GeoInformatica, , 1–11 ()

TABU SEARCH HEURISTIC FOR POINT-FEATURE CARTOGRAPHIC LABEL PLACEMENT

MISSAE YAMAMOTO, GILBERTO CAMARA AND LUIZ ANTONIO NOGUEIRA LORENA

missae@dpi.inpe.br, gilberto@dpi.inpe.br and lorena@lac.inpe.br
Brazilian Institute of Space Research (INPE), SP, BR

Editor:

Abstract. The generation of better label placement configurations in maps is a problem that comes up in automated cartographic production. The objective of a good label placement is to display the geographic position of the features with their corresponding label in a clear and harmonious fashion, following accepted cartographic conventions. In this work, we have approached this problem from a combinatorial optimization point of view, and our research consisted of the evaluation of the Tabu Search (TS) heuristic applied to cartographic label placement. When compared, in real and random test cases, with techniques such as simulated annealing and genetic algorithm (GA), TS has proven to be an efficient choice, with the best performance in quality. We concluded that TS is a recommended method to solve cartographic label placement problem of point features, due to its simplicity, practicality, efficiency and good performance along with its ability to generate quality solutions in acceptable computational time.

Keywords: tabu search, label placement, heuristic, metaheuristic

1. Introduction

Cartographic label placement refers to the label insertion process in maps and is one of the most challenging problems in geoprocessing and automated cartography. Positioning the labels requires that overlap among labels be avoided, that cartographic conventions and preference be obeyed, that unambiguous association be achieved between each label and its corresponding feature and that a high level of harmony and quality be achieved.

There are three different label-placement problems: labeling of point features (cities, schools, hospital, mountain peaks ...), line features (rivers, roads, ...), and area features (countries, states, oceans, ...). In this article we are concerned with the placement of labels for point features, approaching the problem from a combinatorial optimization viewpoint. This approach requires the precise definition of the associated concepts of potential label positions, cartographic preference and objective function:

- Potential label positions for each point feature. This concept indicates the set of positions which will be considered as candidates. Figure 1 shows a set of four potential label positions for a point feature and each box indicates a region in which a label may be placed.

- Cartographic preference concerns a preference assignment to the placement of a label for a point. The value inside each box from Figure 1 corresponds to the order of preference for placing a label. More desirable positions are indicated by lower values.
- The objective function (F) is a function to be optimized that measures the quality of the label placement distinguishing good elements between potential label positions. In general the quality of labeling depends on the number of overlaps between labels and the cartographic preference for placing a label.

Figure 1. A set of potential label positions and their cartographic preference(best=0.0; worse=0.9)

If we indicate by "npos" the number of potential label positions and by "np" the total number of point features, there are $npos^{np}$ possible configurations, a number which increases exponentially. Since the set of possible solutions is finite, theoretically we could select the best solution by enumeration, but as the number of points increases this becomes unfeasible, because of the combinatorial explosion of possible solutions. Marks and Shieber [12] have shown that the point feature label placement (PFLP) problem is NP-hard. Therefore, we need heuristics and metaheuristics that seek a compromise solution in cost terms.

Several heuristics and metaheuristics have been used to solve the PFLP problem, such as exhaustive search, greedy algorithms, discrete gradient descent, Hirsch's algorithm [9], Lagrangean relaxation [16, 17], Simulated Annealing [3, 2, 13] and others. They are reviewed by [3]. A GA with mask is described in [14].

In [15], we have reviewed the existing algorithms and proposed a new technique to solve the PFLP problem, based on TS metaheuristic. Our results indicate that the TS algorithm has better results in label placement quality than other methods. Given its relative simplicity of implementation and efficient performance, we believe that TS method is a good solution to the PFLP problem.

The rest of the article is described as follows. In Section 2 we introduce the TS optimization algorithm to solve the PFLP problem. In Section 3 we show an example of a TS application for 6 point features. In Section 4 we present and discuss the results obtained on real data and on a standard set of randomly generated points suggested in the literature [3, 14]. The comparison of the TS algorithm with the other techniques is presented in this section. Our conclusions are presented in Section 5.

