Knowledge Representation Lecture 5

Knowledge Representation (Vrije Universiteit Amsterdam)

KR 2021 – Lecture 5

MAXSAT

- Paradigm have different representation
- Models where you can represent dependence & independence of variables
- Propositional logic: If you have lots of clauses (unlike sudoku because the rules are the same) there can be preference between clauses, contradiction between clauses means you cant use it anymore in PL.
    - People have started thinking of ways to deal with contradictions
- SAT problem start with CNF formula
    - Clause set -> inconsistency -> useless to us
- MAXsat is useful in repair, alternative semantic. It is a variant of SAT solving problem that can help in inconsistent scenarios
- Slide 4: F = x must be false, x or y must be true, y must be false but we see that this makes x or y false so it is unsatisfiable
    - So what you can do is find a truth assignment such that x is T, y is F, z is T, and W is F. that satisfies 3/4 clauses
    - This is better than having None
    - This is a solution to MAXSat problem because those or max clauses that can be satisfied
- Many cases in practise you have a variant of MAXSat where some clauses are more important then the other so that will have some positive costs that will occur when you falsify a clause
    - Previously we falsified x (we ignored it), it costs us something
    - Different clauses can have different costs (e.g. infinite, 4, 1)
        - You can ignore a number but not the infinite ones
- These clauses which we assigned cost of infinity, those we cannot ignore- they are called hard clauses
- The other ones are soft clauses
- Now the MAXSat is defined by finding a truth assignment that assigns a minimum finite cost to the function
    - The cost in this case are then of the clauses that are falsified
    - It needs to be finite cost (soft)
        - A hard clause (infinite) would violate MAXSat
- When you minimize or maximize it becomes an optimisation problem
- You have variants of MAXSat
    - Standard
        - Maximize satisfied clauses
    - Weight
        - No infinite assignment
    - Partial
    - Weighted partial
- We are still interested in being more efficient, so it doesn't stop at DPLL. MAXSat helps you with more complicated problems ()

EXAM (ASK ABOUT CHEAT SHEET)

- digital exam on campus, 25 question
- Calculate things; could be question with MAXSat (model in hard way with cost penalty)
- Understand difference between just knowing satisfiability and proving entailment by refutation
- In a proof by contradiction, such as DPLL, I can prove that a formula F is entailed by a knowledge base KB by showing that:
    - The answer is a definition from the lectures, knowledge reproduction question
        - Proof by refutation
    - Answer C: the KB and the negation are unsatisfiable
    - DPLL can only do one thing and that is SAT solving, you give it a set of formulas then all it does is say whether there is a model or not. Now what we want to show (entailment) follows from our KB. That basically means that something is T in all the models. In all the models F is true, this is not something DP can help us with.
        - DP shows whether there is a model, we want to see if something is T in all the models
    - If you assume if the thing you want to find is F then all KB should be unsatisfiable
        - E.g. You want to whether it is raining today with the KB you have. You can then assume that it is not raining (negation) then it should be inconsistent and therefore unsatisfiable with the KB. This way you can proof it is raining outside with your KB
- Consider a language L defined as follows:

  •Syntax
  {A,B,C} ∈ L
  If F1, F2, F3 ∈ L then "(F1*F2*F3)" ∈ L
  •Semantics: let I be an interpretation function:
  I(A)=1, I(B)=2, I(C)=3
  I(F1*F2*F3) = I(F3).I(F1).I(F2)
  What is I(A*B*(C*A*B))?

        - Lecture 2 & 3
        - Logic engineer question about understanding
        - Idea is that we have some kind of real world problem that we want to develop a logical system for. Logical systems have some kind of definition of what valid statements are so that is language over which of the sentences can you make statements in the first place
        - In PL we had propositions as syntactically correct statement
            - We had a or b, a and b- we define in PL what we could talk about
        - This question gives us syntax and semantic- then what is it calculating?
            - I is an interpretation function
        - The dots and stars, they have no definition
        - a, b, c are simple languages
        - f1, f2, f3, are in my language if they have stars between
            - the f can be any of the things that are defined before: for example-

              ((A*B*C) * (A*B*A) * (A*A*A))

                - you can build something complicated like above or simple things like a * a * a
                - kind of like CNF, you can have a simple CNF or a CNF in a CNF

