Approximation Algorithms - Homework 2

Ilay Menahem, and Aaron Ross

December 29, 2025

Problem 1: Weighted Max-Cut Local Search

(a) Polynomial Convergence

Overview

We analyze the $LS_{Max-Cut}$ algorithm where a local move is made only if the weight of the cut increases by a factor of at least $(1+\frac{\epsilon}{n})$. We show that the number of iterations is polynomial in the input size and $1/\epsilon$.

Proof of Validity

Let $W(S)$ denote the weight of the cut $(S, V \setminus S)$. Let S_k be the cut after k iterations, and S_0 be the initial cut. The algorithm updates S_k to S_{k+1} only if:

$$W(S_{k+1}) \geq \left(1+\frac{\epsilon}{n}\right) W(S_k)$$

Therefore, after k iterations:

$$W(S_k) \geq \left(1+\frac{\epsilon}{n}\right)^k W(S_0)$$

The maximum possible weight of any cut is bounded by the total weight of all edges, denoted W_{total}. Thus:

$$\left(1+\frac{\epsilon}{n}\right)^k W(S_0) \leq W(S_k) \leq W_{total}$$

Complexity Analysis

Taking the natural logarithm of the inequality:

$$k \ln\left(1+\frac{\epsilon}{n}\right) + \ln(W(S_0)) \leq \ln(W_{total})$$

Using the inequality $\ln(1+x) \approx x$ for small x (specifically $\ln(1+x) \geq \frac{x}{1+x}$ or simply bounding by linear growth for complexity):

$$k \cdot \frac{\epsilon}{2n} \leq k \ln\left(1+\frac{\epsilon}{n}\right) \leq \ln(W_{total}) - \ln(W(S_0)) = \ln\left(\frac{W_{total}}{W(S_0)}\right)$$

$$k \leq \frac{2n}{\epsilon} \ln \left(\frac{W_{total}}{W(S_0)} \right)$$

Since $\ln(W_{total})$ is polynomial in the input size (number of bits to represent weights), k is polynomial.

(b) Improved Bound with Specific Initialization

Algorithm Modification

Initialize $S_0 = \{v^*\}$, where $v^* = \arg\max_{v \in V} w(\{v\}, V \setminus \{v\})$.

Complexity Analysis

1. **Lower Bound of Initialization:** Let $w(v)$ be the sum of weights of edges incident to v. We know $\sum_{v \in V} w(v) = 2W_{total}$. By the Pigeonhole Principle, there exists a vertex v^* such that:

$$w(v^*) \geq \frac{2W_{total}}{n}$$

The initial cut weight is $W(S_0) = w(v^*)$, so $W(S_0) \geq \frac{2W_{total}}{n}$.

2. **Upper Bound on Iterations:** The optimal cut W_{OPT} is bounded by W_{total}. Substituting this into the iteration bound from part (a):

$$k \leq O\left(\frac{n}{\epsilon}\right) \ln \left(\frac{W_{total}}{2W_{total}/n} \right) = O\left(\frac{n}{\epsilon}\right) \ln \left(\frac{n}{2}\right) = O\left(\frac{n \log n}{\epsilon}\right)$$

Thus, the number of iterations is $O(\frac{n \log n}{\epsilon})$.

Problem 2: Hitting Set

Algorithm: Recursive Local Ratio

The algorithm takes a universe U, a collection of sets $\mathcal{C}$, and weights w.

Algorithm 1 Local Ratio for Hitting Set

function HittingSet($U, \mathcal{C}, w$)
  Remove all $x \in U$ with $w(x) \leq 0$ (add to solution H).
  if $\mathcal{C} = \emptyset$ **then return** $\emptyset$
  end if
  Pick any set $S_i \in \mathcal{C}$.
  Let $\epsilon = \min_{x \in S_i} w(x)$.
  Define $w_1(x) = \begin{cases} \epsilon & \text{if } x \in S_i \\ 0 & \text{otherwise} \end{cases}$
  Define $w_2(x) = w(x) - w_1(x)$.
  $H' \leftarrow$ HittingSet($U, \mathcal{C}, w_2$)
  if $H' \cap S_i = \emptyset$ **then**
    Find $x \in S_i$ such that $w_2(x) = 0$ (must exist as we subtracted min).
    $H \leftarrow H' \cup \{x\}$
  else
    $H \leftarrow H'$
  end if
  return H
end function

Approximation Analysis

We prove that $w(H) \leq S_{max} \cdot w(H_{OPT})$, where $S_{max} = \max_i |S_i|$. According to the Local Ratio Theorem, it suffices to show the approximation holds for w_1 and w_2.

