Parallel Algorithms for Submodular Function Maximization

Chandra Chekuri
Univ. of Illinois, Urbana-Champaign

Joint work with Kent Quanrud (UIUC → Purdue)

RIMS Kyoto, Jan 13, 2020
Based on two recent papers

- Submodular Function Maximization in Parallel via the Multilinear Relaxation (SODA 2019)
- Parallelizing Greedy for Submodular Set Function Maximization in Matroids and Beyond (STOC 19)
Max k-Cover

- \( U = \{1, 2, \ldots, m\} \)
- \( X_1, X_2, \ldots, X_n \) each a subset of \( U \)

**Max k-Cover:** Find \( k \) sets from \( X_1, X_2, \ldots, X_n \) to maximize union of chosen sets
Submodular Function Maximization

Max $k$-Cover is a special case of more general problem

$$\max \ f(S) \quad |S| \leq k$$
A real valued set function $f: 2^N \rightarrow R$ is submodular if

$$f(A+j) - f(A) \geq f(B+j) - f(B) \text{ for all } A \subseteq B \text{ and } j \not\in B$$

Equivalently

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \text{ for all } A, B \subseteq N$$
Coverage in Set Systems

- $X_1, X_2, \ldots, X_n$ subsets of set $U$
- $N = [n]$ and $f(A) = | \cup_{i \in A} X_i |$
Cut functions in graphs

- $G=(V,E)$ undirected graph
- $f: 2^V \rightarrow R^+$ where $f(S) = |\delta(S)|$
Submodular Set Functions

- **Monotone**: \( f(A) \leq f(B) \) for all \( A \subseteq B \)
  
  Example: coverage in set systems, matroid rank function

- **Non-negative**: \( f(A) \geq 0 \) for all \( A \)
  
  Example: cut function of directed graphs

- **Symmetric**: \( f(A) = f(N \setminus A) \) for all \( A \)
  
  Example: cut function of undirected graphs

This talk: focus on monotone functions
Submodular Function Maximization

$$\max \ f(S) \quad |S| \leq k$$

- $S \leftarrow \emptyset$
- While ($|S| \leq k$)
  - $j \leftarrow \arg\max_i f(S \cup \{i\}) - f(S)$
  - $S \leftarrow S \cup \{i\}$
- Output $S$

**Theorem:** [Nemhauser-Wolsey-Fisher’78] Greedy yields a $(1-1/e)$ approximation for monotone $f$.

**Optimal:** under $P \neq NP$, poly-time oracle calls
Constrained Submodular Function Maximization

\[ \max \ f(S) \]
\[ |S| \leq k \]

\[ \max \ f(S) \]
\[ S \text{ is independent set in a matroid} \]

\[ \max \ f(S) \]
\[ S \text{ is independent in intersection of } p \text{ matroids} \]

\[ \max \ f(S) \]
\[ A 1_S \leq b \]

\( f \) provided as a value
oracle: given \( S \),
oracle provides \( f(S) \)
Submodular Function Maximization

Classical topic from late 70’s [Cornuejols-Fisher-Nemhauser-Wolsey …]

Performance of Greedy for monotone:

- $(1-1/e)$ for cardinality constraint
- $\frac{1}{2}$ for matroid constraint and $\frac{1}{p+1}$ for intersection of $p$ matroids and more generally for a $p$-system constraint
Submodular Function Maximization

- **Much progress** in (approximation) algorithms and hardness in the last 15 years.
  - Non-negative functions
  - Local search
  - Math. programming approach via multilinear relaxation

- **Many new applications:** algorithmic game theory, machine learning, data summarization and others

- **Ongoing work** …

**CAVEAT:** will not be able to mention many refs
Mathematical programming approach

For submodular function maximization [Calinescu-C-Pal-Vondrak’07]

- **Initial goal**: to improve $\frac{1}{2}$ for matroid constraint to $(1-\frac{1}{e})$
- Necessary for handling more complicated sets of constraints
- Many developments including structural results, lower bounds, concentration properties, …
Math. Programming approach

\[
\max f(S) \quad \text{S is feasible}
\]

\[
\max F(x) \quad x \in P
\]

\[F(x)\] is the multilinear extension of \( f \) to \([0,1]^N\)

Round \( x \)
Can we parallelize submodular function maximization?

