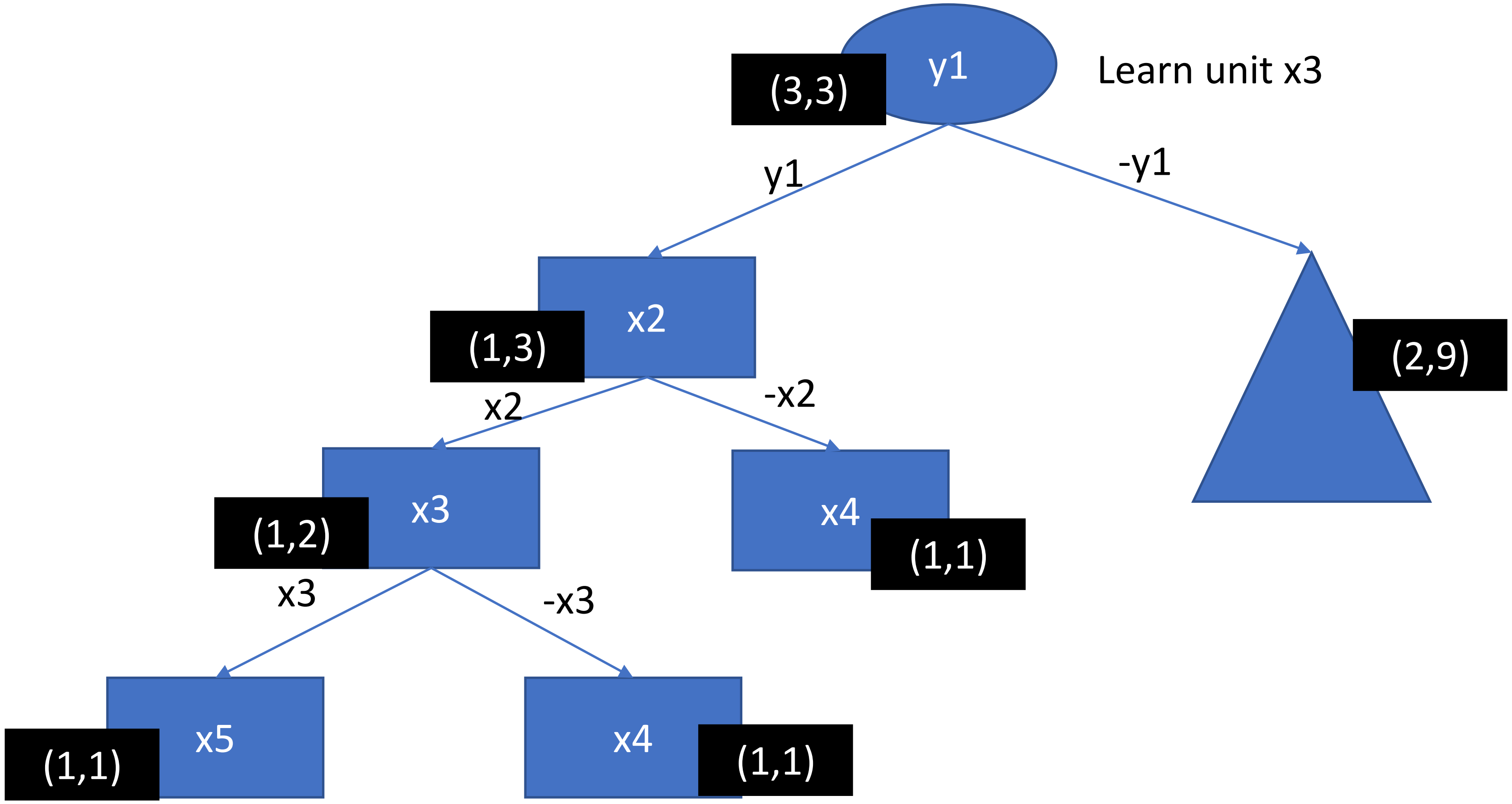
Potential solution

- Problem can only happen during the selection phase
- More unit propagation can be done because of clause learning
- Key observation:
  - Recall: $\mu = l_1 l_2 \dots l_n$is an assignment for ϕ then for each i, l_i is unit in $\phi_{l_1 l_2 .. l_{i-1}}$ or l_i is in the outermost block of $\phi_{l_1 l_2 .. l_{i-1}}$.
  - Suppose that μ is an assignment in ϕ, and C are clauses learned by Q-resolution, if $(\phi \ \wedge C)_\mu$ has no contradictory clauses, then μ is still a valid assignment in $(\phi \ \wedge C)_\mu$
  - No unit propagation can be blocked: if literal l is unit in ϕ_μ, then after clause learning l is still unit in $(\phi \ \wedge C)_\mu$

