A System Architecture for Decision-making Support on ISR Missions with Stochastic Needs and Profit

Nan Hu[a], Diego Pizzocaro[b], Thomas La Porta[a] and Alun Preece[b]

[a]Department of Computer Science and Engineering, The Penn State University, US;

[b]School of Computer Science and Informatics, Cardiff University, UK

ABSTRACT

In this paper, we propose a system architecture for decision-making support on ISR (i.e., *Intelligence*, *Surveillance*, *Reconnaissance*) missions via optimizing resource allocation. We model a mission as a graph of tasks, each of which often requires exclusive access to some resources. Our system guides users through refinement of their needs through an interactive interface. To maximize the chances of executing new missions, the system searches for pre-existent information collected on the field that best fit the needs. If this search fails, a set of new requests representing users' requirements is considered to maximize the overall benefit constrained by limited resources. If an ISR request cannot be satisfied, feedback is generated to help the commander further refine or adjust their information requests in order to still provide support to the mission. In our work, we model both demands for resources and the importance of the information retrieved realistically in that they are not fully known at the time a mission is submitted and may change overtime during execution. The amount of resources consumed by a mission may not be deterministic; e.g., a mission may last slightly longer or shorter than expected, or more of a resource may be required to complete a task. Furthermore, the benefits received from the mission, which we call profits, may also be non-deterministic; e.g., successfully localizing a vehicle might be more important than expected for accomplishing the entire operation. Therefore, when satisfying ISR requirements we take into account both constraints on the underlying resources and uncertainty of demands and profits.

Keywords: Non-deterministic Resource Allocation, Stochastic Knapsack Problem, Feedback-based ISR Decision-making Support

1. INTRODUCTION

Resource allocation is a fundamental and critical problem studied in different forms in various communities, including military scenarios, in which resources can be classified as many types, such as information (e.g., pictures, videos), platforms (e.g., UAVs, patrols), systems (e.g., cameras, clusters), services (e.g., network bandwidth, storage space), etc., and are often subject to limited capacity. Commanders issue their ISR information needs (which we call ISR missions) and need to make the final decisions on how to execute these missions depending on the availability of necessary resources. However, in most cases, it is neither realistic for the commanders to fully understand the overall picture of available resources, nor to carefully study the plan of resource deployment. Instead, commanders should only issue high-level queries describing their missions such as "detect and track some specific high-valued targets on the crossroad of Street A and B", and expect to know the probability of successful execution of the missions, without worrying about the fine details on how the actual resource allocation is performed.

F. Chen *et al.*[1] defined a mission to be a collection of tasks with temporal and causal relationships, and introduced a system architecture, which exploits the demands of missions for resources and searches for the optimal resource allocation solutions. Each mission instance may have different requirements on resources, and successfully satisfying all requirements of a single mission can generate some profit. In addition, multiple missions are said to be compatible with each other if their combined demands do not exceed the available capacity of requested resources. The objective is to suggest the best subset of compatible missions that produces profit as much as possible. In our paper, we propose an improved system architecture providing more flexible decision-making support on ISR missions via refining users' intent, optimizing resource allocation process, and enhancing the feedback mechanism.

In our system, we apply a new interactive interface, CE-SAM (Controlled English Sensor Assignment to Missions), developed from the work of A. Preece *et al.*[2] and refined in D. Pizzocaro *et al.*,[3] which guides users to 1) build a scalable knowledge base of various functionalities of ISR resources and corresponding ISR requirements they can meet, 2) refine their ISR missions without requiring a technical background in formal query languages or ontology building, and 3) match their missions with proper sets of resources represented in the knowledge base.

Since the capacities of resources are often limited, it is rare that all submitted missions can be fulfilled concurrently, where further adjustment is needed. Moreover, we model both demands for resources and the importance of mission completion realistically in that they are not fully known at the time a mission is submitted and may change overtime during execution, which can be formulated as a variation of stochastic knapsack problem[4,5] (SKP). We developed a heuristic algorithm for this problem, which returns temporal admission decisions along with additional instructive feedback reports, instead of simply returning "accepted" or "denied", to facilitate commanders in understanding current conditions.

