Report Number: wucse-2009-3

2009

Submodular utility optimization in sensor networks for capacity constraints

You Xu, Yixin Chen, Chenyang Lu, Sangeeta Bhattacharya, and Abu Saifullah

With the fast development of wireless sensor network (WSN) technologies, WSNs have widely shifted from a specialized platform for a single application to an integrated infrastructure supporting multiple applications. It is hence a critical problem to allocate multiple applications to multiple sensors in order to maximize user utility subject to various resource constraints. The resulting constrained optimization problem is difficult since it is discrete, nonlinear, and not in closed-form. In this report, we develop an efficient optimization algorithm with rigorous approximation bounds for submodular monotonic optimization with multiple knapsack constraints. Based on a variance reduction formulation, we prove several... **Read complete abstract on page 2.**

Recommended Citation

Xu, You; Chen, Yixin; Lu, Chenyang; Bhattacharya, Sangeeta; and Saifullah, Abu, "Submodular utility optimization in sensor networks for capacity constraints" Report Number: wucse-2009-3 (2009). *All Computer Science and Engineering Research.*
https://openscholarship.wustl.edu/cse_research/14

Department of Computer Science & Engineering - Washington University in St. Louis
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160.

Submodular utility optimization in sensor networks for capacity constraints

You Xu, Yixin Chen, Chenyang Lu, Sangeeta Bhattacharya, and Abu Saifullah

Complete Abstract:

With the fast development of wireless sensor network (WSN) technologies, WSNs have widely shifted from a specialized platform for a single application to an integrated infrastructure supporting multiple applications. It is hence a critical problem to allocate multiple applications to multiple sensors in order to maximize user utility subject to various resource constraints. The resulting constrained optimization problem is difficult since it is discrete, nonlinear, and not in closed-form. In this report, we develop an efficient optimization algorithm with rigorous approximation bounds for submodular monotonic optimization with multiple knapsack constraints. Based on a variance reduction formulation, we prove several important theoretical properties, including the monotonicity and submodularity of functions and the multiple knapsack structure of constraints. Then, by exploiting these properties, we propose a local search algorithm with fractional relaxation of constraints and prove the approximation bound that is better than any known results. Experimentally, we verify the theoretical properties on a large dataset from the Intel Berkeley Lab. Comparison against other constrained search algorithms show that our algorithm is superior in both solution time and quality, making it a practical choice for WSN design.

2009-3

Submodular utility optimization in sensor networks for capacity constraints

Authors: You Xu and Yixin Chen and Chenyang Lu and Sangeeta Bhattacharya and Abu Saifullah

Corresponding Author: chen@cse.wustl.edu

Abstract: With the fast development of wireless sensor network (WSN) technologies, WSNs have widely shifted from a specialized platform for a single application to an integrated infrastructure supporting multiple applications. It is hence a critical problem to allocate multiple applications to multiple sensors in order to maximize user utility subject to various resource constraints. The resulting constrained optimization problem is difficult since it is discrete, nonlinear, and not in closed-form.

In this report, we develop an efficient optimization algorithm with rigorous approximation bounds for submodular monotonic optimization with multiple knapsack constraints. Based on a variance reduction formulation, we prove several important theoretical properties, including the monotonicity and submodularity of functions and the multiple knapsack structure of constraints. Then, by exploiting these properties, we propose a local search algorithm with fractional relaxation of constraints and prove the approximation bound that is better than any known results.

Experimentally, we verify the theoretical properties on a large dataset from the Intel Berkeley Lab. Comparison against other constrained search algorithms show that our algorithm is superior in both solution time and quality, making it a practical choice for WSN design.

Type of Report: Other

Optimizing Quality of Sensing in Shared Sensor Networks with Resource Constraints

You Xu, Abusayeed Saifullah, Yixin Chen, Sangeeta Bhattacharya, and Chenyang Lu

Department of Computer Science and Engineering
Washington University in St. Louis

Abstract. With the fast development of wireless sensor network (WSN) technologies, WSNs have widely shifted from a specialized platform for a single application to an integrated infrastructure supporting multiple applications. It is hence a critical problem to allocate multiple applications to multiple sensors in order to maximize user utility subject to various resource constraints. The resulting constrained optimization problem is difficult since it is discrete, nonlinear, and not in closed-form.

In this paper, we develop an efficient optimization algorithm with rigorous approximation bounds for submodular monotonic optimization with multiple knapsack constraints. Based on a variance reduction formulation, we prove several important theoretical properties, including the monotonicity and submodularity of functions and the multiple knapsack structure of constraints. Then, by exploiting these properties, we propose a local search algorithm with fractional relaxation of constraints and prove the approximation bound that is better than any known results.