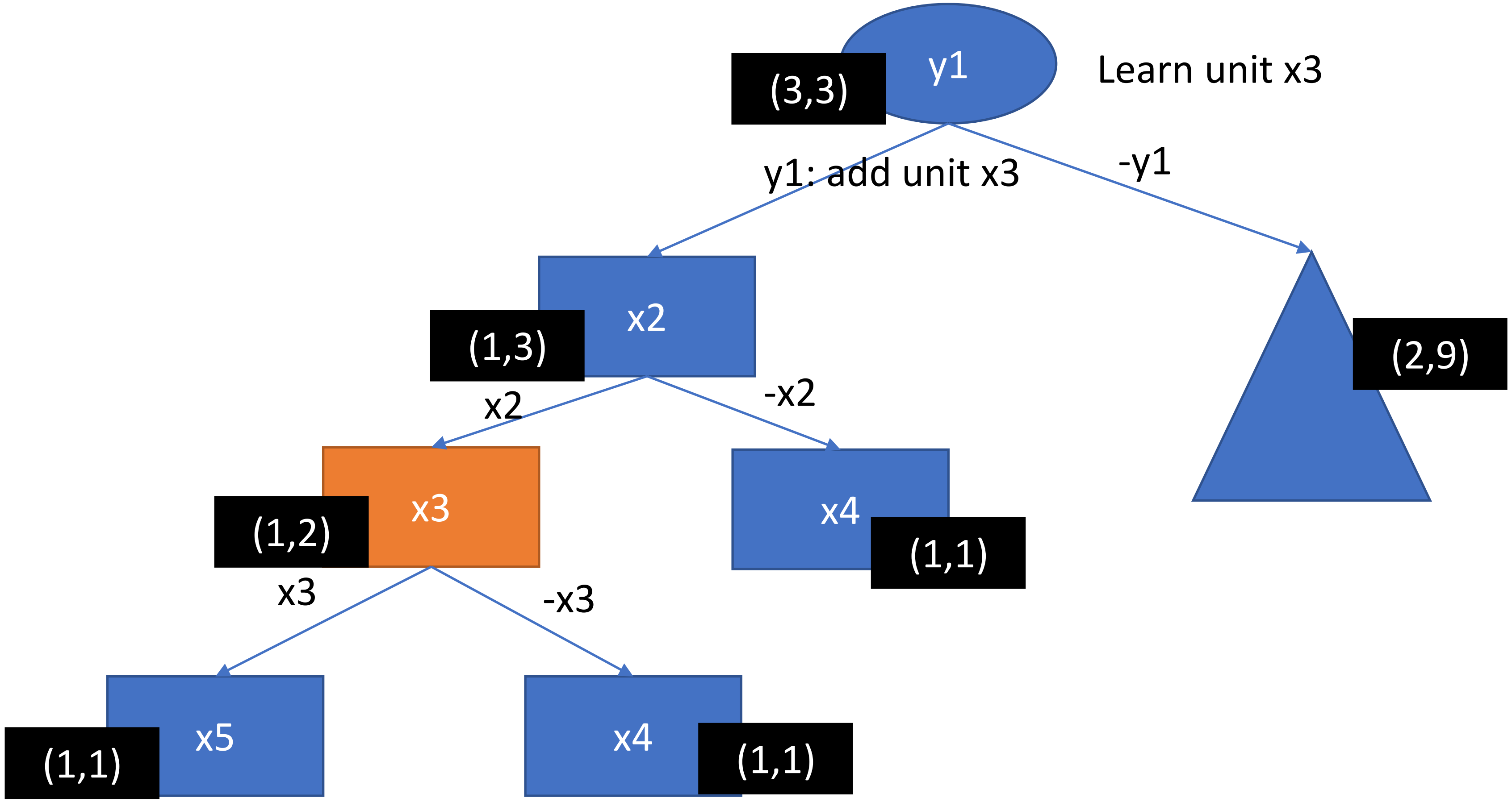
Potential solution

- Algorithm proposed by Schl ̈oter (2017) in the SAT world
- Lazy fix the search tree
- During the selection phase, for each branching variable $|l|$, if l or $-l$ has not been assigned, we still branch on $|l|$
- Otherwise, if l (resp. $-l$) is assigned as unit, we replace the current node with the left (resp. right) child
- The proof and disproof number of the current node is set to the proof and disproof number of the left (resp. right) child
- Important invariant: all unit propagation must be done immediately