For example, suppose that a commander submits the mission "detect and track some specific high-valued targets on the crossroad of Street A and B". To complete this mission, some cameras (either fixed or mounted on other mobile sensing platforms, like UAVs) already deployed by the roadside may be required. However, the ability to detect and track partially depends on the traffic conditions (e.g., big trucks may block the sight of our targets), which causes the actual number of necessary cameras to be unpredictable. Assume that the number of cameras needed follows some known distribution, such as Poisson distribution $\mathcal{P}(10)$, and there are only 10 cameras available at that time. In this case, according to the theory of statistics, our system could inform the commander that the failure probability of his mission is about 42%, and provide some suggestions, such as temporarily preempt 2 more cameras may significantly decrease the failure chance to one half as 21%, as flexible decision-making support if additional adjustments are necessary.

The rest of this paper is organized as follows. Section 2 explains some basic concepts of ISR resource, task and mission, and presents our system architecture. Section 3 discusses in details our stochastic resource allocation problem setting, and shows an example scenario to explain our feedback based decision-making support mechanism. Finally, Section 4 concludes this paper.

2. SYSTEM ARCHITECTURE

In this section, we define some basic concepts of ISR resource, tasks and missions and explain the condition when a mission-resource matching can generate profit. We then illustrate our system architecture including descriptions of components and operations within the system. We describe how the system guides users to compose queries to perform their missions, optimizes the resource allocation plans, and helps users take the best decisions through providing instructive feedback reports.

2.1 Basic Concepts of Resource, Task and Mission

In the scope of this paper, ISR resources essentially can be divided into two groups: information that is requested by the commanders to complete their missions, and other types of resources regard to provide such information. For example, assume that a mission requires the collection of some pictures of an area. We can either directly provide pictures to the commander if they have already been taken by other missions, or allocate some ISR assets like cameras to take the pictures requested. Here both pictures and cameras can be considered as resources, while the difference is that the former is directly what the commander expects, and the latter is utilized to collect the former.

Among all types of resources with different features, only the ones with limited quantity (e.g., the fixed cameras deployed along the North Road) that are not able to be shared by an infinite number of missions (e.g., a UAV sometimes may serve multiple detection missions simultaneously, but the number of paralleled missions has a limit) are taken into account in our resource allocation problem.

In our model, an ISR mission consists of multiple tasks, each of which is required to return some supporting information to the mission. A task is taken as "completed" when the portion of information supposed to be generated by this task has been collected. For a "completed" task, its returning information may be obtained either directly from the outcome of other tasks, or from the output of a bundle of proper ISR assets. Thus, an incomplete task demands a set of resources to execute, and it is said to be satisfied when all requested resources are allocated during the process of the mission.

A mission is expected to produce some profit only when all its tasks are completed. When a mission is admitted, we first complete some of its tasks by match their needs with some pre-existing sets of information, and then allocate a set of resources that enables simultaneous execution of all tasks remaining. This type of over allocation is inefficient because not all tasks use all resources consistently.

However, the advantage of this over-allocating policy is that it increases the probability of achieving a successful completion of admitted missions in case in which some tasks require more resources than expected. The spare resources can, in fact, be used as a backup and scheduled among the active tasks to meet excessive needs or improve performance.

Figure 1 shows an example that illustrates how missions and resources are matched. Two missions M_1 and M_2 are represented as diamonds, while subordinate tasks and different types of resources are indicated by circles and squares, respectively. Triangles represent pre-existing information sets, among which those that are shadowed are unavailable at this time. An edge between a task and a resource or an information set indicates that this task requires either all linked resources or this information set to be completed, and a mission is comprised of all linked tasks.