2. TABU SEARCH FOR PFLP

TS is a heuristic procedure proposed by Fred Glover to solve combinatorial optimization problems. The basic idea is avoiding that the search for best solutions stops when a local optimum is found [4, 5, 6, 11, 7, 8]. In order to avoid convergence to a local minimum, the TS algorithm maintains a list of points where a label position has been changed ("tabu list") which are considered to be non-acceptable

(hence the use of the term taboo). The algorithm makes a search of the solution space, aiming at getting the best alternative that is not considered taboo. The TS heuristic also includes special provision for solutions in the tabu list to be considered as alternatives, based on an aspiration criterion that determines when tabu restrictions can be overridden. The following outline describes the TS algorithm for PFLP:

1. Pre-compute all potential overlappings between label positions, recording for each potential label a list of which points and label positions overlap it.
2. Generate an initial configuration, labeling each point with its best cartographic preference position.
3. Repeat the following steps, until we reach a solution without overlappings or until a pre-specified limit of iterations is reached:
   (a) Create a initial candidate list for this iteration.
   (b) Recalculate the candidate list to find the minimum cost label positions for each point referenced in the candidate list.
   (c) Choose the best candidate from the list, based on the cost, taking into account the tabu list and the aspiration criterion.
   (d) Perform the configuration change, designating the solution obtained as the new current solution. Each configuration change consists in modifying the label position of one location.
   (e) Update the tabu list.

The proposed TS algorithm applied to the PFLP problem involves six components: F, tabu list, candidate list, configuration changes, aspiration criteria and long term memory. They are described in the following.

2.1. Objective function

The TS algorithm used here is entirely deterministic and selects the best allowable candidates. Therefore it is necessary to examine and to compare candidates, bringing about a large amount of computation, specially when the number of the points is large. Then the best F is the one whose cost can be easily calculated, making the search efficient and at the same time obtaining quality solutions. The minimization objective function used is

$$\sum_{i=1}^{np} C(i)$$

where,

np = number of points;

C(i) = cost of each point "i", defined by

$$C(i) = \alpha_1 \; overlap(i) + \alpha_2 \; preference(i)$$

and,

overlap(i) = number of overlappings for the label associated with point i (Figure 2). This number refers both to overlappings with other labels and point symbols.

preference(i) = cartographic preference of the active label in the point i and overlapping labels with it;

α_1 = weight for overlapping labels;

α_2 = weight for cartographic preference.

The parameters α_1 and α_2 are manipulated by the user that can choose what is more important, non-overlapping or cartographic quality. If $\alpha_2 = 0$, cartographic preference is not considered.

Figure 2. Conflict evaluation

2.2. Tabu list

The tabu list is an essential component of the algorithm, and stores points where the last label position has been changed.

In [15], we have used a dynamically-sized list, because the PFLP problem needs a large tabu list at the beginning to avoid focusing conflict solutions only at certain regions of the map. Therefore, as the number of overlapping labels decrease, the tabu list can be reduced, as the search will be conducted in smaller regions of the map, for the final adjustments to be made.

The tabu list size used was 7 + INT(0.25 * number of labels that overlap). After some iterations the number of labels that overlap decreases and consequently the tabu list size. The weight 0.25 as well as the tabu list size recalculation after each 50 consecutive iterations were established after tests and experiments made in configurations with 100, 250, 500, 750 and 1000 points (details in [15]).

2.3. Candidate list

The candidate list will be composed of the (point, label, cost) triples that have higher individual costs in the current configuration. The costs associated to each point are calculated as described in Section 2.6. In general (depending on the weights α_1, α_2) higher-cost solutions have a large number of overlaps and their labels are in the least desirable cartographic positions.

The list size is recalculated after 50 consecutive iterations using the expression: 1 + INT (0.05 * number of labels that overlap). The weight 0.05 was chosen after tests in 9 different configurations of 1000 points. The average of labels without overlapping from 9 different configurations for weights 0.03, 0.04, 0.05, 0.06 and 0.07, has shown that the weight 0.05 produces better results (details in [15]).

2.4. Configuration changes

To generate a configuration change, all triples in the candidate list are used to search a best position to label. After the changes in label positions, the candidate

which provides the smallest individual cost is chosen. Solutions generated for a point that is part of the tabu list are discarded, and the next best alternative is selected.