Experimentally, we verify the theoretical properties on a large dataset from the Intel Berkeley Lab. Comparison against other constrained search algorithms show that our algorithm is superior in both solution time and quality, making it a practical choice for WSN design.

1 Introduction

While wireless sensor networks (WSNs) have traditionally been used as specialized platforms for single applications, recent years have witnessed the emergence of integrated WSNs as shared infrastructure for multiple applications. A key challenge faced by shared WSNs is to deal with severe resource constraints of WSN devices. For example, a representative WSN platform, the TelosB mote [1], only has 10 KB of RAM, a 250 Kbps radio, and a 16-bit CPU running at 8 MHz. These resource constraints necessitate the allocation of nodes to contending applications to maximize the overall Quality of Sensing (QoS) subject to multiple resource constraints. Multi-application allocation in a shared WSN is a challenging optimization problem. The objective functions for this type of problem is typically defined over a finite discrete set representing the assignments of applications to different subsets of nodes. The resulting constrained optimization problem is difficult since it is discrete, nonlinear, and does not have close form objective functions.

To tackle this challenging constrained optimization problem, a key observation is that the QoS of many networked sensing applications has *submodular* properties with regard to node allocation due to the nature of many physical phenomena. Intuitively, a function f that maps a subset of a set S to a real value is submodular if it has a *diminishing return* property, which means that adding an element to a smaller subset of S makes a bigger difference than adding it to a larger subset of S.

Formally, we can define the function with this attribute as a *submodular function*:

Definition 1 *Given a finite set S of n elements and a function $f : 2^S \mapsto R$, f is submodular if and only if $f(A \cap B) + f(A \cup B) \leq f(A) + f(B)$ for any $A, B \subseteq S$. f is monotonic if and only if $f(A) \leq f(B)$ for any $A \subseteq B \subseteq S$.*

The submodularity of node allocation in WSNs is due to the inherent property that sensor readings from different nodes are often correlated. For instance, the temperature readings from different nodes in a same room are often correlated to each other. Allocating a new node to a temperature monitoring application therefore results in diminishing improvement to the QoS as the set of nodes allocated to the application grows. Submodularity of sensor allocation for monitoring temperature [3] and water quality [9] has been observed in previous studies of real-world data sets.

While optimization of submodular functions has been extensively studied, the classical submodular optimization theory [8] has several fundamental limitations that severely restrict its applicability. In addition to requiring submodularity of functions, the theory also requires that the constraints yield a matroid. In QoS optimization for shared WSNs, the classical assumptions are typically violated. In fact, the optimization problem we study in this paper is more difficult than the classical submodular optimization problem since its constraint is not a uniform matroid constraint. Instead, its constraints are not even multiple matroid constraints, but rather multiple knapsack constraints, which have been rarely studied until some recent work is proposed [10].

In this paper, we first give a constrained optimization formulation of the multi-application allocation problem in shared WSNs. We show that the QoS optimization problem in a shared WSN can be formulated based on a *variance reduction* function that maps a set of nodes to a QoS value in presence of inter-node QoS correlation. Then we provide several major theoretical results. 1) We show that the problem is a submodular optimization problem with multiple knapsack constraints. 2) We prove that our variance reduction formulation is a monotonic function. 3) We propose a fractional relaxation based greedy local search algorithm for solving the constrained optimization problem and prove that our algorithm can achieve a 1/3-approximation bound, a bound better than any other existing results. Our algorithm and results are not limited to sensor applications, but are general for submodular monotonic optimization with multiple knapsack constraints.

Empirically, we verify the submodularity and monotonicity of our variance reduction using Intel Berkeley Lab's real-world data on humidity and temperature monitoring [2] . We also present the experimental results and analyze the scalability and execution time of our algorithm. We compare our algorithm against two standard allocation algorithms, randomized allocation and simulated annealing. The results demonstrate that our algorithm achieves a much higher system QoS and is much more efficient than other algorithms. As a result, the algorithm is an attractive and practical solution for application allocation in shared sensor networks.

2 Related Work

We overview the existing theory for submodular optimization. Problems with submodular objective functions have been extensively studied. However, most existing work require special matroid structure of the constraints.

Optimization with multiple applications under resource constraints is difficult since the problem often violates the classical assumptions of monotonicity, submodularity, and uniform-matroid constraints. We discuss classical and recent results below.

Definition 2 *Given a finite set S of n elements, a submodular optimization problem (SOP) is*

$$\textit{maximize}\ _{V\subseteq S} f(V), \quad \textit{subject to:}\ \ h_i(V)=0, \quad i=1,\cdots,M \qquad (1)$$

where $f: 2^S \mapsto R$, f is submodular function on S and $h_i : 2^S \mapsto \{0,1\}, i = 1,\cdots,M$ are constraint functions.