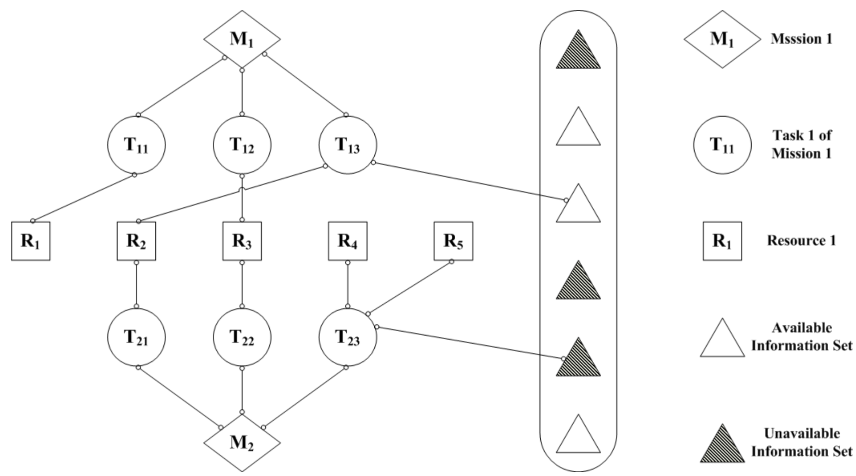

Figure 1. Mission-resource Matching Graph

For more details about the relationship among missions, tasks and resources, we refer to the work in F. Chen *et al.*[1], in which tasks of the same mission have a temporal or causal relationship with each other and can be represented as a task transition graph. The conversational user interfaces (CUIs) used in our CE-SAM component is able to deal with potential task relationships by modelling the high-level user requests into interlinked tasks. In fact, one of the advantages of CUIs is that they allow to reason about hypothetical objects of future events which do not have an immediate graphical representation and therefore cannot be represented in the graphical user interfaces (GUIs).

2.2 System Components and Operations

Compared to previous work in F. Chen's paper[1], our system architecture is improved in the following aspects: 1) reasoning component including CUIs which allows interactive refinements of mission requirements, 2) stochastic resource allocation problem solver that is adaptable to more complicated and realistic conditions, and 3) instructive feedback reports generated from the cooperation of 1) and 2) that provide more flexible and powerful decision-making support.

Figure 2 shows our system architecture and work flows, in which *"CE-SAM"* and the *"Resource Allocation Solver"* are the two most important components. Starting from refining high-level ISR mission requests from commanders, the operations within our system form two separate feedback loops, which are derived from those two key components, respectively.

