Mixed Robust/Average Submodular Partitioning

Kai Wei[1] **Rishabh Iyer**[1] **Shengjie Wang**[2] **Wenruo Bai**[1] **Jeff Bilmes**[1]

[1] Department of Electrical Engineering, University of Washington
[2] Department of Computer Science, University of Washington
{kaiwei, rkiyer, wangsj, wrbai, bilmes}@u.washington.edu

Abstract

We investigate two novel mixed robust/average-case submodular data partitioning problems that we collectively call *Submodular Partitioning*. These problems generalize purely robust instances of the problem, namely *max-min submodular fair allocation* (SFA) [8] and *min-max submodular load balancing* (SLB) [15], and also average-case instances, that is the *submodular welfare problem* (SWP) [16] and *submodular multiway partition* (SMP) [2]. While the robust versions have been studied in the theory community [7, 8, 11, 15, 16], existing work has focused on tight approximation guarantees, and the resultant algorithms are not generally scalable to large real-world applications. This is in contrast to the average case, where most of the algorithms are scalable. In the present paper, we bridge this gap, by proposing several new algorithms (including greedy and relaxation algorithms) that not only scale to large datasets but that also achieve theoretical approximation guarantees comparable to the state-of-the-art. We moreover provide new scalable algorithms that apply to additive combinations of the robust and average-case objectives.

1 Introduction

This paper studies two new mixed robust/average-case partitioning problems of the following form:

Problem 1: $\max_{\pi\in\Pi}\Big[\bar{\lambda}\min_i f_i(A_i^\pi) + \frac{\lambda}{m}\sum_{j=1}^m f_j(A_j^\pi)\Big]$, Problem 2: $\min_{\pi\in\Pi}\Big[\bar{\lambda}\max_i f_i(A_i^\pi) + \frac{\lambda}{m}\sum_{j=1}^m f_j(A_j^\pi)\Big]$,

where $0 \leq \lambda \leq 1$, $\bar{\lambda} \triangleq 1-\lambda$, the set of sets $\pi = (A_1^\pi, A_2^\pi, \cdots, A_m^\pi)$ is a partition of a finite set V (i.e, $\cup_i A_i^\pi = V$ and $\forall i \neq j, A_i^\pi \cap A_j^\pi = \emptyset$), and Π refers to the set of all partitions of V into m blocks. The parameter λ controls the objective: $\lambda = 1$ is the average case, $\lambda = 0$ is the robust case, and $0 < \lambda < 1$ is a mixed case. In general, Problems 1 and 2 are hopelessly intractable, even to approximate, but we assume that the $f_1, f_2, \cdots, f_m$ are all monotone non-decreasing (i.e., $f_i(S) \leq f_i(T)$ whenever $S \subseteq T$), normalized ($f_i(\emptyset) = 0$), and submodular [6] (i.e., $\forall S, T \subseteq V$, $f_i(S) + f_i(T) \geq f_i(S\cup T) + f_i(S\cap T)$). These assumptions allow us to develop fast, simple, and scalable algorithms that have approximation guarantees, as is done in this paper. These assumptions, moreover, allow us to retain the naturalness and applicability of Problems 1 and 2 to a wide variety of practical problems. Submodularity is a natural property in many real-world ML applications [13, 10, 12, 17]. When minimizing, submodularity naturally model notions of interacting costs and complexity, while when maximizing it readily models notions of diversity, summarization quality, and information. Hence, Problem 1 asks for a partition whose blocks each (and that collectively) are a good, say, summary of the whole. Problem 2 on the other hand, asks for a partition whose blocks each (and that collectively) are internally homogeneous (as is typical in clustering). Taken together, we call Problems 1 and 2 *Submodular Partitioning*. We further categorize these problems depending on if the f_i's are identical to each other (*homogeneous*) or not (*heterogeneous*).[1]

[1]Similar sub-categorizations have been called the "uniform" vs. the "non-uniform" case in the past [15, 7].

