Maximizing Utility of Sensor-Mission Assignment with Uncertain Demands

Diego Pizzocaro*, Matthew P. Johnson†, Hosam Rowaihy‡,
Stuart Chalmers§, Alun D. Preece*, Amotz Bar-Noy†, and Thomas La Porta‡
*School of Computer Science, Cardiff University, UK
§Department of Computing Science, University of Aberdeen, UK
†Department of Computer Science, City University of New York, US
‡Department of Computer Science and Engineering, Pennsylvania State University, US

***Abstract*—A sensor network is usually required to support multiple missions to be accomplished simultaneously. Since missions might compete for the exclusive usage of the same sensing resource we need to assign individual sensors to missions. Missions are characterized by an uncertain demand for sensing resource capabilities. We model this assignment problem by introducing the Sensor Utility Maximization (SUM) model. SUM is NP-Complete and is a special case of the well known Generalized Assignment Problem (GAP). We compare a pre-existing algorithm developed for GAP with a new greedy algorithm which appears to offer the best trade-off between quality of solution and computation cost.**[1]

I. Introduction and Motivations

A sensor network in the field is usually required to support multiple sensing tasks or missions to be accomplished simultaneously. Since missions might compete for the exclusive usage of the same sensing resource we need to assign individual sensors to missions. Missions are usually characterized by an uncertain demand for sensing resource capabilities. Consider for example a mission that requires video sensors to identify a target but the weather conditions and visibility range in the field are not exactly known: in this case the required number of sensors and the resolution of their cameras cannot be precisely determined. We can for example specify only the highest resolution of the cameras needed by the mission or the maximum number of video sensors required. If instead two missions require to identify two different targets that are located in nearby regions on the map, then these missions might compete for the exclusive control of a particular video sensor. Indeed the mission to which the video sensor will be assigned might decide to point the camera in a direction that could be completely opposite to where the other mission would require it.

II. Sensor Utility Maximization Model

We model this assignment problem by introducing the Sensor Utility Maximization (SUM) model which can be represented as a complete weighted bipartite graph as shown in Figure 1. Each mission M_j is associated with a priority p_j and with a utility demand d_j, which represents the maximum utility demand that a mission might require. Furthermore each sensor-mission pair is associated with a utility offer e_{ij}.

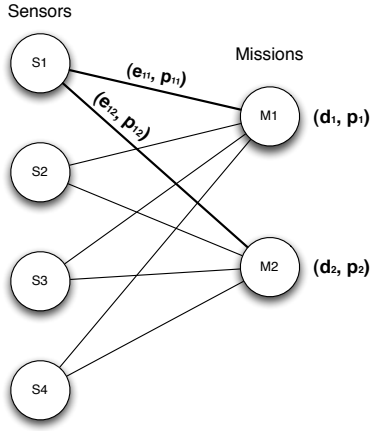

Fig. 1: SUM model as a complete weighted bipartite graph.

We also define the benefit or profit p_{ij} that a sensor S_i can bring to a mission M_j as the fraction of mission's demand that the sensor is able to satisfy, scaled by the priority of the mission: simply stated it holds the following equation $p_{ij} = e_{ij}/d_j \times p_j$.

The goal is to find a sensor assignment that maximizes the total profit, while ensuring that the total utility cumulated by each mission does not exceed its demand d_j.

In the Integer Linear Programming formulation of the model we use one decision variable called x_{ij}, that is 1 if sensor S_i is assigned to mission M_j.

$$\begin{array}{ll} \text{Maximize:} & \sum_{j=1}^{m}\sum_{i=1}^{n} p_{ij}x_{ij}\ ,\ p_{ij} = e_{ij}/d_j \times p_j. \\ \text{Such that:} & \sum_{i=1}^{n} x_{ij}e_{ij} \leq d_j,\ \forall M_j \in M, \\ & \sum_{j=1}^{m} x_{ij} \leq 1,\ \forall S_i \in S, \text{ and} \\ & x_{ij} \in \{0,1\} \end{array}$$

For each mission M_j we require that the sum of the utility received by M_j does not exceed its own max utility demand d_j. This assumption is based on the principle that sensing resources are in high demand and should not be wasted[2].

[1]This research was sponsored by the U.S.Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF- 06-3-0001.

[2]JP 2-01 Joint and National Intelligence Support to Military Operations, http://www.dtic.mil/doctrine/jel/new_pubs/jp2_01print.pdf , pages III-10-11, accessed July 25, 2008.

This constraint highlights also that the SUM model is a generalization of the well known Multiple Knapsack Problem [1] that is NP-Complete. We can therefore conclude that SUM is an NP-Complete problem too.

III. Algorithms

SUM is a special case of the Generalized Assignment Problem (GAP) [2], which groups many knapsack-style problems. We solved SUM using a pre-existing algorithm developed for GAP, and also a new greedy algorithm that we especially designed for SUM.