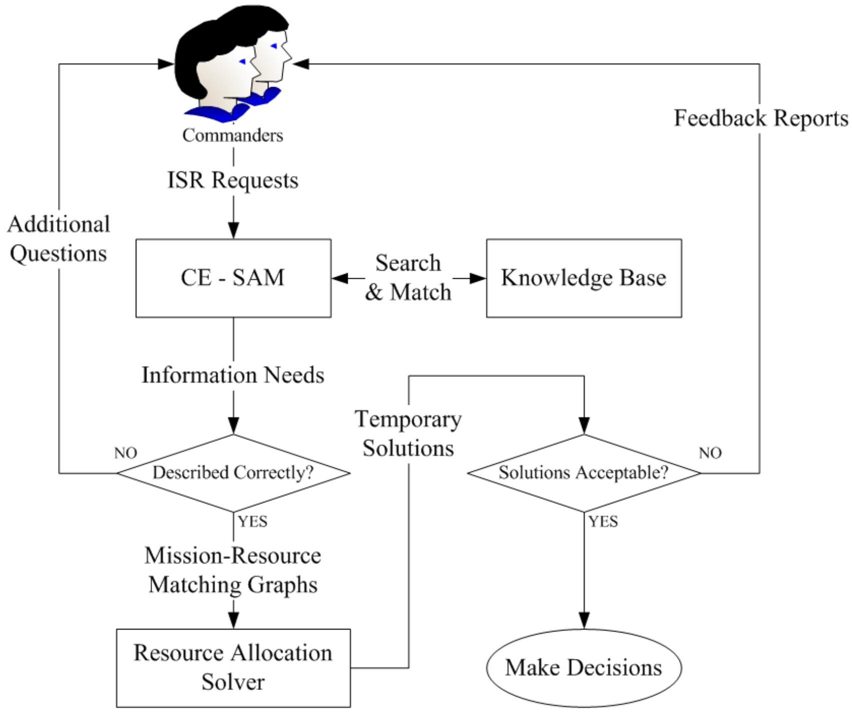

Figure 2. System Architecture and Work Flows

One loop starts when the commanders submit their requests and is repeated until our system finally understands the information needs correctly through raising questions via CE-SAM to guide users to refine the ISR requirements of their missions based on their actual information needs. A precisely described mission can be then decomposed as a set of related tasks, based on which the mission is eventually matched with necessary resources.

The other feedback cycle takes as input the well defined mission-resource matching graphs generated by the first loop, solves the resource allocation problem, and returns temporary solutions along with the feedback reports to the commanders. Acceptable solutions lead the commanders to make their final decisions immediately; otherwise, the feedback reports describe details about current conditions such as success probability and bottleneck resources, and provide some alternative solutions to assist the commanders in making readjustments on their ISR mission requests.

The CE-SAM component provides an interactive interface that, via a mixture of Natural Language parsing and conversational interaction, helps the users refine their information needs and express those in a form which is understandable for the system. The CUIs allow for complex user interactions with our system without requiring extensive training or a professional technical background (e.g., in formal query languages or ontology building). To leverage the advantages of CUIs, CE-SAM guides users through refining and satisfying their information requirements in the context of ISR operations. CE-SAM allows to relate the information needs to pre-existing concepts in the ISR knowledge base, via conversational interactions implemented on a tablet device. The knowledge base is represented using Controlled English (CE) – a form of controlled natural language that is both human-readable and machine processable (i.e., can be used to implement automated reasoning). Users interact with the CE-SAM conversational interface using natural language, which the system converts to CE for feeding-back to the user for confirmation (e.g., to reduce misunderstanding). This process not only allows users to access the ISR assets that can meet their mission needs, but also assists them in extending the CE knowledge base with new concepts.

In Figure 3 we show the prototype of our conversational interface which assists users in the refinement from high level information requirements to low level ISR information needs, represented using the ISR ontology as shown in A. Preece's work[2]. Basically CE-SAM helps to translate the information needs of ISR missions in the form of questions or statements (e.g., "I am looking for intruders") to an equivalent ISR sensing task representation which the system can then use in order to match the requests with existed information sets and ISR assets via our stochastic resource allocation algorithm. We refer to our previous work[2] and latest paper[3] for more details about CE-SAM.