Problem 1 (Max-(Min+Avg))		Problem 2 (Min-(Max+Avg))	
	Approximation factor		Approximation factor
$\lambda = 0$, MATCHING [8]	$1/(n - m + 1)$	$\lambda = 0$, SAMPLING [15]	$O(\sqrt{n} \log n)$
$\lambda = 0$, ELLIPSOID [7]	$O(\sqrt{n} m^{1/4} \log n \log^{3/2} m)$	$\lambda = 0$, ELLIPSOID [7]	$O(\sqrt{n} \log n)$
$\lambda = 0$, BINSRCH [11]	$1/(2m - 1)$	$\lambda = 1$, GREEDSPLIT† [18, 14]	2
$\lambda = 1$, GREEDWELFARE [5]	$1/2$	$\lambda = 1$, RELAX [2]	$O(\log n)$
$\lambda = 0$, GREEDSAT*	$(1/2 - \delta, \frac{\delta}{1/2+\delta})$	LOVÁSZ ROUND*	m
$0 \leq \lambda \leq 1$, GENERALGREEDSAT*	$\lambda/2$	$0 \leq \lambda \leq 1$ GENERALLOVÁSZ ROUND*	m
$\lambda = 0$, Hardness	$1/2$ [8]	$\lambda = 0$, Hardness*	m
$\lambda = 1$, Hardness	$1 - 1/e$ [16]	$\lambda = 1$, Hardness	$2 - 2/m$ [4]

Table 1: Summary of our contributions and existing work on Problems 1 and 2.[2] See text for details.

Previous Work: Special cases of Problems 1 and 2 have appeared previously. Problem 1 with $\lambda = 0$ is called *submodular fair allocation* (SFA), and Problem 2 with $\lambda = 0$ is called *submodular load balancing* (SLB), robust optimization problems both of which previously have been studied. For SLB even in the homogeneous setting, [15] show that the problem is information theoretically hard to approximate within $o(\sqrt{n/\log n})$. They give a sampling-based algorithm achieving $O(\sqrt{n/\log n})$ for the homogeneous setting. However, the sampling-based algorithm is not practical and scalable since it involves solving, in the worst-case, $O(n^3 \log n)$ instances of submodular function minimization. Another approach approximates each submodular function by its ellipsoid approximation (again non-scalable) and reduces SLB to its modular version (minimum makespan scheduling) leading to an approximation factor of $O(\sqrt{n} \log n)$ [7]. SFA, on the other hand, has been studied mostly in the heterogeneous setting. When f_i's are all modular, the tightest algorithm, so far, is to iteratively round an LP solution achieving $O(1/(\sqrt{m} \log^3 m))$ approximation [1], whereas the problem is NP-hard to $1/2 + \epsilon$ approximate for any $\epsilon > 0$ [8]. When f_i's are submodular, [8] gives a matching-based algorithm with a factor $1/(n - m + 1)$ approximation that performs poorly when $m \ll n$. [11] proposes a binary search algorithm yielding an improved factor of $1/(2m - 1)$. Similar to SLB, [7] applies the same ellipsoid approximation techniques leading to a factor of $O(\sqrt{n} m^{1/4} \log n \log^{3/2} m)$. These approaches are theoretically interesting, but they do not scale to large problems. Problems 1 and 2, when $\lambda = 1$, have also been previously studied. Problem 2 becomes the *submodular multiway partition* (SMP) for which one can obtain a relaxation based 2-approximation [2] in the homogeneous case. In the heterogeneous case, the guarantee is $O(\log n)$ [3]. Similarly, [18, 14] propose a greedy splitting 2-approximation algorithm for the homogeneous setting. Problem 1 becomes the *submodular welfare* [16] for which a scalable greedy algorithm achieves a $1/2$ approximation [5]. Unlike the worst case ($\lambda = 0$), many of the algorithms proposed for these problems are scalable. The general case ($0 < \lambda < 1$) of Problems 1 and 2 differs from either of these extreme cases since we wish both for a *robust* and average case partitioning, and controlling λ allows one to trade off between the two.