Proof. **Inductive Step:** Assume the algorithm is an S_{max}-approximation for w_2.
For w_1:

- The optimal solution H_{OPT} must pick at least one element from S_i to cover it. Thus, $w_1(H_{OPT}) \geq \epsilon$.
- Our solution H adds at most one element specifically for S_i (if not already covered). However, even if we consider the set of all chosen elements, strictly speaking, $w_1(H) = \epsilon \cdot |H \cap S_i|$.
- While $|H \cap S_i|$ could be large in theory, the standard Local Ratio analysis for covering problems relies on comparing the cost paid to the optimum.
- A simpler view: The total weight removed from the system is ϵ per element in S_i. The optimal solution pays at least ϵ.

- The algorithm ensures minimality with respect to the decomposition.
- Specifically for Hitting Set (dual of Set Cover with frequency $f = S_{max}$), any minimal solution is an S_{max}-approximation with respect to w_1.

$$w_1(H) \leq |S_i| \cdot \epsilon \leq S_{max} \cdot \epsilon \leq S_{max} \cdot w_1(H_{OPT})$$

Total:

$$w(H) = w_1(H) + w_2(H) \leq S_{max} w_1(H_{OPT}) + S_{max} w_2(H_{OPT}) = S_{max} \cdot w(H_{OPT})$$

□

Problem 3: Multiple Choice Maximum Coverage

Overview

We prove the local search algorithm is a $\frac{1}{2}$-approximation. Let $S = (j_1, \ldots, j_k)$ be the local optimum solution and $O = (o_1, \ldots, o_k)$ be the global optimal solution.

Definitions (from Hint)

Let $U_S = \bigcup_{i=1}^{k} S_{i,j_i}$ be the set of elements covered by our solution. For each $1 \leq i \leq k$, define:

$$A_i = S_{i,j_i} \setminus \bigcup_{r \neq i} S_{r,j_r} \quad \text{(Unique contribution of set } i \text{ in } S\text{)}$$

$$B_i = S_{i,o_i} \setminus \bigcup_{r \neq i} S_{r,j_r} \quad \text{(Gain of switching set } i \text{ to optimal choice given others fixed)}$$

Proof

1. **Local Optimality:** Since the algorithm terminates, switching any single index j_i to o_i does not improve the value. The loss of removing S_{i,j_i} is exactly $|A_i|$. The gain of adding S_{i,o_i} (relative to the other current sets) is exactly $|B_i|$.

$$|B_i| - |A_i| \leq 0 \implies |B_i| \leq |A_i|$$

2. **Summation:** Summing over all $i = 1 \ldots k$:

$$\sum_{i=1}^{k} |B_i| \leq \sum_{i=1}^{k} |A_i|$$

   Note that $\{A_i\}$ are disjoint subsets of U_S, and their union is exactly U_S (since every element in the union must be unique to some set or shared; if shared, it is not lost when removing one, but A_i captures the "ownership"). Actually, more precisely: $\sum |A_i| \leq |U_S| = V(S)$. In fact, $\sum |A_i| = V(S)$ is not strictly true if there is overlap, but $\sum |A_i| \leq V(S)$ holds. Wait, let us refine the definition of A_i. The hint defines A_i as the contribution of set i *given the others*. The sets A_i are disjoint. $\bigcup A_i \subseteq U_S$. Thus $\sum |A_i| \leq V(S)$.

3. **Bounding the Optimum:** Let $U_{OPT} = \bigcup S_{i,o_i}$. Consider any element $x \in U_{OPT} \setminus U_S$. Since x is not in U_S, it is not in any S_{r,j_r} for any r. Since $x \in U_{OPT}$, there exists some i such that $x \in S_{i,o_i}$. Therefore, $x \in S_{i,o_i} \setminus \bigcup_{r \neq i} S_{r,j_r}$, which means $x \in B_i$. This implies:

$$U_{OPT} \setminus U_S \subseteq \bigcup_{i=1}^{k} B_i$$

$$|U_{OPT}| - |U_S| \leq |U_{OPT} \setminus U_S| \leq \sum_{i=1}^{k} |B_i|$$

4. **Conclusion:** Combining the inequalities:

$$|U_{OPT}| - V(S) \leq \sum_{i=1}^{k} |B_i| \leq \sum_{i=1}^{k} |A_i| \leq V(S)$$

$$|U_{OPT}| \leq 2V(S) \implies V(S) \geq \frac{1}{2} V(OPT)$$

Problem 4: Generalized Steiner Forest (GSF)

(a) Minimal GSF is a 2-approximation for w'

Claim: Let $w'(u,v) = |\{u,v\} \cap T|$. Any minimal GSF F satisfies $w'(F) \leq 2w'(F_{OPT})$.