Definition 3 *Given a finite set S, an independent system I on S is a set of subsets of S such that: 1) the empty set $\emptyset$ is in I, and 2) for every set $K \subseteq S$, if $K \in I$, then every subset of K is in I.*

Definition 4 ***(Matroid)*** *Given a finite set S, an independent system M on S is a matroid if it satisfies: for any $A \subseteq M$ and $B \subseteq M$ where A has more elements than B, there always exists an element $a \in A$ which is not in B such that $B \cup \{a\} \in M$.*

Definition 5 ***(Knapsack Constraint)*** *Given a finite set S, a constraint h in a SOP on S is a knapsack constraint if it satisfies that, $h(V) = 0$ if and only if $\sum_{s_i \in S} w_i * x_i \leq R$, where x_i is a binary variable representing if s_i is in V ($x_i = 1$) or not ($x_i = 0$), w_i is a positive weight assigned to each s_i, and R is a positive constant.*

Definition 6 ***(Multiple Knapsack Constraints)*** *For a SOP, a set of its constraints is called multiple knapsack constraints if each and every constraint in the set is a knapsack constraint. Here the weight vector w might be different for knapsack constraints across the set.*

We summarize key existing results for submodular optimization below. The results of Nemhauser *et. al.* showed that the maximization of a monotonic submodular function under the constraints of a uniform matroid is NP-complete [8]. Therefore, the optimization of a submodular function generally does not pursue the optimal solution. Instead, an approximated solution is acceptable. Fisher, Nemhauser and Wolsey's seminal work [8] showed that a greedy algorithm can achieve a $1 - 1/e$ approximation rate in maximizing a non-decreasing submodular function on a uniform matroid. Feige et al. [6] further proved that there is no $(1 - 1/e + \epsilon)$-approximation polynomial algorithm for any constant $\epsilon > 0$, unless $P = NP$. Thus, $(1 - 1/e)$ is considered to be the best approximate rate we can get for polynomial algorithms.

The above results are for submodular optimization with uniform matroid constraints. For other SOPs, we can classify them into those with monotonic or non-monotonic objectives. For SOPs with monotonic submodular functions, Fisher et. al. [7] proposed a $1/2$ approximation algorithm for SOPs with single matroid constraints. Some recent work improved this bound to $(1 - 1/e)$ assuming that f is the sum of some matroid ranks [10]. For SOPs with one knapsack constraint, Sviridenko [12] proposed a $(1 - 1/e)$-approximation algorithm. For SOPs with multiple matroid constraints, we can achieve the bound of $1/(k + 1)$ where k is the number of matroid constraints [10]. There is no result specifically for SOPs with multiple knapsack constraints. A related work by Conforti [4] shows that if all the constraints are in an independent system, which is a general case for knapsack and matroid constraints, we can get a $\frac{1}{1+k+1/k}$-approximation rate.

For SOPs with nonmonotonic objective functions, the only known existing work is a very recent $(\frac{1}{k+2+1/k+\epsilon})$-approximation algorithm [10] for k-matroid constraints based on a local search procedure. The algorithm also provides a $(\frac{1}{5} - \epsilon)$ approximation rate for SOPs with multiple knapsack constraints.

3 Problem Formulation

In this paper, we formulate our objective function by a variance reduction function.

Suppose V is the set of sensor nodes we will deploy application t on. For any two sensor nodes i and $j \in V$, $i \neq j$, the covariance σ_{ij} can be calculated based on measured data. We define a **kernel matrix** $\mathcal{K}$ where the elements in the ith row and jth column is σ_{ij}. $\mathcal{K}$ is essentially the covariance matrix for sensor nodes. For two subsets of sensor nodes A and $B \subseteq V$, we denote the covariance matrix of them by $\mathcal{K}_{AB}$, with rows corresponding to A and columns corresponding to B extracted from $\mathcal{K}$.

Given a finite subset $A \subset V$ with applications allocated, if the system is Gaussian, for any sensor node $y \in V$, we can predict its reading using maximum likelihood estimation as $\arg\max_{X_y} P(X_y|X_A)$. Note that the distribution of X_y has variance as

$$\sigma^2_{y|A} = \mathcal{K}_{yy} - \mathcal{K}_{yA}\mathcal{K}^{-1}_{AA}\mathcal{K}_{Ay}$$

Thus, for a given allocation A, the variance of the unknown region $\bar{A} = V \backslash A$ is

$$\sigma^2_{\bar{A}|A} = tr(\mathcal{K}_{\overline{A}\bar{A}}) - tr(\mathcal{K}_{\bar{A}A}\mathcal{K}^{-1}_{AA}\mathcal{K}_{A\bar{A}}),$$

where $tr()$ is the trace function of a matrix.

