Beighle, John

CSCE686 Homework

Spring'20

HW7: (due 6/16) "Please indicate appropriate references"

a) *(10pts) Talbi #2.3 "VCP - Neighborhood"*

A vertex cover of an undirected graph G = (V, E) is a subset V ′ of the vertices of the graph containing at least one of the two end points of each edge:

$V' \subseteq V : \forall\{a, b\} \in E : a \in V' \, OR\, b \in V'$

The vertex cover problem is the optimization problem of finding a vertex cover of minimum size in a graph. In the graph of the Fig. 2.45, A, C, E, F is an example of a vertex cover. B, D, E is another vertex cover that is smaller. The vertex cover problem is related to the independent set problem: V ′ is an independent set if and only if its complement, V \ V ′, is a vertex cover set. Hence, a graph with n vertices has an independent set of size k if and only if the graph has a vertex cover of size n − k. Propose a representation and a neighboring structure for the vertex cover problem. Can we apply the proposed solution to the independent set problem? Propose an incremental way to evaluate the neighbors.

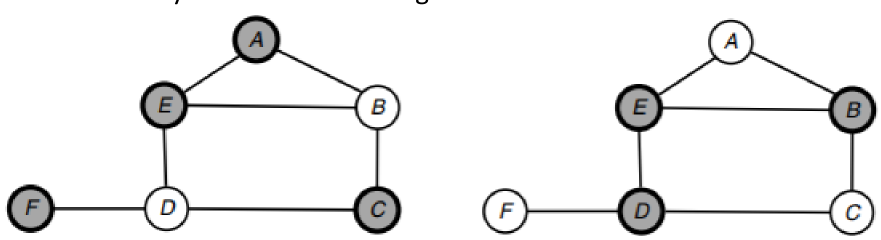


FIGURE 2.45 The vertex cover problem.

Answer:

Propose a representation and a neighboring structure for the vertex cover problem:

Shift one key-vertex into non-key-vertex, meanwhile shifting its adjacent vertices which are also non-key-vertices into key-vertices to maintain the legality of the solution. More precisely, refer to $V_r(v) = \{u \mid (u, v) \in E$ and $u \notin V'\}$, where E is the set of edges and V' is the set of all the key-vertices. The neighborhood move MV(v) is defined to remove a key-vertex v from V' and add all the associated vertices in $V_r(v)$ into V'.[9]

Can we apply the proposed solution to the independent set problem?

Yes. Because of the bi-directional implication relationship of the two problems that is outlined in the problem statement.

Propose an incremental way to evaluate the neighbors:

See lines 7-16:

```
Input: Undirected graph G(V, E), N_start, α
Output: The best solution V'_gbest found so far
iter ← 0
while iter < N_start do
   Generate initial solution V' /* See Section 3.2 */
   no_improve ← 0
   while no_improve < α do
      V'_lbest ← TabuSearch(V') /* See Section 3.3* /
      if V'_lbest is better than V'_gbest then
         V'_gbest ← V'_lbest
         no_improve ← 0
      else
         no_improve ← no_improve + 1
      end if
      V' ← Perturbation(V'_lbest)  /* See Section 3.4 */
   end while
   iter ← iter + 1
end while
```

b) *(10pts) How would you use the neighborhood of question a) to solve the VCP using Simulated Annealing? (brief discussion of algorithm via Talbi and class notes)*

Answer:
Yes. Start by using the algorithm segment from class lecture below and apply the neighborhood of problem a.

```
while k < kmax and e > emax          // While time remains & not good enough:
   sn := neighbor(s)                  // Pick some neighbor.
   en := E(sn)                        // Compute its energy.
   if en < eb then                    // Is this a new best?
      sb := sn; eb := en ; s = sn?          // Yes, save it.
```

