Sensor Assignment In Virtual Environments Using Constraint Programming*

Diego Pizzocaro, Stuart Chalmers and Alun Preece
University of Aberdeen, Computing Science, Aberdeen, UK
{dpizzocaro,schalmer,apreece}@csd.abdn.ac.uk

Abstract

This paper describes a method for assignment and deployment of sensors in a virtual environment using constraint programming. We extend an existing model (multiple knapsack problem) to implement this assignment and placement, according to a given set of requirements (modelled as a utility extension).

1 Sensor Assignment & Deployment

In military/rescue operations the first step in gathering intelligence is the deployment of sensor devices to acquire knowledge of the domain and information to aid in pre–mission planning[1]. The assignment of sensors to areas where this information could be found, therefore, becomes of vital importance. The optimal assignment of sensors given a pre-defined commanders intent means that we can allocate (possibly limited) resources and use these to the best possible extent. In this paper we describe a method of sensor assignment and deployment using an extension to the multiple knapsack problem and how this can be deployed in a virtual environment as a web service.

In assigning sensors in a virtual environment, we are trying to aid in the placement and best usage of detectors which can scan the field of battle for information to help in the formation of plans or deployment of troops. Generally in such scenarios we have the following resources, requirements and methods: a finite number of sensors with various capabilities, a given set of areas to be covered by various sensor capabilities and a set of methods directing the placement of the sensors (optimality, maximum coverage etc.).

Our main aim in this paper is to describe how we have modelled the assignment of sensors, given these criteria, using a variation on the multiple knapsack model, and how we have formulated the problem in terms of this model.

---

*This work is continuing via participation in the International Technology Alliance funded by the US Army Research Laboratory and UK Ministry of Defence. http://www.usukita.org/

[1] Doctrine for Intelligence Support to Joint Operations: http://www.dtic.mil/doctrine/jel/new_pubs/jp2_0.pdf, Checked on 06/05/2007

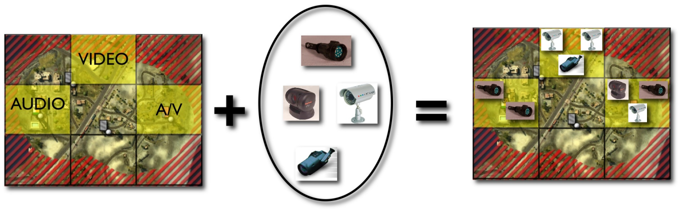

Figure 1: Graphical representation of the Sensor Assignment problem.

Given these criteria, and given the fact that we have utilised the gaming environment Battlefield 2[2] , we made the following assumptions to aid in the construction (and testing) of our model: 1) The areas requiring sensor coverage is modelled as set of pre-defined areas on a given Battlefield 2 map. 2) Each area is assigned a requirement for the capability required in that zone. The requirements on sensors are limited to AUDIO, VIDEO and AUDIO/VIDEO (A/V) capabilities and 3) Three corresponding sensor types are available.

Our model consists of two separate subproblems, both modelled as multiple knapsack: **Sensor Assignment**[3] where we assign sensors to zones, and **Sensor Deployment** where we optimally deploy sensors in their assigned zones. For example, the commander may specify (as in Fig. 1) three areas, each requiring AUDIO, VIDEO and A/V respectively. Thus the problem becomes twofold: Assigning the appropriate sensors to the correct zone, and deploying the sensors to then maximize their coverage of that zone.

1.1 Sensor Assignment as a Multiple Knapsack Problem

The knapsack problem looks at maximizing the number of items placed in a bag (that bag having a maximum weight). Given a set of items, each with a *cost* and a *value*, and a knapsack with a given capacity[4], we have to determine which items to insert in the knapsack so that the *total cost* of the chosen items is less than or equal to the knapsack's capacity while maximizing the *total value* of the chosen items. This is done using an *objective function*, to differentiate between candidate solutions. The *Multiple Knapsack Problem*[5], is a version of the knapsack problem with multiple knapsacks, each with a different capacity.

The assignment problem considers the zones selected by the commander and the information required from these zones. Given a set of zones each with its own information requirement, and a set of sensors each with its own capabilities, we assign each sensor to a zone maximizing the total area covered[6].

For this we extend the multiple knapsack problem to include information about the capabilities of the sensors and the type of information required from

---

[2] http://www.ea.com/official/battlefield/battlefield2/us/ checked 15/06/07

[3] A sensor is idealized as a circular area with a radius r_i, and a center (x_i, y_i).