Our contributions: In contrast to Problems 1 and 2 in the average case (i.e., $\lambda = 1$), existing algorithms for the worst case ($\lambda = 0$) are not scalable. This paper closes this gap, by proposing new classes of algorithmic frameworks to solve SFA and SLB: (1) greedy algorithms; and (2) a Lovász extension based relaxation algorithm. For SFA under the heterogeneous setting, we propose a "saturate" greedy algorithm (GREEDSAT) that iteratively solves instances of submodular welfare problems. We show GREEDSAT has a bi-criterion guarantee of $(1/2 - \delta, \delta/(1/2 + \delta))$, which ensures at least $\lceil m(1/2 - \delta) \rceil$ blocks receive utility at least $\delta/(1/2 + \delta)OPT$ for any $0 < \delta < 1/2$. For SLB, we first generalize the hardness result in [15] and show that it is hard to approximate better than m for any $m = o(\sqrt{n/\log n})$ even in the homogeneous setting. We then give a Lovász extension based relaxation algorithm (LOVÁSZ ROUND) yielding a tight factor of m for the heterogeneous setting. As far as we know, this is the first algorithm achieving a factor of m for SLB in this setting. Next we show algorithms that handle generalizations of SFA and SLB to Problems 1 and 2. In particular we generalize GREEDSAT leading to GENERALGREEDSAT to solve Problem 1. GENERALGREEDSAT provides a guarantee that smoothly interpolates in terms of λ between the bi-criterion factor by GREEDSAT in the case of $\lambda = 0$ and the constant factor of $1/2$ by the greedy algorithm in the case of $\lambda = 1$. For Problem 2 we introduce GENERALLOVÁSZ ROUND that generalizes LOVÁSZ ROUND and obtain an m-approximation for general λ. The theoretical contributions and the existing work for Problems 1 and 2 are summarized in Table 1.

2 Robust Submodular Partitioning (Problems 1 and 2 when $\lambda = 0$)

Notation: we define $f(j|S) \triangleq f(S \cup j) - f(S)$ as the gain of $j \in V$ in the context of $S \subseteq V$. We assume w.l.o.g. that the ground set is $V = \{1, 2, \cdots, n\}$.

[2] Results obtained in this paper are marked as $*$. Methods for only the homogeneous setting are marked as $\dagger$.

GreedSat for SFA: We give a "greedy" style algorithm – "Saturate" Greedy to solve SFA (GreedSat, see Alg. 1). Similar in flavor to the one proposed in [12] GreedSat defines an intermediate objective $\bar{F}^c(\pi) = \sum_{i=1}^m f_i^c(A_i^\pi)$, where $f_i^c(A) = \frac{1}{m}\min\{f_i(A), c\}$ (Line 2). The parameter c controls the saturation in each block. f_i^c satisfies submodularity for each i. Unlike SFA, the combinatorial optimization problem $\max_{\pi\in\Pi} \bar{F}^c(\pi)$ (Line 6) is much easier and is an instance of the submodular welfare problem (SWP) [16]. In this work, we solve Line 6 by the efficient greedy algorithm as described in [5] with a factor $1/2$. It should be clear that one can also implement a computationally expensive multi-linear relaxation algorithm as given in [16] to solve it with a tight factor $(1 - 1/e)$. Setting the input argument $\alpha = 1/2$ as the approximation factor for Line 6, the essential idea of GreedSat is to perform a binary search over the parameter c to find the largest c^* such that the returned solution $\hat{\pi}^{c^*}$ for the instance of SWP satisfies $\bar{F}^{c^*}(\hat{\pi}^{c^*}) \geq \alpha c^*$. GreedSat terminates in solving $O(\log(\frac{\min_i f_i(V)}{\epsilon}))$ instances of SWP. Theorem 2.1 gives a bi-criterion optimality guarantee.

Algorithm 1: GreedSat

1: Input: $\{f_i\}_{i=1}^m, m, V, \alpha$.
2: Let $\bar{F}^c(\pi) = \frac{1}{m}\sum_{i=1}^m \min\{f_i(A_i^\pi), c\}$.
3: Let $c_{\min} = 0, c_{\max} = \min_i f_i(V)$
4: **while** $c_{\max} - c_{\min} \geq \epsilon$ **do**
5: $\quad c = \frac{1}{2}(c_{\max} + c_{\min})$
6: $\quad \hat{\pi}^c \in \text{argmax}_{\pi\in\Pi} \bar{F}^c(\pi)$
7: $\quad$ **if** $\bar{F}^c(\hat{\pi}^c) < \alpha c$ **then**
8: $\quad\quad c_{\max} = c$
9: $\quad$ **else**
10: $\quad\quad c_{\min} = c; \hat{\pi} \leftarrow \hat{\pi}^c$
11: $\quad$ **end if**
12: **end while**
13: Output: $\hat{\pi}$.