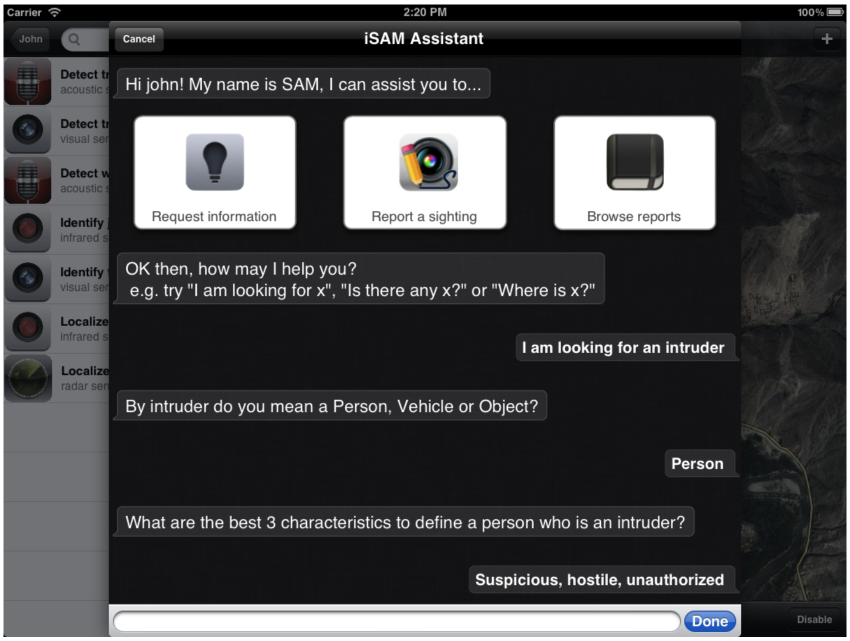

Figure 3. CE-SAM conversational interface: refinement of the information needs

The output of CE-SAM is the mission-resource matching graph in Figure 1, which is utilized by the Resource Allocation Solver component. This component runs our heuristic algorithm to solve the stochastic resource allocation problem even if both the demands for resources and profits of missions are not deterministic. The output would be a subset of submitted missions which is recommended to be admitted for a good payoff while also taking into account the availability of current resources. Besides, our algorithm also generates feedback reports for the commanders to describe rationale on both admitted and non-admitted mission sets, such as probability of a successful execution for the admitted set, and the bottleneck of resources that blocks those non-admitted missions. Both the temporary solutions and the feedback reports are delivered back to the commanders via the CE-SAM conversational interface so that the commanders can either agree with the system-suggested resource allocation plan, or readjust their ISR needs depending on the assistance from the feedback reports. A commander could for example reconsider to set a lower chance of success for admitting more missions, temporarily preempt some amount of bottleneck resources to facilitate current mission set, or partially cut down information more or less to skip off the unavailable resources.

3. RESOURCE ALLOCATION PROBLEM

In this section, we formally define and discuss our stochastic resource allocation problem setting, and present an example case thereafter. Taking at input a set of missions with stochastic demands and profits, our algorithm searches for a subset of missions to be admitted, which can be supported by sufficient resources (i.e., the probability that actual required resources exceed available capacity is low) and is capable of maximizing the total benefit. Since the total profit is no longer constant, we change our objective to search for the optimal profit that can be achieved with high confidence.

3.1 Problem Formulation

Suppose that a set R of resources and a set M of missions are given. Each resource R_i has a capacity c_i, and each mission M_j can produce a profit V_j if each of its demand D_{ij} for R_i is satisfied. Our objective is to find the optimal subset of M to be admitted, which leads to the best total profit without violating any capacity constraint in all dimensions. We use a binary decision variable x_j to indicate whether M_j is admitted or not. That is, $x_j = 1$ means that M_j is authorized to exclusively access resources as requested; otherwise, $x_j = 0$. The problem thus can be formulated as the following linear program:

$$max \quad \sum_j V_j x_j \tag{1}$$

$$s.t. \quad \sum_j D_{ij} x_j \leq c_i \quad \forall i \tag{2}$$

In a general knapsack problem[6] setting, both D_{ij}'s and V_j's are constants. In our problem, however, they are modeled as random variables following some known distributions, which is believed to be more reasonable when simulating most cases in the real world. For example, a mission may last slightly longer or shorter than expected, consuming more or less resources than originally requested; or a detection mission may turn out to be either successful or failed, which is unpredictable until its completion, achieving total different levels of profit. Therefore, we take into account such uncertainty of both demands and profits as well as the capacity constraints on the underlying resources.

Although it is ideal that the chosen subset of M is finally allocated adequate resources as expected, for the demands following unbounded distributions (e.g, Gaussian, Poisson), a low capacity overflow frequency may be tolerated because the capacity constraints always have a chance to be violated. Let p_i denotes the tolerant overflow rate of R_i, then equation (2) is replaced by the following equation:

$$\Pr(\sum_j D_{ij} x_j > c_i) < p_i \quad \forall i \tag{3}$$

The problem now, with stochastic demands and constant profits, is in the form of a chance-constrained program[7], and may be solved by scenario approximation[8,9], sample average approximation[10,11], and heuristic equivalent transformation[4].