2.5. Aspiration criteria

In some situations, it is necessary to consider alternatives that are part of the tabu list. In such cases, an aspiration criterion is used to override the taboo restriction in two cases:

- A solution is selected if it has a lower value of F than the best solution obtained so far.
- If all candidate solutions are part of the tabu list and fail to meet the above criterion, then the candidate with higher permanency time in the tabu list is chosen.

2.6. Long term memory

Very often, the costs of different solutions are equal, resulting in the same points being entered into the candidate list. Therefore, there is a need to diversify the search. We used a frequency-based memory strategy that counts the number of times that a point has changed its label position and after 50 consecutive iterations divides the accumulated value for each point by the maximum value, obtaining a normalized frequency. This information is used to apply penalties for points that did not bring improvement. The cost $C(i)$ of each point "i" was therefore modified to the individual costs:

$CN(i) = C(i) - normalized\ frequency(i)$

where the normalized frequency(i) is an instrument for search diversification.

3. Example of a TS application for 6 point features

In order to clarify the use of the TS algorithm, this section describes in detail its use. We consider an initial configuration of 6 points features, as illustrated in Figure 3. Each point has four potential label positions (L0, L1, L2 and L3) and each label position has an associated cost (0.0, 0.4, 0.6 and 0.9), where 0.0 indicates the best position and 0.9, the worst (Figure 1).

In the initial configuration there are 5 labels overlapping. The point features and their labels are: P0 = Youngstown (10 characters), P1 = Yankton (7 characters), P2 = Yakima (6 characters), P3 = Worcester (9 characters), P4 = Wisconsin Dells (15 characters) and P5 = Winston-Salem (13 characters). Winston-Salem is the only one without overlapping.

Figure 3. initial configuration

We consider for the example, equal overlapping and cartography preference, so we use $\alpha 1 = \alpha 2 = 1$. The expression 2+INT(0.25 * number of labels that overlap) is used to compute tabu list size, but if the list size is greater than 4 its becomes equal to 4. The candidate list size adopted is 2+INT(0.05 * number of labels that overlap) and they are recalculated at each five iterations. In the following the algorithm to solve conflicts between labels is presented step by step.

Initial state: tabu list size = 4; candidate list size = 2; solution = {(P0, L0, 1.0), (P1, L0, 3.0), (P2, L0, 3.0), (P3, L0, 2.0), (P4, L0, 3.0), (P5, L0, 0.0)}; tabu list = { }; F = 12.0; best solution = 12.0 and labels that overlap = 5

Iteration 1: candidate list = {(P1, L0, 3.0), (P2, L0, 3.0)}; recalculated candidate list = {(P1, L1, 2.4), (P2, L1, 3.4)}; chosen candidate = (P1, L1, 2.4); solution = {(P0, L0, 1.4), (P1, L1, 2.4), (P2, L0, 2.0), (P3, L0, 2.0), (P4, L0, 3.4), (P5, L0, 0.0)}; tabu list = {P1}; F = 11.2; best solution = 11.2 and labels that overlap = 5.

The candidate list is composed at start with the triples (P1, L0, 3.0) e (P2, L0, 3.0), the two first higher-cost cases. Next, the four potential label position of point P1 are examined and the label position with the lowest cost is chosen. The same procedure is repeated for point P2. In the example, the label position L1 with the cost 2.4 is chosen for point P1 and label position L1 with cost 3.4 is selected for point P2.

The candidate (P1, L1, 2.4) was chosen because its cost is lower than the candidate (P2, L1, 3.4) and P1 is not in the tabu list. Applying the same steps iteratively, we will obtain the following results:

Iteration 2: candidate list = {(P4, L0, 3.4), (P1, L1, 2.4)}; recalculated candidate list = {(P4, L2, 1.6), (P1, L2, 2.6)}; chosen candidate = (P4, L2, 1.6); solution = {(P0, L0, 1.4), (P1, L1, 1.4), (P2, L0, 1.0), (P3, L0, 1.0), (P4, L2, 1.6), (P5, L0, 1.6)}; tabu list = {P4, P1 }; F = 8.0; best solution = 8.0 and labels that overlap = 6.