Theorem 2.1. *Given $\epsilon > 0$, $0 \leq \alpha \leq 1$ and any $0 < \delta < \alpha$, GreedSat finds a partition such that at least $\lceil m(\alpha - \delta)\rceil$ blocks receive utility at least $\frac{\delta}{1-\alpha+\delta}(\max_{\pi\in\Pi}\min_i f_i(A_i^\pi) - \epsilon)$.*

For any $0 < \delta < \alpha$ Theorem 2.1 ensures that the top valued $\lceil m(\alpha - \delta)\rceil$ blocks in the partition returned by GreedSat are $(\delta/(1-\alpha+\delta) - \epsilon)$-optimal. δ controls the trade-off between the number of top valued blocks to bound and the performance guarantee attained for these blocks. The smaller δ is, the more top blocks are bounded, but with a weaker guarantee. We set the input argument $\alpha = 1/2$ as the worst-case performance guarantee for solving SWP so that the above theoretical analysis follows. However, the worst-case factor is often achieved by very contrived examples of submodular functions. For the ones used in practice, the greedy algorithm often leads to near-optimal solution [12]. Setting α as the actual performance guarantee for SWP (often very close to 1), the bound can be significantly improved. In practice, we suggest setting $\alpha = 1$.

2.1 Lovász Round for SLB (Problem 2 with $\lambda = 0$)

In this section, we investigate the problem of SLB. Existing hardness result in [15] is $o(\sqrt{n/\log n})$, which is independent of m and implicitly assumes that $m = \Theta(\sqrt{n/\log n})$. However, the applications for SLB are often dependent on m, which is often chosen as $m \ll n$. We offer the hardness analysis in terms of m in the following Theorem.

Theorem 2.2. *For any $\epsilon > 0$, SLB cannot be approximated to a factor of $(1-\epsilon)m$ for any $m = o(\sqrt{n/\log n})$ with polynomial number of queries even under the homogeneous setting.*

For the rest of the paper, we assume $m = o(\sqrt{n/\log n})$ for SLB, unless stated otherwise. Theorem 2.2 implies that SLB is hard to approximate better than m. However, arbitrary partition $\pi \in \Pi$ already achieves the best approximation factor of m that one can hope for under the homogeneous setting, since $\max_i f(A_i^\pi) \leq f(V) \leq \sum_i f(A_i^{\pi'}) \leq m \max_i f(A_i^{\pi'})$ for any $\pi' \in \Pi$.

Lovász Round: Therefore we consider only the heterogeneous setting for SLB, for which we propose a tight algorithm – Lovász Round (see Alg. 2). The algorithm proceeds as follows: (1) apply the Lovász extension of submodular functions to relax SLB to a convex programming, which is exactly solved for a fractional solution (Line 2); (2) map the fractional solution to a partition using the θ-rounding technique as proposed in [9] (Line 3 - 6). The Lovász extension, which naturally connects a submodular function f with its convex relaxation $\tilde{f}$, is defined as follows: given any $x \in [0,1]^n$, we obtain a permutation σ_x by ordering its elements in non-increasing order, and thereby a chain of sets $S_0^{\sigma_x} \subset, \dots, \subset S_n^{\sigma_x}$ with $S_j^{\sigma_x} = \{\sigma_x(1), \dots, \sigma_x(j)\}$

Algorithm 2: Lovász Round

1: Input: $\{f_i\}_{i=1}^m, \{\tilde{f}_i\}_{i=1}^m, m, V$.
2: Solve for $\{x_i^*\}_{i=1}^m$ via convex relaxation.
3: Rounding: Let $A_1 =, \dots, = A_m = \emptyset$.
4: **for** $j = 1, \dots, n$ **do**
5: $\quad \hat{i} \in \text{argmax}_i\, x_i^*(j); A_{\hat{i}} = A_{\hat{i}} \cup j$
6: **end for**
7: Output $\{A_i\}_{i=1}^m$.