Proof. **Lower Bound (OPT):** Consider the optimal solution F^*. Since $|T_i| \geq 2$, every terminal $v \in T$ must be connected to at least one other terminal. Thus, every $v \in T$ has degree at least 1 in F^*.

$$w'(F^*) = \sum_{(u,v)\in F^*} |\{u,v\} \cap T| = \sum_{v\in T} \deg_{F^*}(v) \geq \sum_{v\in T} 1 = |T|$$

Upper Bound (Minimal Forest F): Since F is a minimal GSF, every leaf in every connected component of F must be a terminal in T. (If a leaf were not a terminal, the incident edge could be removed). For any tree in F, the sum of degrees of the vertices in T is strictly less than $2 \times$ (number of vertices in T in that tree). Summing over all components:

$$w'(F) = \sum_{v\in T} \deg_F(v) \leq 2|T| - 2 \times (\#\text{components}) < 2|T|$$

Ratio:

$$w'(F) < 2|T| \leq 2w'(F^*)$$

□

(b) Local Ratio Algorithm

1. **Base Case:** If all connectivity requirements are satisfied by 0-weight edges, return the set of those edges (pruned to be minimal).
2. **Weight Definition:** Define $w'(e) = |e \cap T|$.
3. **Compute ϵ:** Let $\epsilon = \min\{\frac{w(e)}{w'(e)} \mid w'(e) > 0, e \in E\}$.
4. **Decomposition:**
$$w_1(e) = \epsilon \cdot w'(e)$$
$$w_2(e) = w(e) - w_1(e)$$
5. **Recursion:** Solve recursively for w_2 to obtain forest F'.
6. **Pruning (Crucial):** F' is a valid GSF. Remove edges from F' one by one as long as the connectivity requirements T_i remain satisfied. Let the result be F.
7. **Return** F.

Approximation: By the Local Ratio Theorem, since F is a minimal GSF, it is a 2-approximation for w_1 (by part a). By induction, it is a 2-approximation for w_2. Thus, it is a 2-approximation for w.

Problem 5: Knapsack with Partition Constraints

Overview

We first define a Dynamic Programming (DP) algorithm that runs in pseudo-polynomial time, then apply the scaling technique to obtain an FPTAS.

Dynamic Programming

Let the items be ordered such that items in S_1 appear first, then S_2, etc. We compress the state to handle one group at a time.

- $D_{prev}[v]$: Min weight to achieve value v using previous groups.
- $D_{curr}[c][v]$: Min weight to achieve value v using previous groups AND exactly c items from the current group.

Transitions: For each group S_j with limit $k(j)$:

1. Initialize $D_{curr}[0][v] = D_{prev}[v]$.
2. For each item $i \in S_j$ (weight w_i, value v_i):

$$D_{curr}[c][v] = \min\left(D_{curr}[c][v], \quad D_{curr}[c-1][v-v_i] + w_i\right)$$

   (Iterate c from $k(j)$ down to 1).

3. After processing all items in S_j, update D_{prev}:

$$D_{prev}[v] = \min_{0 \le c \le k(j)} D_{curr}[c][v]$$

The complexity is $O(n \cdot n \cdot V_{total}) = O(n^2 V_{total})$ since the inner c loop runs at most n times.

FPTAS Construction

To achieve a $(1-\epsilon)$-approximation:

1. Let $V_{max} = \max_i v_i$.
2. Define scaling factor $K = \frac{\epsilon V_{max}}{n}$.
3. Define scaled values $v_i' = \lfloor \frac{v_i}{K} \rfloor$.
4. Run the DP algorithm using weights w_i and values v_i'.
5. Output the set of items corresponding to the optimal DP state satisfying weight $\le B$.

Complexity and Bound

The maximum possible scaled value is bounded by:

$$V'_{total} \leq \sum v'_i \leq n \cdot \frac{V_{max}}{K} = n \cdot \frac{n}{\epsilon} = \frac{n^2}{\epsilon}$$

The DP running time becomes:

$$O\left(n^2 \cdot \frac{n^2}{\epsilon}\right) = O\left(\frac{n^4}{\epsilon}\right)$$

This is polynomial in n and $1/\epsilon$.

Approximation Proof: For the optimal set P^*, the difference between the true value and the scaled value is:

$$\sum_{i \in P^*} v_i - K \sum_{i \in P^*} v'_i < \sum_{i \in P^*} K = |P^*|K \leq nK = \epsilon V_{max} \leq \epsilon OPT$$

Thus, the solution returned is at least $(1 - \epsilon)OPT$.