If we consider the QoS as the confidence of the measurement, natually, we want to minimize the variance of $\bar{A}$ given A such that the quality of sensing is maximized. Namely, we want to maximize the negation of the variance. Since we have $tr(\mathcal{K}) = tr(\mathcal{K}_{AA}) + tr(\mathcal{K}_{\bar{A}\bar{A}})$, the objective function to one application is:

$$\text{maximize} \quad Q = tr(\mathcal{K}_{AA}) + tr(\mathcal{K}_{\bar{A}A}\mathcal{K}^{-1}_{AA}\mathcal{K}_{A\bar{A}}) \tag{2}$$

Now we turn to multiple applications. When there are multiple applications to allocate onto sensor nodes, our objective function is a sum of variance reductions for each application. Suppose we have p applications to allocate, the objective function is to maximize $\sum_{t=1}^{p} Q_t$, where each Q_t is the Q function for application t.

Since sensors usually have limited amount of resources, we have resource constraints in our formulation. For example, two critical resources for applications to share are CPU usage and memory consumption. For each sensor, the total CPU and memory consumed by all the applications assigned to the sensor cannot exceed its limits. In general, suppose there are r resources each having a maximum usage of $U_i, 1 \leq i \leq r$, the constrained optimization formulation is

$$\begin{aligned} (P_1): \qquad & \text{maximize } \sum_{t=1}^{p} Q_t \qquad\qquad (3) \\ & \text{subject to } \sum_{t=1}^{p} a_{t,j} R_{i,t,j} \leq U_i, \ \forall i = 1, \cdots, r, \ \forall j = 1, \cdots, n \end{aligned}$$

where $a_{t,j}$ is a binary variable denoting if application t is assigned to node j, $R_{i,t,j}$ is the usage of resource i when application t is assigned to a sensor node j, and n is the number of sensors.

4 Theoretical Properties

Now we study the characteristics of P_1. These characteristics facilitate the design and analysis of our search algorithm.

4.1 Submodular objective and knapsack constraints

Our variance reduction formulation is generaly not submodular for an arbitary kernal matrix $\mathcal{K}$. However, it can be shown that in the absence of conditional suppressor variables, the variance reduction formulation of Q is indeed submodular [5]. Since the conditional supressor is a statistical attribute, the verification of such attribute is based on given data sets. We note that a statistical verification for the conditional supressor attribute is inherently the submodularity verification for two arbitary subsets. Therefore, we will directly verify the submodularity of the variance reduction function empirically based on some real world data.

When the variance reduction function Q_t for each application t is submodular, since the summation of submodular functions is still a submodular function, the objective function of P_1 $\sum_{t=1}^{p} Q_t$ is submodular. Further, we see that constraints in P_1 are multiple knapsack constraints since they satisfy Definition 6.

In short, our QoS based application allocation problem is a submodular optimization problem (SOP) with multiple knapsack constraints.

4.2 Monotonicity

In addition to the submodularity of the objective function of P_1, we find that it is also monotonic. The monotonicity is a strong property that enables us to improve the approximationbound.

Assuming that the application set P contains p applications and the node set V contains n nodes. Any application-to-node assignment (or assignment for short) is a subset of $\mathcal{M} = P \times V$. We have the following theorem:

Theorem 1 *For p applications and n sensor nodes, if A and B are two application-to-node assignments where $A \subseteq B \subseteq \mathcal{M}$, we have $Q(A) \leq Q(B)$ if all kernel matrices for all p applications are semi-positive definite (s.p.d.).*

Since Q is a sum of Q_t for $t = 1, \cdots p$, we only need to prove that Q_t is monotonic if $\mathcal{K}_t$ is s.p.d. We prove this result by proving the following lemma.

Lemma 1. *Given a s.p.d. matrix $\mathcal{K}$ in the block separated form as*

$$\mathcal{K} = \begin{pmatrix} A & D & E \\ D^T & B & F \\ E^T & F^T & C \end{pmatrix},$$

$$tr(A) + tr((DE)(DE)^T A^{-1}) \leq tr((A+B) + \begin{pmatrix} E \\ F \end{pmatrix} \begin{pmatrix} E \\ F \end{pmatrix}^T \begin{pmatrix} P & Q \\ Q^T & R \end{pmatrix}) \tag{4}$$

where

$$\begin{pmatrix} P & Q \\ Q^T & R \end{pmatrix} = \begin{pmatrix} A & D \\ D^T & B \end{pmatrix}^{-1} \tag{5}$$

Proof: We first expand both sides of (4) to $tr(A) + tr(D^T A^{-1} D + E^T A^{-1} E) \leq tr(A + B) + tr(E^T P E + F^T Q^T E + E^T Q F + F^T R F)$

The proof consists of two parts.

Part 1: First we prove that $tr(A + D^T A^{-1} D) \leq tr(A + B)$.