However, as mentioned before, according to our setting, the profit variables are also stochastic, which makes the problem more complicated to evaluate the total combined profits of different subset selections. For instance, when facing two candidate subsets of missions, one of which has 80% chance to produce more than 100 units of profit, while the other shows 80% probability to return at least 50 units of profit but 20% confidence to return more than 200 units, it is hard to tell which one is better since for the second subset the probability of achieving a very high level of benefit is unacceptably low. Similarly, a better expectation of the combined profit is not necessarily reliable, because sometimes those great numbers ranging within the low probability zone significantly increase the value of expectation. Therefore, to define our objective more reasonably, another probability, q, is introduced, based on which we can calculate the maximum acceptable level of reward obtained from a given set of admitted missions.

For an admitted set S, if the chance of getting at least some level of profit λ^S is equal to or higher than q, we call this λ^S meaningful, and define λ^S_q as the greatest meaningful λ^S. Then our objective is changed to search for the optimal λ^S_q among all potential admitted subset choices. Take the example in the last paragraph and set q as 80%. Let the two subsets be S_1 and S_2, respectively. For S_1, the probability of achieving any level of profit no more than 100 units should be no less than that of getting 100, which is just 80%. Thus here $\lambda^{S_1}_q$ is 100. Similarly, we can calculate $\lambda^{S_2}_q$ as 50. As a result, we prefer S_1 to S_2 if only one of them can be selected, because in this case $\lambda^{S_1}_q$ is greater than $\lambda^{S_2}_q$.

We can describe the relationship between λ^S, λ^S_q and q for a given S as follows:

$$\Pr(\sum_j V_j x_j \geq \lambda^S) \geq q$$
$$\Pr(\sum_j V_j x_j \geq \lambda^S_q) = q \tag{4}$$

Then combining (3) and (4), finally we are able to formulate our problem setting as follows:

$$\begin{aligned} max \quad & \lambda^S_q \\ s.t. \quad & \Pr(\sum_j D_{ij} x_j > c_i) < p_i \quad \forall i \\ & \Pr(\sum_j V_j x_j \geq \lambda^S_q) = q \\ where \quad & S = \{M_j | x_j = 1\} \end{aligned}$$

3.2 An Example Case

Details about our stochastic resource allocation heuristic are out of the scope of this paper, and can be found in the work of N. Hu *et al.*[12] Here we present an example scenario to show how our problem formulation described above leads to a recommended solution along with a flexible decision-making support feedback report.

Suppose that a commander submits 3 missions (M_1, M_2 and M_3). After the requirement refinement loop, it is found that, to produce profit, every mission needs some amount of resource R, whose available capacity is 50 units at that time. Both demands for R and profits are random variables following known normal distributions, details of which are shown in Table 1.

Random Variables	Demands		Profits	
Parameters	Means	Variances	Means	Variances
M_1	10	100	15	5
M_2	15	50	30	5
M_3	20	25	20	30

Table 1. Distribution Details of Missions

Furthermore, the commander sets p and q as 0.2 and 0.8, respectively, which means the admitted mission set should have no more than 20% chance to require more than 50 units of R, and the expected profit could be met with at least 80% confidence.

According to the theory of statistics, we can calculate the result of every combination of mission which is shown in following Table 2.

Solutions	x_1	x_2	x_3	Necessary R	Overflow Rates	λ^S_q
S_1	1	1	0	35	0.02	42
S_2	1	0	1	39	0.04	30
S_3	0	1	1	42	0.04	45
S_4	1	1	1	56	0.35	60

Table 2. Results of Some Combinations of Missions

As defined in previous section, our objective is to maximize λ^S_q while the capacity overflow rate is lower than 20%. Therefore, the best solution is S_3, which admits M_2 and M_3 and achieves the greatest λ^S_q without violating the capacity overflow constraint. Moreover, based on the information displayed in Table 2, additional suggestions could be reported to the commander as feedback that assists this desicion-making process. Such feedback reports offer some alternative suggestions, which include but are not limited to:

- "Trade profit for lower resource consumption": although $\lambda_q^{S_1}$ is slightly lower than $\lambda_q^{S_3}$, the fact that S_1 requests much less R makes it possibly a better solution leaving more resources for the missions submitted afterwards;
- "Loose overflow constraint" or "Preempt some bottleneck resources": considering S_4's one-third profit improvement over S_3, either taking the risk of S_4's higher failure probability or rescheduling 6 more units of already allocated R to temporarily serve this mission set may pay off.

