CEE 5290/COM S 5722/ORIE 5340 Heuristic Methods for Optimization
Homework 4: Satisfiability and Tabu Search
Assigned: Fri Sept 23, 2011
Due: Fri Sept 30, 2011
TA Office Hours: Tues (3:00-4:30), Thur (3:00-4:30) in Hollister 203
Prof.Shoemaker Office Hours: Tues ,Wed, Fri 2:30-3:30

Readings: Lecture notes on Satisfiability by Prof Selman (see blackboard) and Tabu Search

1. ***(no code writing)*** **Satisfiability Formulation**: In propositional calculus it is possible to represent logical expressions as clauses by applying a sequence of simple logical operators. For example, "C is true if either A or B is true" can be represented as C = $A \vee B$, where $\vee$ represents a logical 'OR'. Similarly, $\wedge$ represents a logical 'AND'; and $\neg$ represents a logical 'NOT'.

A problem can be stated in conjunctive normal form (CNF) as a sequence of clauses that must be satisfied for the problem to be solved. For example,

$$E = (A \vee B) \wedge (\neg A \vee C) \wedge (C \vee \neg D)$$

is in conjunctive normal form with 3 clauses and 4 variables. Note that the variables take on either a 0 (FALSE) or 1 (TRUE) assignment. We will attempt to formulate a simple assignment problem as a 2-SAT (containing 2 literals in each clause) problem.

Ann, Brad, Cindy, Dan, Eugene and Frank have to get back to their apartment from a club. They have exactly one car, and not all can fit into the car. Anyone not in the car will have to get a ride from the bartender who will not leave for another hour. After an inebriated argument they decide on the following:

(a) Only Eugene and Dan can drive, so at least one of them should be in the car

(b) At least one of the women should be in the car

(c) If Ann is not in the car then Brad and Frank must accompany her since the bartender has been hitting on her

(d) If Cindy is in the car then Frank cannot be in the car. If Cindy is not in the car then Eugene must go with her

Consider the problem of assigning people to go in the car given the constraints above:

(i) How many decision variables should this problem have in order to be formulated as a 2-SAT problem? What is the total number of possible assignments?

(ii) Identify appropriate clauses using only 2 literals in each to reflect their requirements. Use the first letter of the name to represent each person in the clauses. Give the CNF representation for the problem. (Hint: Six clauses should be sufficient)

(iii) Is this problem satisfiable? If so, come up with a satisfiable assignment. (Hint: You do not have to enumerate all possible assignments; this can be solved by logical reasoning)

2. ***(involves coding)*** **Solving 3-SAT with Tabu Search**: Design and Implement a Tabu Search to solve the Satisfiability instance which is provided on the course web page (as a MATLAB load-able text file **uf20_01.txt**; type *load uf20_01.txt* at the command prompt). This is a randomly generated 3-SAT instance with 20 variables, 91 clauses, and is taken from the SATLIB library of benchmark problems on the web. Each row in the matrix corresponds to a different clause. For example the first row, which reads [4 -18 19] means that the first clause is (s4 + (~s18) + (s19)), where + indicates a logical OR, and ~ a logical NOT, and s*k* is the k^{th} binary variable.

Write a short-term tabu search algorithm (Tabu.m) to solve the Satisfiability instance uf20_01.txt:

- Base the Tabu Algorithm on the pseudo-code given on the last page of this homework. NOTE: this is modified from the version in the Xeroxed text.

- Define neighbors in this satisfiability problem as those solutions that differ from the current solution by only one bit.

- Use the *best solution aspiration criterion*: accept a tabu move if and only if it leads to a cost lower than any previously encountered. For this problem, a flip on a bit is tabu if the flip has occurred on the same bit in the last K moves.

- The algorithm parameter that you should *e*xperiment with is the tabu tenure length K=|T|.

- Let your tabu list T consist of the index of bits which have been flipped in the last K iterations, i.e. it is tabu to flip a bit that has been flipped within the past K iterations.

- This is a *deterministic* Tabu implementation (i.e.set V=number of neighbors in the neighborhood (Note: This means that all memebers of the neighborhood of the current solution)).

- You must also write a cost function for this satisfiability problem (costSAT.m).

Questions

(i) Submit via email Matlab code for *only* Tabu.m, costSAT.m (cost function evaluation) and m-files for the neighborhood implementations *if* your neighborhoods are implemented outside of Tabu.m.

(ii) How many different solutions or variable assignments (i.e. satisfiable and unsatisfiable solutions) are there in this problem?

(iii) Design an experiment to determine the best best values of K. Briefly outline the design of the experiment and any assumptions. There are many reasonable answers.

(iv) *Fo*r the best value of K found above, solve the instance 100 times from 100 different initial solutions. You should determine the number of iterations you wish to run and justify why you picked that number (e.g. *x* iterations is sufficient to find a satisfiable solution). In the past, students have used number of iterations of around 100. Report the following:

- The number of trials for which you find a satisfying solution and if you found more than one satisfiable solution (not how many total, just note if there are more than one).
- The number of cost function evaluations required on average to find *the satisfying solutions* (do not count trials that do not find satisfying solutions in this average)
- For any trials that do not find satisfying solutions, report the number of cost function evaluations per trial and average value of your best objective function(aspiration level) over 100 trials .

3. *(no code writing)*CYCLING: Assume your decision domain is the set of permutations of the numbers 1 to 10. Assume you want to design a cycle so that your neighborhood is based on all possible pairwise swaps involving the number in the k^{th} position of the current solution. (For example for (9 7 6 5 4 3 2 1 8), the number in the k^{th} postion for k=3 position is 6) Then in the first interation k=1, in the second iteration k=2, etc.

i.) How many permutations are there in the domain?
ii.) What is the size of the neighborhood?
iii.) If your initial solution value is (9 8 7 6 5 4 3 2 1 0), what are the members of the neighborhood in the first iteration?
iv.) What do you do in the 21^{st} iteration to create the neighborhood?
v.) What is the cycle length of this scheme?
vi.) What is the advantage of using this cycling scheme over using the whole domain as the neighborhood?
vii.) If you used the whole domain as the neighborbood, how many iterations would you use?

(Permutation decision vectors have been discussed in lecture and some additional information relevant to this question will be discussed in lecture on Mon. Sept. 23.)

Corrected version of TABU Search Algorithm in the text:

Start with an initial feasible solution
Initialize Tabu lists (**T**) and aspiration level (**AL**)
FOR fixed number of iterations **DO**
    Generate a candidate list of neighboring solutions **V*** from the neighborhood
    Find best solution (**S*)** in **V***
        IF move S to S* is not in **T THEN**
            Accept move and update solution
            Update Tabu list
            IF cost(**S***) < **AL THEN**
                Update aspiration level
            ENDIF
        ELSE
            IF cost(**S***) < **AL THEN**
                Accept move and update best solution
                Update tabu list and aspiration level
            ELSE /* correction */
                Find next best S (S_2*) in **V*** that is not Tabu
                Accept move to S_2*
                Update tabu list
            ENDIF
        ENDIF
        Increment iteration number
END

NOTES on above:

- General algorithm that can be implemented deterministically or stochastically.
- Size of the candidate list **V*** must be greater than Tabu tenure in the above algorithm since it assumes there will always be at least one move that is not Tabu
- Candidate list = sample/subset of neighboring solutions