[4] We generalise this to have weight = cost, and total cost = total weight for a knapsack

[5] We consider a particular type of multiple knapsack problem called 0-1 multiple knapsack

[6] At this stage we do not have the sensor coordinate positions, only the zone assignment

each zone. More formally, we define a two-dimensional variable, x_{ij}:

$$x_{ij} = \begin{cases} 1 & \text{if sensor } i \text{ is in zone } j \\ 0 & \text{otherwise} \end{cases}$$

where for i and j

$\forall i \in \mathbf{N} = \{1, ..., m\}$ i is in the set $\mathbf{N}$ of sensors

$\forall j \in \mathbf{M} = \{1, ..., m\}$ j is in the set $\mathbf{M}$ of zones

We define the constants t_{a_i} and t_{b_i} for each sensor respectively as:

$$t_{a_i} = \begin{cases} 1 & \text{if sensor } i \text{ has AUDIO} \\ 0 & \text{otherwise} \end{cases}$$

and similarly for t_{b_i} and VIDEO. We define w_i = the area covered by the sensor i and c_j = the area of the zone j and subdivide the set of zones into subsets requiring the same type of information thus:

$\mathbf{M_a} = \{j \in M \mid \text{zone j has AUDIO required}\}$

$\mathbf{M_b} = \{j \in M \mid \text{zone j has VIDEO required}\}$

$\mathbf{M_{a,b}} = \{j \in M \mid \text{zone j has AUDIO/VIDEO required}\}$

This allows us to define constraints stating that the area covered by the set of sensors assigned to a zone is less than or equal to the area of the zone (1), and that each sensor is allocated to only one zone (2). We also ensure that there is at least one sensor in each zone selected (3).

$$\sum_{i \in N} w_i \cdot x_{i,j} \leq c_j \qquad \forall j \in M \tag{1}$$

$$\sum_{j \in M} x_{i,j} \leq 1 \qquad \forall i \in N \tag{2}$$

$$\sum_{i \in N} x_{i,j} \geq 1 \qquad \forall j \in M \tag{3}$$

The constraints showing the types of information needed in each zone are the most important part of the model as they extend the basic multiple knapsack. For example, (4) states that only AUDIO sensors are enabled in AUDIO zones[7].

$$\sum_{i \in N} t_{a_i} \cdot x_{i,j} = \sum_{i \in N} x_{i,j} \qquad \forall j \in M_a \tag{4}$$

Finally we define two alternatives that can be applied as the objective function, (5) maximizes the total area covered by the sensors, while (6) minimizes the number of sensors used while maximizing the total area covered.

$$\max \sum_{i \in N} \sum_{j \in M} w_i \cdot x_{i,j} \tag{5}$$

$$\max \sum_{i \in N} \sum_{j \in M} w_i \cdot x_{i,j} - \sum_{j \in M} \sum_{i \in N} x_{i,j} \tag{6}$$

---

[7] We have similar constraints for each sensor type (AUDIO and A/V).

1.2 Sensor Deployment

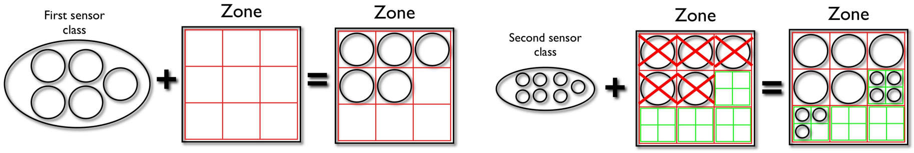

Figure 2: Sensor Deployment Example.

For each set of sensors assigned, we apply the knaspack model, finding an optimal positioning within the given zone. We deploy each sensor so that the areas covered by the sensors do not overlap and that we maximize coverage (Fig. 2). We assume that the length of the side of each zone and the length of the radius of each sensor have to be of power two otherwise could insert a sensor in a zone "out of shape" (Fig. 3). Here we are trying to insert a sensor of area 7 in a zone with a remaining area of ≥ 7. Without this the solver will assign the sensor to that zone, even though we cannot change the shape of the sensor coverage area.

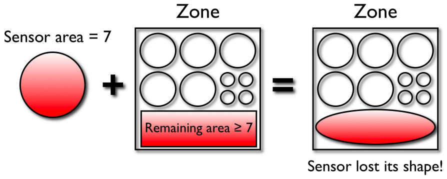

Figure 3: The Sensor Assignment model with no heuristic.

1.3 Extending the Sensor Assignment model

Currently we consider three possible sensor types. If we want to add sensors with other capabilities (e.g. INFRARED), we simply add the relevant constraints and constants to the model (as in Section 1.1). As an example, adding "INFRARED", we first define this constant for each sensor:

$$t_{c_i} = \begin{cases} 1 & \text{if sensor } i \text{ has INFRARED} \\ 0 & \text{otherwise} \end{cases}$$

The types of zones available (in addition to those in section 1.1) now include:

$$\mathbf{M_c} = \{j \in M \mid \text{zone j has INFRARED required}\}$$

$$\mathbf{M_{a,c}} = \{j \in M \mid \text{zone j has AUDIO/INFRARED required}\}$$

$$\mathbf{M_{b,c}} = \{j \in M \mid \text{zone } j \text{ has VIDEO/INFRARED required}\}$$

$$\mathbf{M_{a,b,c}} = \{j \in M \mid \text{zone j has AUDIO/VIDEO/INFRARED required}\}$$