        - semantics tell us the meaning of a is 1 (interpretation function) – I applied to a gives us 1- base case
    - I(F1*F2*F3) = I(f3).I(f1).I(f2)
        - I(A * B * C) = I(C). J(A). J(B) = 3.1.2
    - I(A * B(C*A*B))
        - F1 = A, F2= B, F3=(C * A * B)
        - J(C * A *B). J(A). J(B)
        - 2.312.1
    - Has to be mathematically valid
- Fill in the truth value of the following formulas under the assumption that A,B,C, and D are all false. Fill in the truth values of the formulas (use the words T, F)

  (A & B) v -C v -D is __________
  (A & B) v (-C & D) is __________
  (A->B) v (C->D) is __________
  (-A->B) v (-C->D) is __________

    - falseAandfalseB (T) or notfalseC (T) or NotfalseD (T) = True
    - False
    - True
    - False
- Which of the following is true? The propositional statement (PVQ) -> (P&Q)

  A is satisfiable, but not valid
  B is valid
  C is a contradiction
  D neither valid, satisfiable nor a contradiction

    - Contradiction means unsatisfiable
        - Make a truth table

P	Q	PvQ	Q&P	
1	1	1	1	1
1	0	1	0	0
0	1	1	0	0
0	0	0	0	1

ction

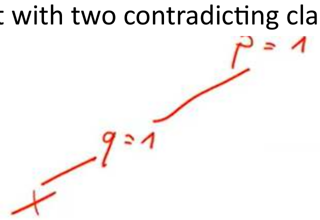
    - Answer is a
        - Model makes it T, there are ways it can be T
        - But is not true in all models so it is not valid
- Automated reasoning can often be seen as search procedure, that tries to find an order of rules to formulas in a knowledge base. Which of the following statements about the DPLL algorithm and search is correct?

  A DPLL recursively searches through all possible variable assignments for a model, i.e. an interpretation that satisfies all th clauses.
  B DPLL iteratively searches through the set of all clauses for one that is satisfied by a given interpretation.
  C DPLL exhaustively applies a set of transformation rules to produce a contradiction
  D DPLL cannot be seen as a search procedure, it only randomly assigns values to propositions until it finds an assignment that satisfies all the clauses

    - Automatic reasoning can be seen as a search procedure. In this question we need to use knowledge
    - Answer is A

    - DPLL looks for a possible partial variable assignment then builds a tree with each node is a partial assignment, we have state space model where each state is a partial assignment, you take an action assigning one of the values by doing that we are trying to find an interpretation that satisfies all clauses
        - Goes systematically through all of them- but when you take pruning into account A is the right answer
    - When you stop recursion because you find inconsistency then all the assignment you would check underneath (the tree you are pruning) are impossible assignment so you can prune it
    - Answer C is wrong- we use transformation when we make CNF but after that we do not use any rules
        - This is in DL where you use calculus to find contradiction
        - Here we are using a model that satisfy clauses, contradiction is that it is not going well
- Consider the following KB (set of clauses) and show whether the KB is satisfiable or not using DPLL. Use the following order: unit, pure, split alphabetic with a positive value

  1: ¬ p v q v r
  2: p v ¬ q v ¬ r
  3: ¬ p v ¬ q v ¬ r
  4: p v q
  5: p v r
  6: ¬ p v ¬ q v r

    - Do it on a piece of paper
    - We don’t a unit clause, we don’t have pure either
    - The p has to be positive (and it is the first letter) so it is T (p=1)
        - Then we can delete clause 2,4, and 5 because having p=1 make the clause true, thus the clause can be deleted
        - You can simplify non-ps (negations)
    - Then can again check unit clauses, pure literals, so you can branch (alphabetically)
    - The q has to be positive (q comes after p) so it is T q =1
        - What can you delete?
            - Non-q in clause 3 and 6
        - Makes clause 1 true so you can delete it
    - You are left with two contradicting clauses (r and non-r)
        - 

    - Now you have to backtrack
        - We keep p T and make q F
            - Branch on q
        - Delete clause 1,2,6
    - You are left with r and it is a unit clause so it becomes T
    - Then our model is p = 1, q = 0 and r = 1
- GSAT is complete w.r.t Propositional Logic satisfiability? T or F?
    - If you would flip one of the assignment with polarity, are you closer to the solution, do you satisfy more clauses?
    - You do some local search and find for solution

- If you want one solution it is a good approach but if you want all models then this is not good because it is not complete
- It can also say that it didn't find anything but there could be one that it was unable to find
- Answer is False, it is not complete