CS430: Dynamic Programming for Sensor Resource Management

HuaXia Wang, Jieshu Zhang, Yibin Qiu
Department of Computer Science
Illinois Institute of Technology

***Abstract*- This paper describes a dynamic programming based approach to solve simple problem of sensor resource management.**

I. Introduction

Modern sensor suites contain sensors that are agile, allowing fast redirection (e.g., electronic scanning) and fast switching between modes (e.g., radar modes such as SAR and MTI), giving a large number of control options. However, allocation of sensor resources is also subject to a variety of constraints. Developing sensor control algorithms that satisfy such constraints and maximize the value of information obtained is the goal of Sensor Resource Management (SRM) research. [1]

Specifically, we formulize the SRM problem as a simple one. Given the plane environment with multiple targets, we aim to minimize the total cost of obtained sensors by developing a sensor control algorithm that satisfies the constraint.

II. Problem Statement

A. Problem Define

Suppose a horizontal strip L in the plane is modeled as the highway in the plane. An infinite set T of targets, noted as T_0, T_1, …, T_i, …, T_n, are located on L, and an infinite set S of wireless sensors, noted as S_0, S_1, …, S_j, …, S_m, are located outside L. For each target $T_i \in T$, the location of such target is represented as (x_i, y_i). For each sensor $S_j \in S$, the location of such sensor is represented as (x_j, y_j). A sensor S_j can monitor a target T_i if and only if the Euclidean distance d_E between S_j and T_i is at most one. That is $d_E = \sqrt{(x_j-x_i)^2+(y_j-y_i)^2} \wedge d_E \leqslant 1$. Suppose that for each sensor $S_j \in S$, it has a positive cost $C(S_j)$ and for each target $T_i \in T$, it can be monitored by at least one sensor in S. Consider a subset S' of sensors in S. S' is said to be a cover if each target in T is covered by at least one sensor in S'. The cost of S' is the total costs of the sensors in S'. The objective is to find a cover S' whose cost is minimized.

B. Define Sub-problems and Recurrence

Assume that n sensors in S are ordered by its cost $C(S_j)$ in ascending order and that m targets in T are ordered by the value of its x-coordinate x_i in ascending order.

The recurrence is:

$$OPT(i) = \text{Min}\{OPT(i-1)+\text{Min}\{C(s(i))\},\ OPT(i-2)+\text{Min}\{C(s \text{ may cover } s(i-1) \text{ and } T_i)\}\}$$

Consider the optimal solution for T_i. Either T_i shares the same sensor with T_{i-1} or finds a new in-range sensor with lowest cost. In former condition, then we have a set of sensors with minimum cost to cover the targets from T_1 through T_{i-2}. Then we find a minimum cost sensor among the ones that are shareable for both T_i and T_{i-1}. That is the distance from such sensor to T_i and to T_{i-1}, respectively, satisfies d_E. Thus the total cost is OPT (i-2)+Min{C(s may cover s(i-1) and T_i)}. In later condition, then we have a set of sensors with minimum cost to cover the targets from T_1 through T_{i-1}. Then we find a new sensor with lowest cost to which the distance from Ti satisfies d_E. Thus the total cost is Min {OPT (i-1)+Min{C(s(i))}.

III. Algorithm Design

As base cases, we define OPT(0) = 0 and OPT(1) = min{C(s)} (s satisfy d_E (T_1, s)). This produces the following algorithm.

```
for i = 1 to n
   if i = 0 then
      return 0
   else if i = 1 then
      find sensor S can cover Ti
      M(1)=min{C(s(1))}
      return M(1)
   else if target i has been covered then
      M(i) = M(i-1)
      return M(i)
   else
      M(i) = Min{OPT(i-1)+Min{C(s(i))}, OPT(i-2)+Min{C(s
may cover s(i-1) and Ti)}}
      return M(i)
   Endif
Endfor
```

IV. Proof of Correctness

The correctness of recurrence is proved as follows.

(1) When n = 1, for the first target, it is easy to find a sensor with minimum cost for T_1 among all of its in-rang sensors. Thus we get Min{C(s(1))}. 's(1)' denotes 'for each in-range sensors of T_1'.

(2) When n = i – 1, for the i – 1th target, OPT(i - 1) = Min{OPT(i - 2)+Min{C(s(i - 1))}, OPT(i-3)+Min{C(s may cover s(i - 2) and T_{i-1})}}.
   - Case1: the target does not share the same sensor with previous target. Then OPT(i - 1) = OPT(i - 2)+Min{C(s(i - 1))}.
   - Case 2: the target shares the same sensor with previous target. Then OPT(i - 1) = OPT(i - 3)+Min{C(s may cover s(i - 2) and T_{i-1})}.

(3) When n = i, for the ith target, OPT(i) is determined in different cases:
   - Case 1: the target does not share the same sensor with previous target. Then OPT(i) = Min{C(s(i))} + OPT(i - 2)+Min{C(s(i - 1))}(based on case 1 in step 2) or OPT(i) = Min{C(s may cover s(i - 1) and T_i)} + OPT(i - 3)+Min{C(s may cover s(i - 2) and T_{i-1})}(based on case 2 in step 2).
   - Case 2: the target shares the same sensor with previous target. Then OPT(i) = Min{C(s may cover s(i-1) and T_i)} + OPT(i - 2).

Generally, the induction follows the recurrence formula. For each iteration, it gets the value of minimum cost of selected sensors S'.

V. Algorithm Analysis

Sorting, as the initialization part of the algorithm, takes n*log(n) time. In the javadoc documentation it is said that the sort method under java.Collections class uses merge sort. Thus the running time of sorting targets is O(n*log(n)) and that of sorting sensors is O(m*log(m)).

The running time of finding in-range sensors for a specific target is bounded by the number of sensors which is O(m).

The computing of getting another sensor may cover the previous targets that covered by the previous sensor can be done in O(mn) time in total. Theoretically, the running time in each iteration is O(mn) and thus O(mn^2) in total. However, it is impossible that one target is covered by all of the sensors in this problem. So the running time of the computing in overall iterations is O(mn).

The computing of finding a minimum cost sensor covering a specific target among a set of possible sensors can be done also in O(mn).

Thus the running time of the algorithm is O(mn).

References

[1] R.B. Washburn, M.K. Schneider, and J.J. Fox, *Stochastic Dynamic Programming Based Approaches to Sensor Resource Management*, 2002.

[2] J. Kleinburg and E. Tardos, "Dynamic Programming", *Algorithm Design*, pp. 251-311, 2006.