Iteration 3: candidate list = {(P5, L0, 1.6), (P4, L2, 1.6)}; recalculated candidate list = {(P5, L2, 0.6), (P4, L1, 1.8)}; chosen candidate = (P5, L2, 0.6); solution = {(P0, L0, 1.4), (P1, L1, 1.4), (P2, L0, 1.0), (P3, L0, 1.0), (P4, L2, 0.6), (P5, L2, 0.6)}; tabu list = {P5, P4, P1}; F = 6.0; best solution = 6.0 and labels that overlap = 4.

Iteration 4: candidate list = {(P0, L0, 1.4), (P1, L1, 1.4)}; recalculated candidate list = {(P0, L1, 1.8), (P1, L0, 2.0)}; chosen candidate = (P0, L1, 1.8); solution = {(P0, L1, 1.8), (P1, L1, 1.8), (P2, L0, 1.0), (P3, L0, 1.0), (P4, L2, 0.6), (P5, L2, 0.6)}; tabu list = {P0, P5, P4, P1}; F = 6.8; best solution = 6.0 and labels that overlap = 4.

Iteration 5: candidate list = {(P1, L1, 1.8), (P0, L1, 1.8)}; recalculated candidate list = {(P1, L0, 1.0), (P0, L0, 1.4)}; chosen candidate = (P1, L0, 1.0); solution = {(P0, L1, 0.4), (P1, L0, 1.0), (P2, L0, 2.0), (P3, L0, 1.0), (P4, L2, 0.6), (P5, L2, 0.6)}; tabu list = {P1, P0, P5, P4}; F = 5.6; best solution = 5.6 and labels that overlap = 3.

The points P0 and P1 are in the tabu list, but the candidate (P1, L0, 1.0) was chosen, because the F = 5.6 of the new configuration is better than the best solution = 6.0 reached.

Iteration 6: candidate list = {(P1, L0, 1.0), (P2, L0, 2.0)}; recalculated candidate list = {(P1, L1, 1.8), (P2, L3, 1.9)}; chosen candidate = (P2, L3, 1.9); solution = {(P0, L1, 0.4), (P1, L0, 0.0), (P2, L3, 1.9), (P3, L0, 1.9), (P4, L2, 0.6), (P5, L2, 0.6)}; tabu list = {P2, P1, P0, P5}; F = 5.4; best solution = 5.4 and labels that overlap = 2.

In this iteration we recalculate the tabu list size, candidate list size and the cost of the solution for each point using the normalized frequency: tabu list size = 2; candidate list size = 2; solution = {(P0, L1, 0.01), (P1, L0, 0.0), (P2, L3, 1.44), (P3, L0, 1.9), (P4, L2, 0.14), (P5, L2, 0.14)}; tabu list = {P2, P1}.

The same procedure is executed until we reach a solution without overlapping or until a pre-specified limit of iterations.The Figure 4 shows the result of TS application on the example of 6 points features.

Figure 4. After TS application for example of the Figure 3

4. Results

In order to verify the performance of the TS algorithm in real datasets, we used the dataset available in [10]. The set consists of 128 point features from regions of the USA map. Each label varies in length depending on the name of the city it represents, creating a realistic situation. The area in geographical coordinates is: longitude (W 123^0 0' 0" W 73^0 0' 0"), latitude (N 24^0 0' 0" N 51^0 0' 0") and the projection is LAMBERT/HAYFORD.

We made tests using different values of α_1 (that handle the level of consideration of the number of overlaps) and α_2 (that handle the level of consideration of the cartographic preference). The parameters considered for tests are:

- tabu list size = 7 + INT (0.25 * number of labels that overlap);
- candidate list size = 1 + INT (0.05 * number of labels that overlap);
- number of iterations for recalculations = 50;
- character height of the label = 1.0 mm;
- potential label positions = 8.

The results of the tests are reported on Table 1.

When the cartographic preference weight is larger than the weight of the number of overlappings, the number of labels that overlap is large, but the labels try to occupy the best cartographic positions. On the other hand, as cartographic preference decreases and the weight for number of overlappings increases, the number of labels that overlap decreases, but it is possible to see that the labels occupy any of the potential positions. Figure 5 presents a zoom of the crowded area of the example layout. Results of the tests for different character heights and different scale are in [15].