Since $\mathcal{K}$ is an *s.p.d.* matrix, for any real vector x, y , we have

$$(x, y)^T \begin{pmatrix} A & D \\ D^T & B \end{pmatrix} (x, y) > 0. \tag{6}$$

Since an *s.p.d.* matrix always has a Cholesky factorization, we rewrite A as U^TU and plug in to the inequality above. Note that U is invertible and $U^{-T} = (U^T)^{-1}$; the left side becomes $x^TU^TUx + x^TU^TU^{-T}Dy + y^TD^TU^{-1}Ux + y^TBy + y^TD^TU^{-1}U^{-T}Dy - y^TD^TA^{-1}Dy$. Thus, (6) is actually $(Ux + U^TDy)^T(Ux + U^{-T}Dy) + y^T(-D^TA^{-1}D + B)y > 0$.

Note that since the linear system $Ax + Dy = 0$ w.r.t x always has a solution for any y since A is *s.p.d.*, the vector $(Ux+U^TDy)$ can always be 0 for any given y. This implies that $y^T(-D^TA^{-1}D + B)y > 0$ for any y. Thus, we know that $(-D^TA^{-1}D+B)$ is *s.p.d*,, i.e. $tr(-D^TA^{-1}D+B) > 0$ and $tr(D^TA^{-1}D) < tr(B)$.

Part 2: Now we prove $tr(E^TA^{-1}E) \leq tr(E^TPE+F^TQ^TE+E^TQF+F^TRF)$, which can be written as

$$tr(E^T(P - A^{-1})E + F^TQ^TE + E^TQF + F^TRF) \geq 0 \tag{7}$$

Since we have $AP+DQ^T = I$ and $AQ+DR = 0$, namely, $P+A^{-1}DQ^T = A^{-1}$ and $QR^{-1}Q^T+A^{-1}DQ^T = 0$, we get $E^T(P-A^{-1})E = E^T(-A^{-1}DQ)E = E^T(QR^{-1}Q^T)E$

Plugging this to (7), we get $tr(E^T(QR^{-1}Q^T)E + F^TQ^TE + E^TQF + F^TRF) \geq 0$, which is essentially

$$\begin{pmatrix} E^T & F^T \end{pmatrix} \begin{pmatrix} Q & 0 \\ R & I \end{pmatrix} \begin{pmatrix} R^{-1} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} Q & R^T \\ 0 & I \end{pmatrix} \begin{pmatrix} E & F \end{pmatrix}^T$$

Since R^{-1} is *s.p.d.*, we see the trace of the above is non-negative. □

4.3 Experimental verification of properties

We have also verified the monotonicity and submodularity of the variance reduction function on Intel Berkeley Lab's data set [2]. This data set contains temperature, humidity, light, and voltage readings collected from 54 sensor nodes deployed in Intel Berkeley Research Lab between February 28th and April 5th, 2004. In particular, we have separated temperature and humidity data of one day recorded by 20 sensor nodes. Using this trimmed data set, we have computed temperature and humidity covariance matrices. Using the temperature covariance matrix, the monotonicity of the variance reduction function has been tested on 30843346 pairs of subsets of the set of 20 nodes. Among these tested pairs, a total of 30405760 pairs (98.6%) have satisfied the monotonicity. We have tested the submodularity by taking 30000000 random pairs of subsets of nodes. A total of 29931514 pairs (99.8%) have satisfied the submodularity. We

Phenomenon	Monotonicity			Submodularity		
	Total	Satisfied	% Satisfied	Total	Satisfied	% Satisfied
Temperature	30843346	30405760	98.6	30000000	29931514	99.8
Humidity	35770000	35294259	98.7	30000000	29924700	99.7

Table 1. Submodularity and monotonicity tests on temperature and humidity data

have performed the similar tests on humidity readings as well. The monotonicity and submodularity have been tested on 35770000 and 30000000 pairs of subsets, respectively, and have been satisfied by 35294259 pairs (98.7%) and 29924700 pairs (99.7%), respectively. The test results are summarized in Table 1.

We have observed that only a small fraction of data cannot satisfy the monotonicity or submodularity of the variance reduction. In fact, the data set is noisy and sensor readings may not be always accurate. These results confirm our theoretical results that the variance reduction function is submodular and monotonic.

5 Constrained Optimization Algorithm

In this section, we present our constrained optimization algorithm and prove its approximation bound. We adopt the fractional relaxation framework proposed in [10]. In addition, we exploit the monotonicity of the objective function to obtain a tighter approximation bound.