A. Pre-existing algorithm for GAP

We implemented the approximation algorithm in [2] that, combined with the algorithm in [3], is provably a $(2+\epsilon)$-approximation algorithm. If n is the number of sensors and m is the number of missions, then the time complexity of this algorithm is $O\left(\frac{mn}{\epsilon^2}\right)$, where we have to choose a small ϵ to obtain a solution close to the optimum[3]. This leads to a large multiplicative constant $(1/\epsilon^2)$, that is the reason why the greedy algorithm with a worse time complexity will prove faster in real case scenarios.

B. Ordered Sensor-side Greedy

Algorithm 1 Ordered Sensor-side Greedy

```
1: sort the sensors in order of decreasing max{p_ij}
2: M ← {M_1, ..., M_m}
3: for each sensor S_i do
4:    M_k ← maxarg_{M_j ∈ M}{p_ij} (where p_ij = e_ij/d_j × p_j)
5:    if e_ik + total utility cumulated by M_k ≤ d_k then
6:       assign S_i to M_k
7:       M ← M − {M_k}
8:    end if
9: end for
```

The greedy algorithm that we developed considers sensors one by one and assigns them to the mission to which they can contribute the most. In fact we first sort the sensors in decreasing order of maximum profit offer, i.e. $max\{p_{ij}\}$. Finally starting from the sensor which has the highest maximum profit, the algorithm assigns it to the mission M_k where that sensor is of most use, i.e. to the mission that maximizes p_{ij}.

If n is $O(m \log m)$ then the complexity of this algorithm is $O(nm \log m)$, where "$m \log m$" is given by the fact that we have to find the mission M_j that maximizes p_{ij}. Otherwise if n is not $O(m \log m)$, then the complexity becomes $O(nm \log m + n \log n)$.

IV. Simulation and Conclusion

To evaluate the performance of these algorithms we used the simulation environment implemented in Java that was developed in [4]. In this simulation, the utility e_{ij} of a sensor to a mission is a function of the geographical distance between them: the closer the sensor to the mission, the higher its utility; the utility is max if the sensor and the mission lie at the same location. Sensors and missions are generated in uniformly random locations in a 400m $\times$ 400m field, and when sensors are deployed we ensure that the network is connected. Since the algorithms considered here follow a centralized approach, we create also a base station in the field which represents the node where all the information about the network is collected and where all the computation is performed. Each mission has an exponentially distributed demand and priority, so there will be many missions with low demand and low priority and few missions with high demand and high priority. Figure 2 shows the total profits achieved by each different algorithm as fractions of the upper bound represented by the *optimal fractional solution*, which is the solution to the relaxed linear programming formulation of SUM. Simulation results show that our greedy algorithm appears to offer a good trade-off between quality of solution and computation cost. Indeed, even if the total profit of a solution returned by the greedy algorithm is smaller than the one given by the GAP algorithm, the difference between the two is only 1%; moreover, as we discussed in Section III, the greedy algorithm has a better running time than the GAP algorithm in real scenarios[4].

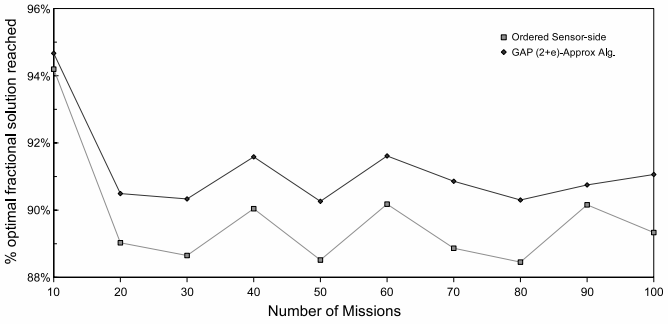

Fig. 2: Ordered Sensor-side and (2+ϵ)-approximation algorithms (500 sensors): percentage of the optimal fractional solution vs missions.

References

[1] S. Khuri, T. Bäck, and J. Heitkötter, "The zero/one multiple knapsack problem and genetic algorithms," *Proceedings of the 1994 ACM symposium on Applied computing*, pp. 188–193, 1994.

[2] R. Cohen, L. Katzir, and D. Raz, "An efficient approximation for the generalized assignment problem," *Inf. Process. Lett.*, vol. 100, no. 4, pp. 162–166, 2006.

[3] O. H. Ibarra and C. E. Kim, "Fast approximation algorithms for the knapsack and sum of subset problems," *J. ACM*, vol. 22, no. 4, pp. 463–468, 1975.

[4] M. Johnson, H. Rowaihy, D. Pizzocaro, A. Bar-Noy, S. Chalmers, T. La Porta, and A. Preece, "Frugal sensor assignment," in *DCOSS '08*.

[5] D. Pizzocaro, M. Johnson, H. Rowaihy, S. Chalmers, A. Preece, A. Bar-Noy, and T. La Porta, "A knapsack approach to sensor-mission assignment with uncertain demands," in *SPIE Europe Security and Defence.* in press, 2008.

[3] We chose $\epsilon = 0.005$ because, in our simulation, this was the smallest value allowable to obtain a reasonable computational time.

[4] Because of space constraints we do not include timing results, which are instead included in the full paper [5].