Table 1. RESULTS FOR DIFFERENT $\alpha 1$ AND $\alpha 2$

layout	**USA1**	**USA2**	**USA3**	**USA4**
$\alpha 1$	1	1	1	3
$\alpha 2$	10	5	1	1
number of iterations	324	140	28	28
number of labels that overlap	26	4	0	0

Christensen et al. [3] and Verner et al.[14] compared several algorithms using standard sets of randomly generated points: grid size of 792 by 612 units, fixed size label of 30 by 7 units and page size of 11 by 8.5 inch. In order to compare the TS algorithm with previous work, we used the standard sets of randomly generated points and simulated the same conditions as described by [3] and followed the same assumptions as [14].

- Number of the points: n = 100, 250, 500, 750, 1000;
- For each problem size, we generated 25 different configurations with random placement of point feature using different seeds;
- For each problem size, we calculated the average percentage of labels placed without conflict of the 25 trials;
- No penalty was attributed for labels that extended beyond the boundary of the region;
- 4 potential label positions were considered;
- cartographic preference was not taken into account;
- No point selection was allowed (i.e., no points are removed even if avoiding superposition is inevitable);
- the parameters used for TS are:
  - tabu list size = 7 + INT (0.25 * number of labels that overlap);
  - candidate list size = 1 + INT (0.05 * number of labels that overlap);
  - number of iterations for recalculation = 50.

Results from our TS application on PFLP (average over 25 trials) are recorded in (Table 2), as follows:

- iteration: number of iterations to reach a state without conflicts or a state of smallest conflicts among labels, limited to the maximum number of iterations. (50 for 100 points, 100 for 250 points, 8000 for 500 points, 15000 for 750 points and 30000 for 1000 points);

- Overlapping labels: labels placed with conflicts;
- Time(sec.): processing time to get state without conflicts or the state of smallest conflict among labels.
- Results(minimum): the minimum quantity of labels placed without conflict.
- Results(maximum): the maximum quantity of labels placed without conflict.
- Results(standard deviation): the standard deviation of labels placed without conflict.
- Results(average): the average of labels placed without conflict.
- Results(%): the percentage of labels placed without conflict.

Table 2. RESULTS FROM TS ALGORITHM USING THE STANDARD DATASETS

Number of points	**100 points**	**250 points**	**500 points**	**750 points**	**1000 points**
number of iterations	7	45	450	6645	21410
overlapping labels(average)	0.0	0.0	3.6	24.28	99.96
time (sec.)	4	28	114	245	1179
Results(minimum)	100.00	250.00	492.00	715.00	871.00
Results(maximum)	100.00	250.00	500.00	735.00	923.00
Results(standard deviation)	00.00	00.00	2.29	6.67	15.61
Results(average)	100.00	250.00	496.40	725.72	900.04
Results(%)	100.00	100.00	99.26	96.76	90.00

Figure 6 shows a initial label placement for 1000 points and Figure 7 shows a layout after using our TS algorithm.

Regarding the optimization algorithms of the literature, the tabu search showed superior results in quality of label placement. Table 3 shows the percentage of labels placed without conflict for 100, 250, 500, 750 and 1000 points, considering different algorithms of the literature. The lines show the percentage of labels placed without conflict by the optimization algorithms tested in [3] (random placement, greedy-depth first, gradient descent, 2-opt gradient descent, 3-opt gradient descent, Hirsch, Zoraster and simulated annealing), in [14] (GA without masking and GA with masking) and the TS.

The average processing times spent by the GA with masking [14], to solve the PFLP problem for 100, 250, 500, 750 and 1000 points, were 6, 49, 414, 1637 and 7256 seconds respectively, using a Sun-Sparc 10 workstation. TS solved similar problems in 4, 28, 114, 245 and 1179 seconds. Our implementation used a Sun Sparc 20 workstation, UNIX solaris version 2.5, C++ compiler version 4.0.1, and is part of the Map Production module (SCARTA) of the SPRING GIS environment [1].