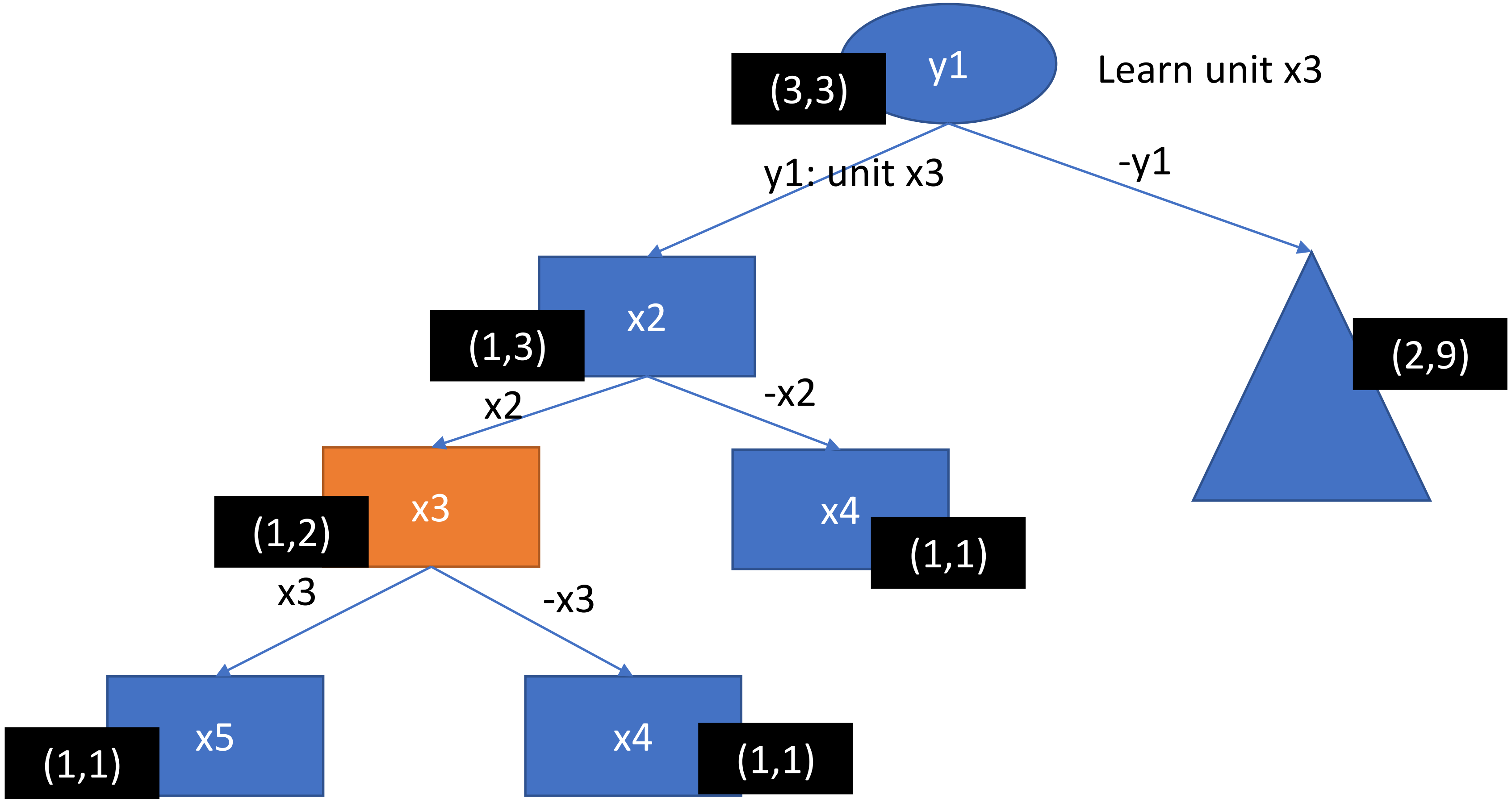
QCDCL and PNS example



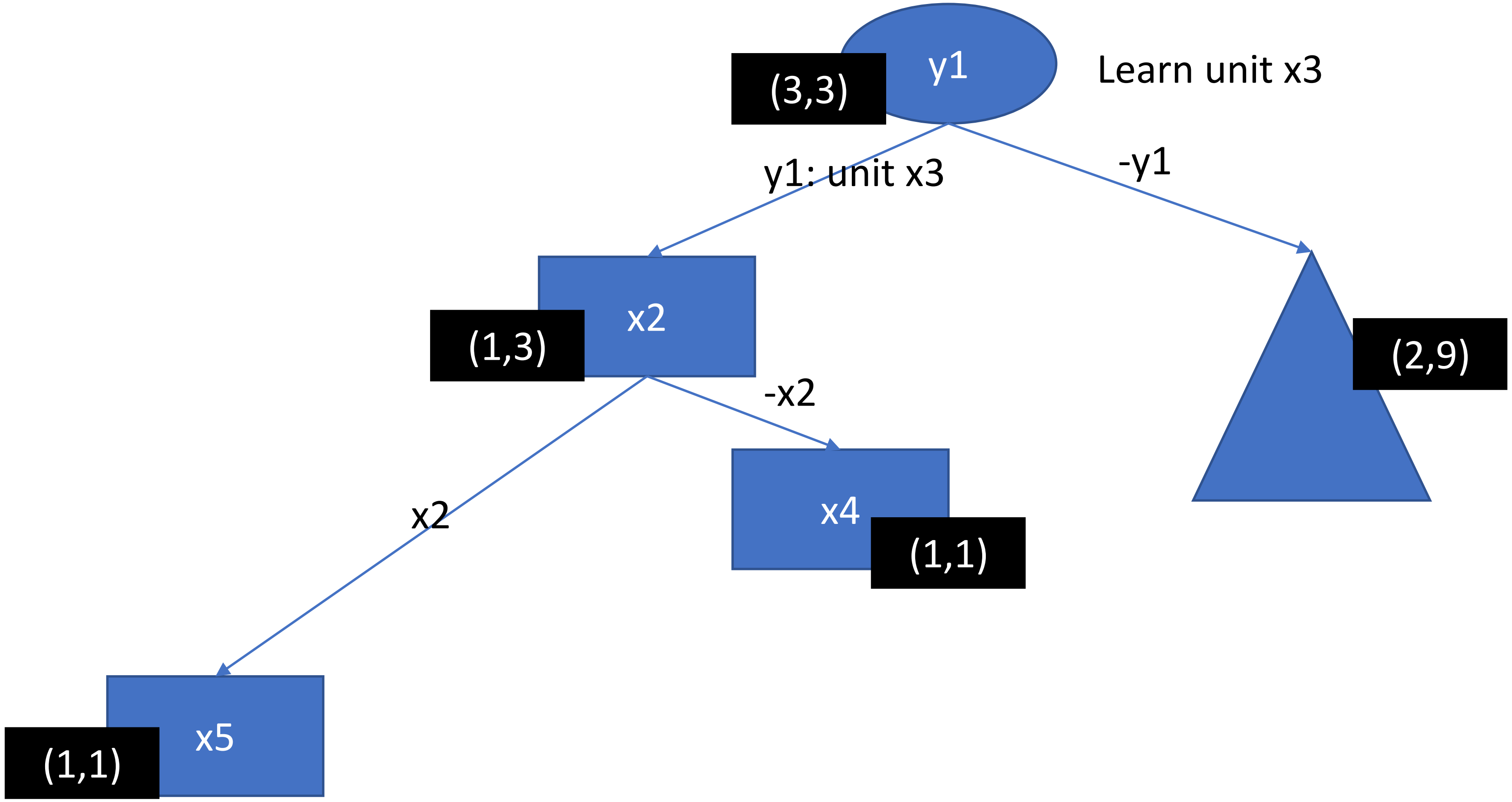
QCDCL and PNS example



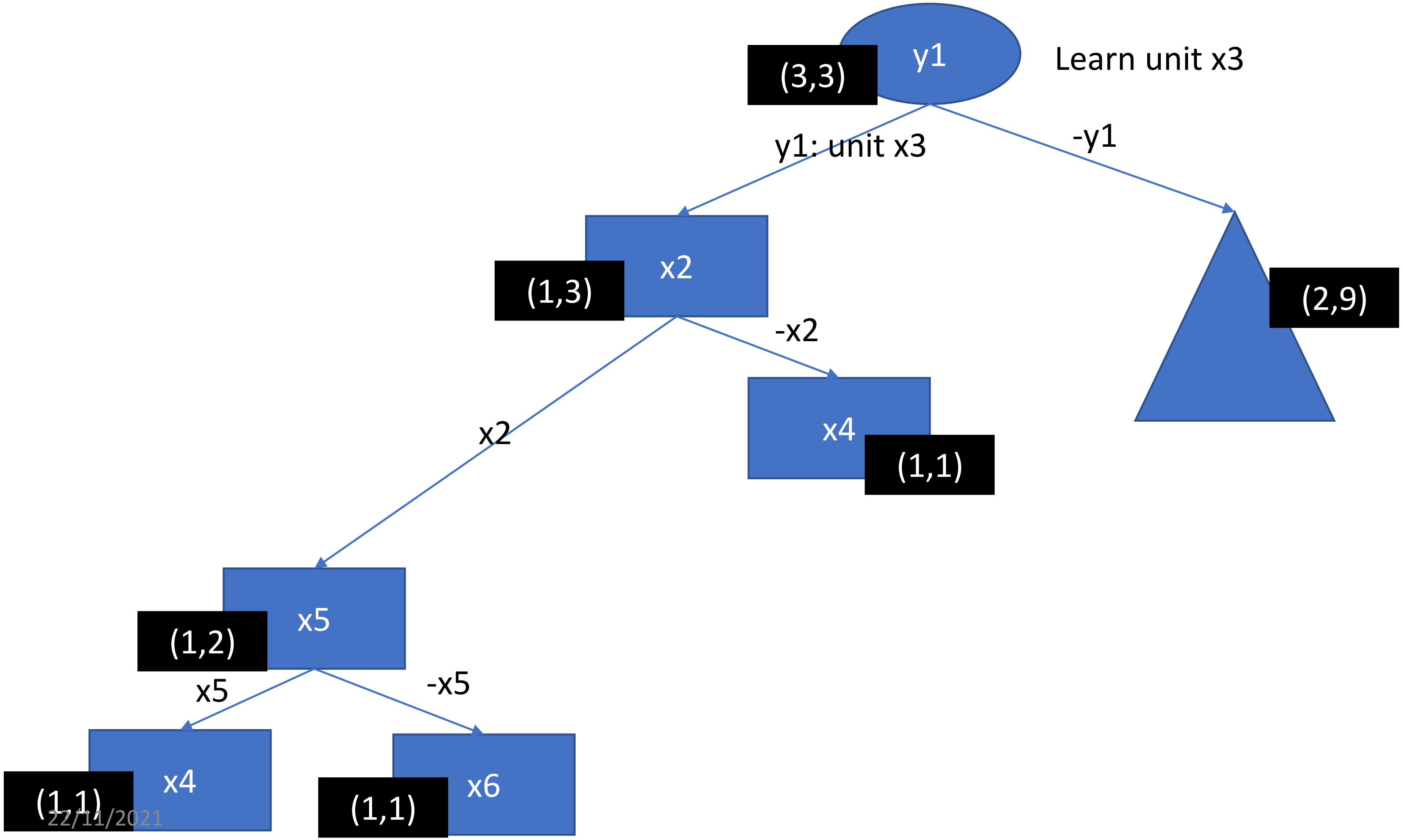
QCDCL and PNS example



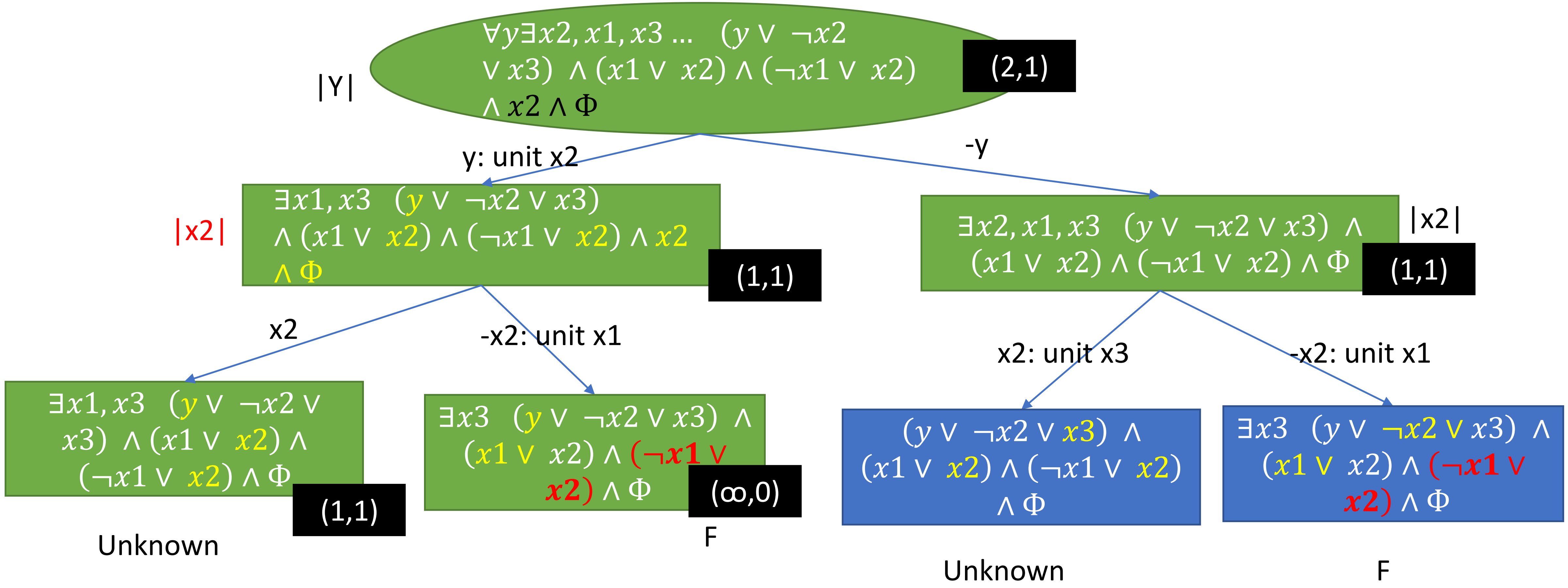
QCDCL and PNS example



QCDCL and PNS example

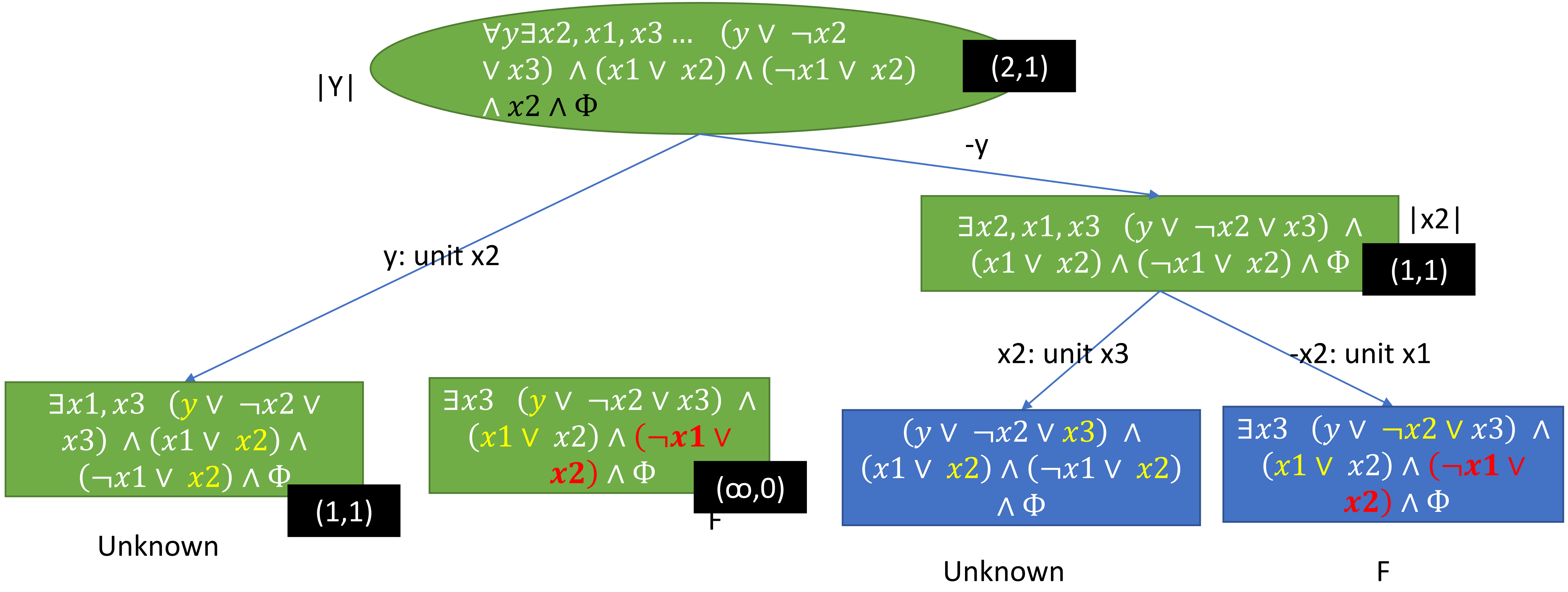


Previous PNS + QCDCL example

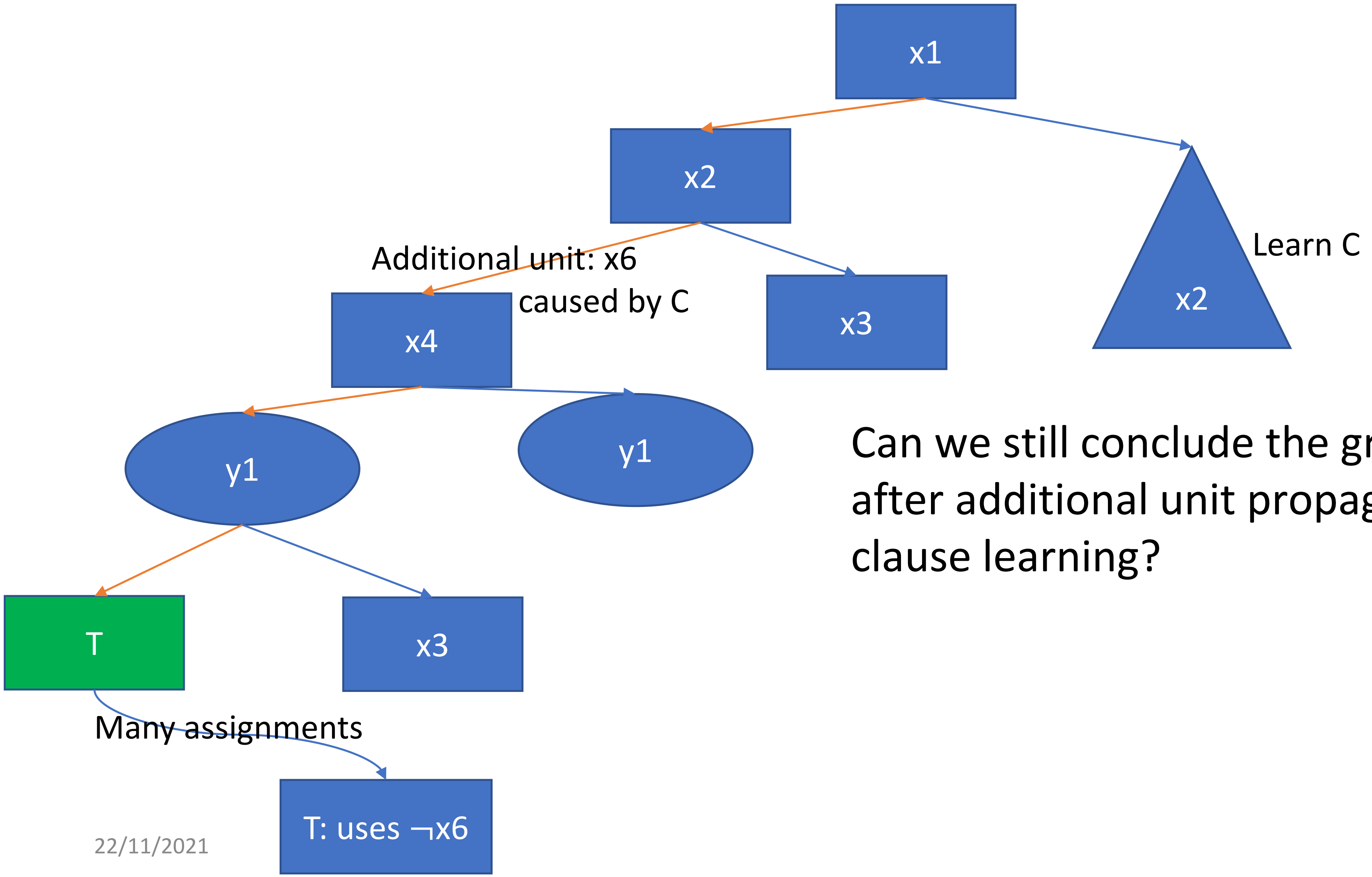