Based on **iterative improvement and threshold value**, at each iteration of the algorithm, a new point is randomly generated. The distance of the new point from the current point, or the extent of the search, is based on a probability distribution with a scale proportional to the temperature. The algorithm accepts all new points that lower the objective, but also, with a certain probability, points that raise the objective. By accepting points that raise the objective, the algorithm avoids being trapped in local minima, and can explore globally for more possible solutions. An annealing schedule is selected to systematically decrease the temperature as the algorithm proceeds. As the temperature decreases, the algorithm reduces the extent of its search to converge to a minimum. [4]

Cooling method and stopping criteria applied are important considerations:
Initial Temperature (Talbi, p 130; Yang, p183)
- Accept most all points in neighborhood (high T)
- Accept only a few points (acceptance deviation)
- Accept more than a few (acceptance ratio)

Equilibrium State Transitions: a priori static vs. adaptive number of transitions
Cooling: Linear vs. nonlinear (geometric, logarithmic, very slow, nonmonotomic, adaptive, etc.)
Stopping Criteria:
- Reaching a low Temp
- Number of iterations without improvement
- Max number of function evaluations

Beighle, John

c) *(10pts) Talbi #2.19 “VRP - Tabu”*

2.19 Tabu list representation. Suppose we must solve the vehicle routing problem that has been defined in Example 1.11. The transformation operator used consists in moving a customer c_k from route R_i to route R_j. The tabu list is represented by the move’s attributes. Propose three representations that are increasing in terms of their severity (space complexity).

Answer:
The search strategy of tabu search incorporates short-term memory which remembers and forbids to revert moves that have been previously made. Parametric adjustments to 2 & 3 below cause have potentially significant space severity implications.

1. Introduce the transformation outlined above moving customers.
2. Introduce *tabu tenure* parameter governs the time a certain move should not be undone.
3. Introduce *aspiration tenure* parameter, instead of picking the best move in each iteration, the algorithm picks a move it has not made for a certain time.

d) *(15pts) Describe an algorithm possibility for your project or MIS, or SCP to be a local Tabu search. Relate to fitness landscape.*

Answer:
TSP is a component of my Project+ (if it’s approved). However, my project is attempting Dynamic TSP. It seems the following algorithms follow a very interesting progression and may come in handy in the project for both deterministic and stochastic algorithm application. The algorithms are explained in detail in: *A Neighborhood Expansion Tabu Search Algorithm Based On Genetic Factors*, by D. Wang et al.

Fitness Landscape: The Tabu search, by itself, only searches in a designated adjacent neighborhood space. Thus, speed is an advantage, but it is easily trapped into local optimization. So, it is an important step to master the neighborhood space for **obtaining the target**. Moreover, the traditional scatter search strategy of tabu search will jump to the sub-optimal solution’s search space when current optimal solution keep certain search times, which waste a lot of time and reduce the scatter search ability. Tabu search has only one initial solution in the search process. It has a strong dependence on the initial solution. A good initial solution might result in a good convergence, a poor initial solution may not. **The way this research combines to benefits of GA with TS it addresses the weaknesses inherent in TS by itself.**

Stopping criteria: 20 iterations per dataset.
Did Authors follow essence of Barr using graphs, charts, pseudocode, etc. effectively? YES. The research could be easily re-create.

[continued on next page…]

Beighle, John

The Tabu TSP Algorithm:

Begin
  Set the parameters; $i = 1$;
  Initialize a path at random
  clear up the tabu list
  while condition is not satisfied **Do**
    Calculate the so_far_bestpath's neighborhood;
    Bestpath; so_far_bestpath;
    update tabu list;
    $i = i + 1$;
End