5.1 Basic definitions

The objective function that we are going to optimize is $f = Q$. All application to node assignments are subsets of the set $V = \mathcal{M} = \mathcal{P} \times \mathcal{V}$, where $\mathcal{P}$ and $\mathcal{V}$ are the sets of applications and sensors, respectively. Since f maps a subset of V to a value, we name V as the **ground set** of f and $|V| = n$. We also use notation $[n]$ to denote the set $\{1, \cdots, n\}$. We use w_r to denote r normalized weight vectors corresponding to r resource constraints, all having capacities normalized to 1. We define $f^* = max\{f(v) : v \in V\}$ and assume each singleton set $\{v\} \in V$ is feasible for the constraints (otherwise, those violating elements can be removed from the problem).

We assume that the global optimal value for f subject to resource constraints is **Opt**. We further assume that any feasible assignment, which is a subset of V, can be partitioned into two sets : a heavy set and a light set, where elements in the heavy set has a weight factor in the knapsack constraints larger than a given threshold. The heavy set and the light set of the optimal solution V^* are denoted by H and L, respectively.

Our **fractional relaxation greedy (FRG)** algorithm consists of the following steps.

Step 1 Identify heavy elements and light elements based on a given threshold. Enumerate all possible assignments of heavy elements and find the optimal subset of heavy elements.

Step 2 Solve a relaxed problem on light elements in which the function f is extended to a continuous function F.

Step 3 Round back the solution to the relaxed problem (from Step 2) to a solution to the original problem. Combine the heavy elements and light elements to get the final solution.

In Step 1, to decide heavy and light elements, we choose a fixed constant $\eta > 0$. Let $c = \frac{16}{\eta}$, we define $\delta = \frac{1}{4c^3k^4}$. An element $e \in V$ is heavy if $w_i(e) \geq \delta$ for some knapsack constraint $i \in [r]$. The rest of them are called light elements. We use H and L to denote the heavy and light element sets in an optimal integral solution, respectively. In this case, H_i has at most $1/\delta$ elements for each knapsack constraint since $\delta \cdot |H_i| \leq \sum_{e \in H} w_i(e) \leq 1$. Therefore, $|H| \leq r/\delta$, which is a constant. By enumeration, we can get $f(H)$ in a time complexity of $n^{O(r/\delta)}$. So, we only focus on the light elements in Steps 2 and 3.

5.2 Step 2: Solving a relaxed problem

In this step, we extend f defined on the subsets of V to F which is defined on fractional values.

Let F be a function that maps from $[0, 1]^n \to \mathbb{R}^+$. For $y \in [0, 1]^n$, let $\hat{y}$ denote a random $0 - 1$ vector where each element is independently rounded to 1 with probability y_j or 0 otherwise. Then, F is defined as:

$$F(y) = E[f(\hat{y})] = \sum_{S \subseteq V} f(S) \cdot \prod_{i \in S} y_i \cdot \prod_{j \notin S} (1 - y_j)$$

F is a monotonic and concave function which extends f to the continuous domain $[0, 1]^n$. However, extending f to F changes the problem domain from a finite set to a continuous domain, which is not suitable for local search. Hence, we define a new set $\mathcal{G} \in [0 - 1]^n$ called the discretized set of F where for each dimension i, G_i can take values in the form of k/s_i where s_i is a fixed integer and $k \in [s_i]$. We also denote the continuous domain of F that satisfies the resource constraints as $\mathcal{U}$.

```
Result: A local optimum $y$ of the relaxed optimization function $F$
$y = \arg\max F(\{x\})$, $x$ is a singleton set;
while True do
    $\mathcal{N}$ = neighborhood of $y$;
    for $y'$ in $\mathcal{N}$ do
        if $y' \in \mathcal{U} \cap \mathcal{G}$ and $F(y') \geq (1+\epsilon)F(y)$ then $y \leftarrow y'$ ;
    end
end
```

Algorithm 1: The local search algorithm.

Since F is monotonic and concave, the discretization F is still a monotonic function. Previous work [10] also proved that the discretized version of F is still submodular.

We solve the following relaxed problem:

$$max\{F(y) \ : \ \ w_i y \leq 1, \ \ y \in \mathcal{G} \cap \mathcal{U}, \ \ \forall i \in [r], 0 \leq y_i \leq 1\} \qquad (8)$$

We use a local search method to solve (8). For any $y \in \mathcal{G} \cap \mathcal{U}$, y' is in the neighborhood of y if y' differs from y by at most $2r$ elements, among which at most r elements get increased and at most r elements get decreased. Based on the neighborhood, the local search procedure is shown in Algorithm 1.

For $x, y \in \mathbb{R}^n$, we define $(x \vee y)_j := max(x_j, y_j)$ and $(x \wedge y)_j := min(x_j, y_j)$ for $j \in [n]$. Lemma 3.5 in [10] gives the following results:

Lemma 2. *For a local optimum $y \in \mathcal{U} \cap \mathcal{G}$ and any $x \in \mathcal{U}$, we have $(2 + 2n\epsilon) \cdot F(y) \geq F(y \wedge x) + F(y \vee x) - \frac{1}{2n} f^*$ for any $\epsilon > 0$.*

Based on monotonicity, we give the following, stronger result.