With the addition of the corresponding constraints (7,8,9,10) to the model:

$$\sum_{i \in N} t_{c_i} \cdot x_{i,j} = \sum_{i \in N} x_{i,j} \qquad \forall j \in M_c \tag{7}$$

$$\sum_{i \in N} t_{a_i} \cdot t_{c_i} \cdot x_{i,j} = \sum_{i \in N} x_{i,j} \qquad \forall j \in M_{a,c} \tag{8}$$

$$\sum_{i \in N} t_{b_i} \cdot t_{c_i} \cdot x_{i,j} = \sum_{i \in N} x_{i,j} \qquad \forall j \in M_{b,c} \tag{9}$$

$$\sum_{i \in N} t_{a_i} \cdot t_{b_i} \cdot t_{c_i} \cdot x_{i,j} = \sum_{i \in N} x_{i,j} \qquad \forall j \in M_{a,b,c} \tag{10}$$

From this we can see that it is easy to add other capabilities to the model, by simply adding another constant and the corresponding constraint.

2 Implementation & Testing

Figure 4 shows the system architecture of the sensor assignment system consisting of 3 main component: The problem solver web service[8], the Commanders GUI and the Battlefield 2 server. The solver is written using the CHOCO CSP library[9], with decision variables implemented as two dimensional binary domains and constraints representing the zone properties, the commanders requirements and the objective function. The problem, is NP-hard, so our solving time is exponential[10]. We also developed a *mod* for *Battlefield 2* altering the behaviour of the game server, allowing the deployment created by the solver to be placed in the gaming area.

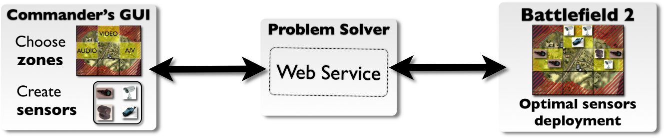

Figure 4: System Architecture

Our baseline tests[11] assigned 15 sensors (5 each of AUDIO, VIDEO and A/V), of radius 64m (AUDIO and VIDEO) and 32m (A/V). The objective was to assign these sensors to 6 zones, all of size 128m with information requirements in each zone: two AUDIO zones, two VIDEO and two A/V. This allocation took

---

[8] Implemented Apache Axis running on an Apache Tomcat server

[9] http://choco.sourceforge.net/ checked on 14/06/07

[10] Section 3 considers relaxing our model to improve the solving speed.

[11] On a Macbook Intel Core Duo 2 GHz, 1GB 667 MHz DDR2 SDRAM with Windows XP

< 20 secs. Decreasing the number of A/V sensors (10 AUDIO, 3 VIDEO and 2 A/V) gave a solving time of < 1 sec. Finally we increased both the sensor radii and zones but kept the same ratio. We had 5 AUDIO(128m), 5 VIDEO (128m) and 5 A/V (64m) sensors with zone size 256m. This, again, took< 20 seconds. We surmised that if the ratio between sensors and zones remains the same then the time taken to assign sensors to zones will be the same.

3 Related & Future work

A number of projects have investigated the use of decision-theoretic approaches to cooperative sensor planning [1], and the use of intelligent cooperative reasoning to select optimal locations during missions. Our sensor assignment can be seen as a *coverage scheme* in the categorisation given in [3]. Similar methods for static sensor coverage can be seen in [2] and [4], where they look not only at the selection of subsets of sensors, but also consider the case of node failure.

The main focus of our work will involve the progression of our knapsack extension for sensor assignment. While the current model has provided a method for optimal sensor placement, we have not yet considered the scenario where all requirements for sensor deployment are not able to be met. Currently we are evaluating a model that will let us relax the problem constraints, finding a solution to a satisfiable subset of the requirements for sensor placement, and developing a better objective function by considering for each item $i : p_i \neq w_i$[12].

References

[1] Diane J. Cook, Piotr Gmytrasiewicz, and Lawrence B. Holder. Decision-theoretic cooperative sensor planning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1013–1023, 1996.

[2] J. Lu, L. Bao, and T. Suda. Coverage-aware sensor engagement in dense sensor networks. *Journal of Embedded Computing (JEC), Special Issue on Embedded System Optimization*, 2007.

[3] H. Rowaihy, S. Eswaran, P. Johnson, T. Brown, A. Barnoy, and T. F. La Porta D. Verma. A survey of sensor selection schemes in wireless sensor networks. In *SPIE Unattended Ground, Sea, and Air Sensor Technologies and Applications IX*, Orlando, FL, 2007.

[4] Kuei-Ping Shih, Yen-Da Chen, Chun-Wei Chiang, and Bo-Jun Liu. A distributed active sensor selection scheme for wireless sensor networks. In *ISCC '06: Proceedings of the 11th IEEE Symposium on Computers and Communications*, pages 923–928, Washington, DC, USA, 2006. IEEE Computer Society.

---

[12] i.e the cost and value for each sensor is not the same (an assumption in section 1.1)