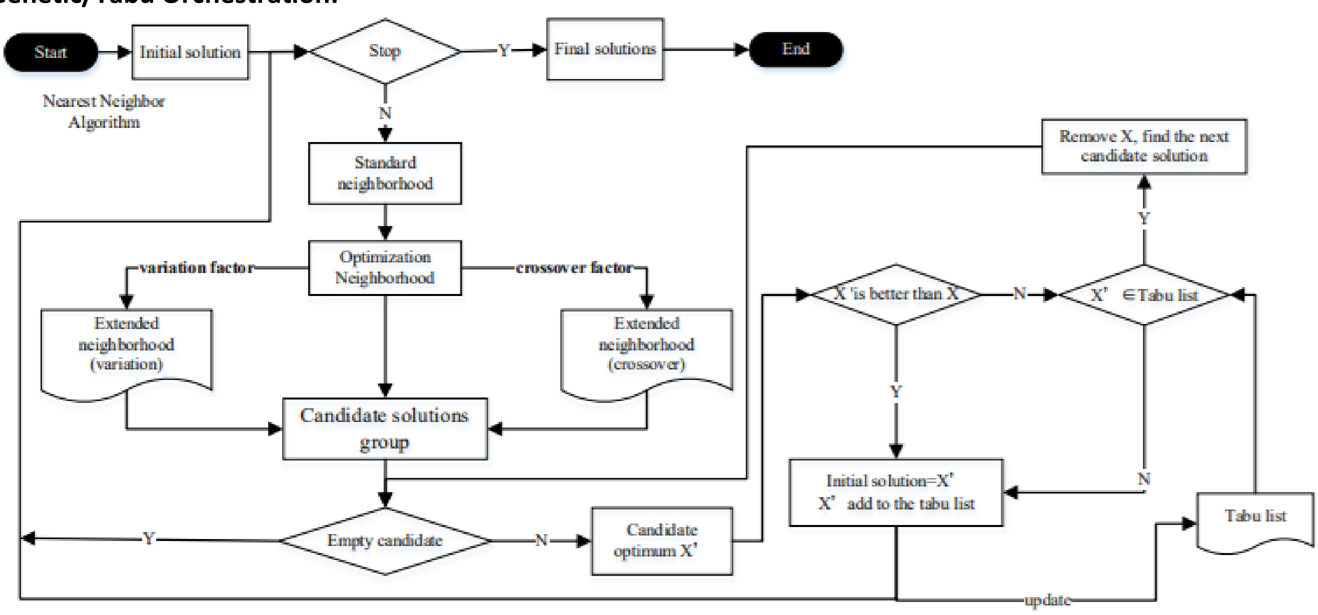
The Genetic Algorithm:

Begin
  Set the parameters; $i = 1$;
  Initialize the population (*pop*);
    while condition is not satisfied **Do**
      Calculate the length of each solution;
      shortPath;
      pop = selection (*pop*);
      pop = crossover (*pop*);
      pop = mutation (*pop*);
      $i = i + 1$
End

A neighborhood Expansion Tabu Search Algorithm Based on Genetic Factors:

Begin
  Calculate the so_far_bestpath's neighborhood;
  Sort and Prefer the so_far_bestpath's neighborhood;
  pop = prefer neighborhood;
  *pop*1 = crossover (*pop*);
  *pop*2 = mutation (*pop*);
  neighborhood = *pop* + *pop*1 + *pop*2;
End

Genetic/Tabu Orchestration:



Beighle, John

References:

1. Vertex Cover: Wikpedia
2. Design & Analysis of Algorithms. Carnegie Mellon University Lecture Series. 2020
3. OpenDSA Data Structures and Algorithms Modules Collection: Chapter 28
4. Metaheuristics: From Design to Implementation, El-Ghazali Talbi, 2009
5. A Tabu Search Heuristic for the Vehicle Routing Problem, M. Gendreau, A. Hertz, G. Laporte, 94
6. A Strategy for Reducing the Computational Complexity of Local Search-Based Methods, and its Application to the Vehicle Routing Problem E. Zachariadis, C. Kiranoudis
7. A Comparative Study of Tabu Search and Simulated Annealing for Traveling Salesman Problem, Sachin Jayaswal
8. A Neighborhood Expansion Tabu Search Algorithm Based On Genetic Factors Dan Wang, Haitao Xiong, Deying Fang, 2016
9. Multi-start Iterated Tabu Search for the MinimumWeight Vertex Cover Problem, T. Zhou, Z. Lu, Y. Wang, J. Ding, B. Peng