Via our CE-SAM interface, the commander sees not only the recommended solution $\{M_2, M_3\}$, but also alternatives as described above. If the commander satisfies with the suggested solution, M_2 and M_3 are executed directly; otherwise, he can tailor his mission according to the feedback reports and reconsiders the new solution recommended by our system, until finally a satisfactory result is reached.

4. CONCLUSION

In this paper, a system architecture is designed and proposed for decision-making support on ISR missions. Commanders can submit their ISR missions via an interactive conversational user interface, which helps refine their information needs and exploits resource requirements of each mission. The interface matches each mission with a bundle of corresponding resources after parsing users' requirements, and then delivers such matching as an input to the component in charge of allocating resource among multiple missions. This resource allocating component runs a heuristic algorithm to solve stochastic knapsack problem even if both the demands for resources and profits produced by missions are non-deterministic and may change overtime. Sets of admitted missions are returned as temporary suggested solutions to the commanders along with some additional details about current conditions, which assist the commander in making best decision by providing alternative selections, indicating current resource bottleneck, etc. Such extra information forms a feedback cycle between the commanders and our system, which converges when both the final resource allocation plan and expected profit are satisfied by the mission issuers.

REFERENCES

[1] F. Chen, T. La Porta, D. Pizzocaro, A. Preece, and M. B. Srivastava, "A system architecture for exploiting mission information requirement and resource allocation," *Proceedings of SPIE Defense, Security and Sensing* **8389**, pp. 838908–9, 2012.

[2] A. Preece, D. Pizzocaro, D. Braines, and D. Mott, "Tasking and sharing sensing assets using controlled natural language," *Proceedings of SPIE Defense, Secuirty and Sensing* **8389**, pp. 838905–14, 2012.

[3] D. Pizzocaro, C. Parizas, A. Preece, D. Braines, D. Mott, and J. Bakdash, "Ce-sam: a conversational interface for isr mission support," *Proceedings of SPIE Defense, Secuirty and Sensing* **8758-6**, 2013.

[4] F. Chen, T. La Porta, and M. B. Srivastava, "Resource allocation with stochastic demands," in *Proceedings of the 2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, *DCOSS '12*, pp. 257–264, IEEE Computer Society, (Washington, DC, USA), 2012.

[5] T. İlhan, S. M. R. Iravani, and M. S. Daskin, "The adaptive knapsack problem with stochastic rewards," *Operations Research* **59**, pp. 242–248, Jan. 2011.

[6] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*, Springer, 2004.

[7] A. Charnes, W. Cooper, and G. Symonds, "Cost horizons and certainty equivalents: an approach to s-tochastic programming of heating oil," *Management Science* **4**(3), pp. 235–263, 1958.

[8] G. Calafiore and M. Campi, "The scenario approach to robust control design," *Automatic Control, IEEE Transactions on* **51**(5), pp. 742–753, 2006.

[9] A. Nemirovski and A. Shapiro, "Scenario approximations of chance constraits," in *Probabilistic and randomized methods for design under uncertainty*, pp. 3–47, Springer, 2006.

[10] J. Luedtke and S. Ahmed, "A sample approximation approach for optimization with probabilistic constraints," *SIAM Journal on Optimization* **19**(2), pp. 674–699, 2008.

[11] B. Pagnoncelli, S. Ahmed, and A. Shapiro, "Sample average approximation method for chance constrained programing: theory and applications," *Journal of optimization theory and applications* **142**(2), pp. 399–416, 2009.

[12] N. Hu, D. Pizzocaro, M. P. Johnson, T. La Porta, and A. D. Preece, "Resource allocation with non-deterministic demands and profits," Tech. Rep. NAS-TR-0166-2013, Network and Security Research Center, Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA, Apr. 2013.