for $j = 1, \ldots, n$. The Lovász extension $\tilde{f}$ for f is the weighted sum of the ordered entries of x: $\tilde{f}(x) = \sum_{j=1}^{n} x(\sigma_x(j))(f(S_j^{\sigma_x}) - f(S_{j-1}^{\sigma_x}))$. Given the convexity of the $\tilde{f}_i$'s , SLB is relaxed to the following convex program:

$$\min_{x_1,\ldots,x_m \in [0,1]^n} \max_i \tilde{f}_i(x_i), \text{ s.t } \sum_{i=1}^{m} x_i(j) \geq 1, \text{ for } j = 1, \ldots, n \tag{1}$$

Denoting the optimal solution for Eqn 1 as $\{x_1^*, \ldots, x_m^*\}$, the θ-rounding step simply maps each item $j \in V$ to a block $\hat{i}$ such that $\hat{i} \in \operatorname{argmin}_i x_i^*(j)$. We obtain the bound for LOVÁSZ ROUND as follows:

Theorem 2.3. LOVÁSZROUND *achieves a worst-case approximation factor* m.

We remark that, to the best of our knowledge, LOVÁSZROUND is the first algorithm that is tight and that gives an approximation in terms of m for the heterogeneous setting.

3 General Submodular Partitioning (Problems 1 and 2 when $0 < \lambda < 1$)

Lastly we study Problem 1 and Problem 2, in the most general case, i.e., $0 < \lambda < 1$. We use the proposed algorithms for SFA (Problem 1 with $\lambda = 0$) and SLB (Problem 2 with $\lambda = 1$) as the building blocks to design algorithms for the general scenarios ($0 < \lambda < 1$). We first generalize the proposed GREEDSAT leading to GENERALGREEDSAT for Problem 1. For Problem 2 we generalize LOVÁSZ ROUND to obtain a Lovász extension based algorithm.

Almost the same as GREEDSAT we define an intermediate objective: $\bar{F}_\lambda^c(\pi) = \frac{1}{m}\sum_{i=1}^{m} \min\{\bar{\lambda} f_i(A_i^\pi) + \lambda \frac{1}{m} \sum_{j=1}^{m} f_j(A_j^\pi), c\}$ in GENERALGREEDSAT. Following the same algorithmic design as in GREEDSAT, GENERALGREEDSAT only differs from GREEDSAT in Line 6, where the submodular welfare is defined on the intermediate objective $\bar{F}_\lambda^c(\pi)$. Let α be the guarantee for solving submodular welfare, GENERALGREEDSAT approximates Problem 1 with the following bounds:

Theorem 3.1. *Given* ϵ, α, *and* $0 \leq \lambda \leq 1$, GENERALGREEDSAT *finds a partition* $\hat{\pi}$ *that satisfies the following:* $\bar{\lambda} \min_i f_i(A_i^{\hat{\pi}}) + \lambda \frac{1}{m} \sum_{i=1}^{m} f_i(A_i^{\hat{\pi}}) \geq \lambda\alpha(OPT - \epsilon)$ *where* $OPT = \max_{\pi\in\Pi} \bar{\lambda} \min_i f_i(A_i^\pi) + \lambda \frac{1}{m} \sum_{i=1}^{m} f_i(A_i^\pi)$. *Moreover, let* $F_{\lambda,i}(\pi) = \bar{\lambda} f_i(A_i^\pi) + \lambda \frac{1}{m} \sum_{j=1}^{m} f_j(A_j^\pi)$. *Given any* $0 < \delta < \alpha$, *there is a set* $I \subseteq \{1, \ldots, m\}$ *such that* $|I| \geq \lceil m(\alpha - \delta) \rceil$ *and* $F_{i,\lambda}(\hat{\pi}) \geq \max\{\frac{\delta}{1-\alpha+\delta}, \lambda\alpha\}(OPT - \epsilon), \forall i \in I$.

Theorem 3.1 generalizes Theorem 2.1 when $\lambda = 0$, i.e., it recovers the bi-criterion guarantee in Theorem 2.1 for the worst-case scenario ($\lambda = 0$). Moreover Theorem 3.1 implies that the factor of α for the average-case objective can almost be recovered by GENERALGREEDSAT if $\lambda = 1$. It also gives an improved guarantee as λ increases suggesting that Problem 1 becomes easier as the mixed objective weights more on the average-case objective. We also point out that the optimality guarantee of GENERALGREEDSAT smoothly interpolates the two extreme cases in terms of λ.