Table 3. COMPARISON OF PFLP ALGORITHMS USING STANDARD DATASETS (percentage of labels placed without conflict)

algorithms	**100 points**	**250 points**	**500 points**	**750 points**	**1000 points**
Tabu Search	100.00	100.00	99.26	96.76	90.00
GA with masking	100.00	99.98	98.79	95.99	88.96
GA without masking	100.00	98.40	92.59	82.38	65.70
Simulated Annealing	100.00	99.90	98.30	92.30	82.09
Zoraster	100.00	99.79	96.21	79.78	53.06
Hirsch	100.00	99.58	95.70	82.04	60.24
3-Opt Gradient Descent	100.00	99.76	97.34	89.44	77.83
2-Opt Gradient Descent	100.00	99.36	95.62	85.60	73.37
Gradient Descent	98.64	95.47	86.46	72.40	58.29
Greedy-depth First	95.12	88.82	75.15	58.57	43.41
Random Placement	84.56	65.63	44.06	29.06	19.53

5. Conclusion

The point feature label placement (PFLP) is a problem of practical importance for geoprocessing and automated cartography. Our work has proposed and evaluated a TS optimization algorithm applied to the PFLP problem. TS algorithm performance was evaluated with real datasets and using a standard test set described in the literature.

In real datasets, issues that influence performance include clustering of point features, paper size and scale, variation in label length and character height. We also tested the impact of cartographic preference choices, allowing different compromises between label placement and the number of overlappings. The results have shown that, as the importance of cartographic preference increases, the algorithm takes longer to reach a non-conflict state. Our tests with real datasets have indicated that the TS method always produces maps with desirable cartographic quality with an acceptable timing, even when applied to situations with natural clusters of the point feature distributions and with labels of variable length.

By using a standard set of randomly generated points and the same conditions described by [3] and [14], we have shown that TS has better results in label placement quality than other methods published in the literature. TS also allows the user to establish his choice of compromise between non-overlapping labels and cartographic prefence. Therefore, TS can be recommended to solve the cartographic label placement problem for point features, due to the quality of its label placement and to its flexibility.

References

1. G. Câmara, R.C.M. Souza, U.M. Freitas, and J.C.P. Garrido. Spring integrating remote sensing and gis with object-oriented data modelling. *Computers and Graphics*, 15:395–403, 1996.
2. J. Christensen, J. Marks, and S. Shieber. *Graphics Gems IV*, chapter Placing Text Labels on Maps and Diagrams. London, Academic Press, Cambridge, Mass., 1994.
3. J. Christensen, J. Marks, and S. Shieber. An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics*, 14:203–232, 1995.
4. F. Glover. Tabu search - part i. *ORSA Journal on Computing*, 1:190–206, 1989a.
5. F. Glover. Tabu search - part ii. *ORSA Journal on Computing*, 2:4–32, 1989b.
6. F. Glover. Tabu search - a tutorial. *Interfaces*, 20:74–94, 1990.
7. F. Glover and M. Laguna. *Modern heuristic techniques for combinatorial problems*, chapter Tabu search. McGraw-Hill, New York, 1995.
8. F. Glover and M. Laguna. *Tabu Search*. Kluwer Publishers, Boston, 1997.
9. S. A. Hirsch. An algorithm for automatic name placement around point data. *American Cartographer*, 9:5–17, 1982.
10. D. E. Knuth. *The stanford graphBase, a platform for combinatorial computing*. Addison-Wesley, New York, 1993.
11. M. Laguna. A guide to implementing tabu search. *Investigacin Operativa*, 4:5–25, 1994.
12. J. Marks and S. Shieber. The computational complexity of cartographic label placement. Technical report, Center for Research in Computing Technology, Harvard University, 1993.
13. E. Shawn, J. Christensen, J. Marks, and S. Shieber. A general cartographic labeling algorithm. *Cartographica*, 33:111–999, 1997.
14. O. V. Verner, R. L. Wainwright, and D. A. Schoenefeld. Placing text labels on maps and diagrams using genetic algorithms with masking. *INFORMS Journal on Computing*, 9:266–275, 1997.
15. M. Yamamoto. Tabu search application for point features cartographic label placement problem. Master's thesis, Applied computing - Brazilian Institute of Space Research (INPE), São José dos Campos, SP, Brazil, 1998. (in portuguese).
16. S. Zoraster. Integer programming applied to the map label placement problem. *Cartographica*, 23:16–27, 1986.
17. S. Zoraster. The solution of large 0-1 integer programming problems encountered in automated cartography. *Operations Research*, 38:752–759, 1990.

Figure 5. Zoom of the crowded area of the layout from example

Figure 6. Before TS application for 1000 random points (number of labels that overlap = 703)

Figure 7. After TS application for 1000 random points (number of labels that overlap = 77)