Lemma 3. *For a local optimum $y \in \mathcal{U} \cap \mathcal{G}$ and any $x \in \mathcal{U}$, we have $(2 + 2n\epsilon) \cdot F(y) \geq F(x) - \frac{1}{2n} f^*$, or $F(y) \geq (\frac{1}{2} - \epsilon)$***Opt** *for any $\epsilon > 0$.*

Proof. Based on the monotonicity result, it is easy to see that for any x satisfying the knapsack constraints, $(2+2n\epsilon) \cdot F(y) \geq F(y \wedge x) + F(y \vee x) - \frac{1}{2n} f^* \geq F(x) + \frac{1}{2n} f^*$. This result shows that the local search algorithm provides an 1/2-approximate solution for the relaxed problem on F. □

5.3 Step 3: Rounding back to a solution

Till now, we have established an 1/2-approximation algorithm for the relaxed problem in (8). In Step 3, we round the relaxed solution back to an integer solution for the original problem P_1 using algorithm 2. In [10], Lee proved the following two lemmas which we will use directly.

```
Data: A fractional solution $y$
Result: A feasible subset $S$ of $V$
$S \leftarrow \emptyset$ ;
foreach element $e$ in $V$ do
    add $e$ to $S$ with probability $(1-\epsilon)y_e$ ;
    if $S$ violates constraints then return $\emptyset$ ;
end
return $S$ ;
```

Algorithm 2: The simple rounding algorithm.

Lemma 4. *Let* $\alpha(S) = max\{w_i(S) : i \in [r]\}$*, for any* $a \geq 1$*,* $Pr[\alpha(S) \geq a] \leq r \cdot e^{-car^2}$*.*

Lemma 5. *For any* $a \geq 0$*,* $max\{f(S) : \alpha(S) \leq a+1\} \leq 2(1+\delta)r(a+1)\mathbf{Opt}$*.*

Now we prove the following theorem. Suppose we repeat Algorithm 2 many times, each time yielding a S set, we calculate the expected value of $f(S)$, assuming $f(\emptyset) = 0$.

Theorem 2 *The simple rounding algorithm for light elements obtains an expected value* $E[f(S)] \geq (1/2 - \eta)$ *Opt.*

Proof. Consider disjoint events $A_0 = \{\alpha(S) \leq 1\}$ and $A_l = \{\alpha(S) \in (l, 1+l]\}$ for $l \in \mathbb{N}$. We use **ALG** to denote $E[f|A_0]Pr[A_0]$. Obviously, ALG is the expected objective function value for this algorithm's output, because we need $\alpha(S) \leq 1$ in order to to satisfy knapsack constraints.

According to Lemma 2 and 3, we get:

$$\begin{aligned} F(x) = E[f] &= E[f|A_0]Pr[A_0] + \sum_{l \geq 1} E[f|A_l]Pr[A_l] \\ &\leq ALG + \sum_{l \geq 1} re^{-clr^2} 2(1+\delta)r(l+1)\mathbf{Opt} \\ &\leq 8\mathbf{Opt} \cdot l \cdot r^2 \cdot e^{-clr^2} \end{aligned}$$

Thus, we have

$$\mathbf{ALG} = F(x) - \sum_{l \geq 1} E[f|A_l]Pr[A_l] \geq F(x) - \frac{8}{c}\mathbf{Opt}$$

From Lemma 5 we know that $F(x) \geq (1/2 - \epsilon)\mathbf{Opt}$. Let $\eta/2 = \epsilon$, we get that $\mathbf{ALG} \geq 1/2\mathbf{Opt} - (\eta/2 + 8/c)\mathbf{Opt} = (1/2 - \eta)\mathbf{Opt}$.

Now we give an approximation bound for the overall solution after combining heavy and light elements.

Theorem 3 *Our fractional relaxation greedy (FRG) algorithm is a* $(\frac{1}{3} - \eta)$*-approximation algorithm for solving* $P1$*.*

Proof. Recall that we use H and L to denote the heavy and light elements in an optimal integer solution, respectively. The enumeration of heavy elements in Step 1 of FRG will give an objective function value at least $f(H)$. Theorem 2 shows that the simple rounding algorithm for light elements produces a solution of expected value at least $(1/2 - \eta) * f(L)$. Thus, using the convexity of the max function, we have

$$\max\{f(H), (\frac{1}{2} - \eta)f(L)\} \geq \frac{1}{3}f(H) + \frac{2}{3}(1/2 - \eta)f(L) \geq (\frac{1}{3} - \eta)f(H \cup L).$$

The above inequality shows the approximation bound. □

Note that our (1/3-η) bound is better than the (1/5-ϵ) bound in [10], the only known result for submodular problems with multiple knapsack constraints.