Next we focus on Problem 2 for general λ. We generalize LOVÁSZ ROUND leading to GENERALLOVÁSZ ROUND. Almost the same as LOVÁSZ ROUND, GENERALLOVÁSZ ROUND only differs in Line 2, where Problem 2 is relaxed as the following convex program:

$$\min_{x_1,\ldots,x_m \in [0,1]^n} \bar{\lambda} \max_i \tilde{f}_i(x_i) + \lambda \frac{1}{m} \sum_{j=1}^{m} \tilde{f}_j(x_j), \text{ s.t } \sum_{i=1}^{m} x_i(j) \geq 1, \text{ for } j = 1, \ldots, n \tag{2}$$

After solving for the fractional solution $\{x_i^*\}_{i=1}^m$ to the convex programming, GENERALLOVÁSZ ROUND then rounds it to a partition using the same rounding technique as LOVÁSZ ROUND. The following Theorem holds:

Theorem 3.2. GENERALLOVÁSZ ROUND *is guaranteed to find a partition* $\hat{\pi} \in \Pi$ *such that* $\max_i \bar{\lambda} f_i(A_i^{\hat{\pi}}) + \lambda \frac{1}{m} \sum_{j=1}^{m} f_j(A_j^{\hat{\pi}}) \leq m \min_{\pi\in\Pi} \max_i \bar{\lambda} f_i(A_i^\pi) + \lambda \frac{1}{m} \sum_{j=1}^{m} f_j(A_j^\pi)$.

Theorem 3.2 generalizes Theorem 2.3 when $\lambda = 0$. Moreover we achieve a factor of m by GENERALLOVÁSZ ROUND for any λ. Though the approximation guarantee is independent of λ the algorithm naturally exploits the trade-off between the worst-case and average-case objectives in terms of λ.

References

[1] A. Asadpour and A. Saberi. An approximation algorithm for max-min fair allocation of indivisible goods. In *SICOMP*, 2010.

[2] C. Chekuri and A. Ene. Approximation algorithms for submodular multiway partition. In *FOCS*, 2011.

[3] C. Chekuri and A. Ene. Submodular cost allocation problem and applications. In *Automata, Languages and Programming*, pages 354–366. Springer, 2011.

[4] A. Ene, J. Vondrák, and Y. Wu. Local distribution and the symmetry gap: Approximability of multiway partitioning problems. In *SODA*, 2013.

[5] M. Fisher, G. Nemhauser, and L. Wolsey. An analysis of approximations for maximizing submodular set functions—II. In *Polyhedral combinatorics*, 1978.

[6] S. Fujishige. *Submodular functions and optimization*, volume 58. Elsevier, 2005.

[7] M. Goemans, N. Harvey, S. Iwata, and V. Mirrokni. Approximating submodular functions everywhere. In *SODA*, 2009.

[8] D. Golovin. Max-min fair allocation of indivisible goods. *Technical Report CMU-CS-05-144*, 2005.

[9] R. Iyer, S. Jegelka, and J. Bilmes. Monotone closure of relaxed constraints in submodular optimization: Connections between minimization and maximization: Extended version.

[10] S. Jegelka and J. Bilmes. Submodularity beyond submodular energies: coupling edges in graph cuts. In *CVPR*, 2011.

[11] S. Khot and A. Ponnuswami. Approximation algorithms for the max-min allocation problem. In *APPROX*, 2007.

[12] A. Krause, B. McMahan, C. Guestrin, and A. Gupta. Robust submodular observation selection. In *JMLR*, 2008.

[13] M. Li, D. Andersen, and A. Smola. Graph partitioning via parallel submodular approximation to accelerate distributed machine learning. In *arXiv preprint arXiv:1505.04636*, 2015.

[14] M. Narasimhan, N. Jojic, and J. A. Bilmes. Q-clustering. In *NIPS*, 2005.

[15] Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *FOCS*, 2008.

[16] J. Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, 2008.

[17] K. Wei, R. Iyer, and J. Bilmes. Submodularity in data subset selection and active learning. In *ICML*, 2015.

[18] L. Zhao, H. Nagamochi, and T. Ibaraki. On generalized greedy splitting algorithms for multiway partition problems. *Discrete applied mathematics*, 143(1):130–143, 2004.