X2 has been assigned as unit! The searching space changed, we can no longer assign x2!

Previous PNS + QCDCL example



Complications in the QBF world



Can we still conclude the green node is SAT after additional unit propagation caused by clause learning?

Complications in the QBF world

- Obvious solution: cut-off the green node and replace it with a newly expanded node
  - Too many cut-off, waste our previous work in the subtree
- Key observation: The cut-off is unnecessary if the node y_1 is a unique implication point (UIP), the truth value of the green node is reusable
  - Proof is related to the definition of UIP and the property of cube learning
  - If y_1 is not a UIP, require further investigation

Thesis B summary

- Modify the reason computation rule with the SAT solver enabled
- Implement QCDCL algorithm with 2-watched literal data structure
- Investigate the algorithm that combines PNS + QCDCL

Outline

- Thesis A recap
- Progress in thesis B
  - Backjumping with SAT solver enabled
  - QCDCL algorithm and implementation
  - Combining QCDCL with Proof number search
- Future work

Future work

- PNS + QCDCL theory and implementation
- PNS heuristics in QBF world

- Expected outcome:
  - 1) worst case: implement a correct PNS + QCDCL solver/pre-preprocessor
  - 2) best case: implement a PNS + QCDCL solver that can achieve some unexpected performance on some families of instances

Thesis C timetable

Week	Task
Term break	Implement solution driven cube learning and debug the full QCDCL + 2wl solver
1 – 4	PNS + QCDCL theory cont. and implementation
5 – 6	PNS heuristics in QBF world + experiments
7 – 8	Thesis C presentation
9 - 11	Thesis C report

Reference

- Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella. Clause/term resolution andlearning in the evaluation of quantified boolean formulas.Journal of Artificial IntelligenceResearch, 26:371–416, 2006.
- Jens Schl ̈oter. A monte carlo tree search based conflict-driven clause learning sat solver. InGI-Jahrestagung, 2017.
- Gent, I., Giunchiglia, E., Narizzano, M., Rowley, A., & Tacchella, A. (2004). Watched Data Structures for QBF Solvers. Lecture Notes in Computer Science, 25–36. https://doi.org/10.1007/978-3-540-24605-3_3