6 Empirical Evaluation

In this section, we present our experimental results. We compare our algorithm with a randomized algorithm and a simulated annealing algorithm.

We have run those three algorithms using Intel Berkeley Lab's temperature and humidity readings of 20 sensor nodes[2]. We have compared the performance of these algorithms in allocating two applications (temperature sensing and humidity sensing) to these 20 nodes. The QoS values of the allocated node sets were computed using the Q function. The memory capacities of the 20 nodes were assigned randomly in the range from 1024 bytes to 2824 bytes and the memory requirements of the applications were considered in the range from 900 bytes to 1000 bytes. The CPU requirements of these applications on 20 nodes were considered randomly between 35% to 80%. One application was given weight 2 while the other was given 1.

The randomized algorithm will randomly assign applications to nodes, as long as the knapsack resource constraints are satisfied.

Simulated annealing (SA) is a probabilistic algorithm for global optimization. We used the SA package in the AIMA library [11] with a user-defined neighborhood function such that for any application assignment y, all its neighbors in $\mathcal{N}(y)$ contain only feasible assignments. To avoid a large neighbor space, we also restrict that $y' \in \mathcal{N}(y)$ differs from y for at most two sensor nodes. In the SA algorithm, for an assignment y and its neighbor y', we set the acceptance probability function $P(y, y')$

Run	Randomized Algorithm		Simulated Annealing		Our FRG Algorithm	
	Objective	Time	Objective	Time	Objective	Time
1	926.70	0.39	1077.81	120	1179.46	7.99
2	918.90	0.38	981.32	120	1221.82	5.68
3	935.51	0.32	1198.49	120	1980.60	6.24
4	932.88	0.33	1456.37	120	1558.16	2.95
5	923.67	0.35	984.49	120	983.12	1.09

Table 2. Solution quality and time (in seconds) of three algorithms.

as $\exp(-\frac{Q(y)-Q(y')}{T})$, where T is a parameter called temperature. We set the initial temperature to 200 and set the cooling factor to be 0.75.

We test 5 different randomly selected subsets of data, each tested with 20 random runs. We report the average quality and time in Table 2. From Table 2, we see that our greedy algorithm is slower than randomized algorithm but achieves much higher (better) utility. Comparing to SA, our algorithm is better in terms of both time and quality. SA is too slow to be practical. Our other tests show that SA needs to spend more than an hour to achieve comparable utility value as our algorithm can achieve in just a few seconds.

7 Conclusion

Optimization of the applications QoS is a critical issue for multiple application deployment in resource constrained shared sensor networks. In this paper, we have formulated this optimization problem using a variance reduction function that maps a set of nodes to a QoS value in presence of inter-node QoS dependency. Despite the computational challenges underlying this problem, the optimization can be achieved by observing the special structures of the problem. We have proven the monotonicity and the submodularity of the variance reduction both theoretically and experimentally using the real-world dataset of the Intel Berkeley Lab [2]. We have proposed an efficient approximation algorithm that can achieve a $\frac{1}{3}$ approximation bound. Our algorithm and results are not limited to sensor applications, but are general for submodular monotonic optimization with multiple knapsack constraints.

The effectiveness and the scalability of our algorithm have been evaluated by experiments with varying number of nodes and applications. We have also compared its performance against two standard algorithms and demonstrated that it outperforms those algorithms in achieving the overall quality of sensing.

References

1. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB _Datasheet.pdf.
2. Intel lab data. *http://db.csail.mit.edu/labdata/labdata.html.*
3. Fang Bian, David Kempe, and Ramesh Govindan. Utility-based sensor selection. In *IPSN'06*.
4. Michele Conforti and Grard Cornujols. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete Applied Mathematics*, 7(3):251 – 274, 1984.
5. Abhimanyu Das and David Kempe. Algorithms for subset selection in linear regression. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 45–54, New York, NY, USA, 2008. ACM.
6. Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998.
7. M. Fisher, G. Nemhauser, and L. Wolsey. An analysis of approximations for maximizing submodular set functions-ii. *Mathematical Programming Study*, 8:73–87, 1978.
8. L. A. Wolsey G. L. Nemhauser and M. L. Fisher. An analysis of approximations for maximizing submodular set functions-i. *Mathematical Programming*, 14(1):265–294, 1978.
9. A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory efficient algorithms and empirical studies. *JMLR*, 1, 2008.
10. Jon Lee, Vahab Mirrokni, Viswanath Nagarjan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints, 2009.
11. S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approach, 2nd Edition.* Prentice-Hall, 2003.
12. Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41 – 43, 2004.