Roughly same as: are there low adaptivity algorithms for submodular function maximization?

Greedy, local search and math programming approach appear inherently sequential
Prior work

- First parallel approx. algorithm for Set Cover [Berger-Rompel-Shor’89]

- Parallel positive LP solving [Luby-Nisan’93] improved by [Young’00] which also leads to algorithms for Set Cover

- Parallel for. Max k-Cover not considered explicitly until more recently [Chierchetti-Kumar-Tomkins’10] but ideas implicit in Set Cover parallel algorithms
Adaptivity Model

- Assume value oracle for $f$
- In *each round* can query $f$ in parallel on poly sets
- Rest of computation is free
- **Measure**: how many rounds needed?

Assuming $f$ can be evaluated in parallel, adaptivity is close to PRAM/NC models within poly-log factors
Cardinality constraint

\[
\max f(S) \\
|S| \leq k
\]

[Balkanski-Singer’18] initiated study of adaptivity for submodular set function maximization.

- \(\frac{1}{3}\) approx. for cardinality in \(O(\log n)\) rounds
- Lower bound of \(\Omega(\log n / \log \log n)\) rounds even for an \(\Omega(1/\log n)\) approximation
Recent Work on Parallel Algorithms

- \((1 - 1/e - \varepsilon)\) approximation in \(O(\log n/\varepsilon^2)\) rounds [Balkanski-Rubinstein-Singer’18, Ene-Nguyen’18, Fahrbach-Mirrokni-Zadimoghaddam’18] [C-Quanrud’18]

- [C-Quanrud’18] \((1 - 1/e - \varepsilon)\) approximation for maximizing \(F(x)\) over explicit given packing constraints

- Better dependence on \(\varepsilon\) and non-monotone functions for packing constraints [Ene-Nguyen-Vladu’18]

- Parallel algorithms for matroids and intersections [Balkanski-Rubinstein-Singer’18, C-Quanrud’18, Ene-Nguyen-Vladu’18]

- Parallel Double Greedy: \((1/2 - \varepsilon)\) approx. for unconstrained non-neg functions [Chen-Feldman-Karbasi’18, Ene-Nguyen-Vladu’18]

**Bottom line:** Parallel approx. algorithms for most constraints of interest
Our Approach

[C-Quanrud’18]

• Continuous optimization approach based on multilinear relaxation. Clean and simple

• Generalizes to packing constraints

• Inspired subsequent work on matroids etc
Parallel Algorithm

\[\max f(S) \quad |S| \leq k\]

\[\max F(x) \quad x_1 + x_2 + \cdots + x_n \leq k\]

Can round \( x \) without loss for matroid constraint [CCPV’07]
Multilinear extension of $f$

[Calinescu-C-Pal-Vondrak’07] inspired by [Ageev-Sviridenko’04]

For $f : 2^N \to \mathbb{R}^+$ define $F : [0,1]^N \to \mathbb{R}^+$

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in [0,1]^N$

$R$: random set, include $i$ independently with prob. $x_i$

$F(\mathbf{x}) = \mathbb{E}[ f(R) ] = \sum_{S \subseteq N} f(S) \prod_{i \in S} x_i \prod_{i \in N \setminus S} (1-x_i)$
Notation

- \( F'(x) \) the gradient of \( F \) at \( x \).
- \( F'(x) = \frac{\partial F(x)}{\partial x_i} \)
- \( \langle u, v \rangle = \sum_i u_i v_i \) is inner product of two vectors
- \( u \lor v \) is new vector such that \( (u \lor v)_i = \max(u_i, v_i) \)
Properties of $F$

- $F$ is *neither concave nor convex*
- $F(x)$ and $F'(x)$ can be estimated by random sampling
- $F$ is a *smooth* submodular function
  - $\frac{\partial^2 F}{\partial x_i \partial x_j} \leq 0$ for all $i,j$.
  - Recall $f(A+j) - f(A) \geq f(A+i+j) - f(A+i)$ for all $A, i, j$
  - $\frac{\partial^2 F}{\partial x_i^2} = 0$ for all $i$ (multilinearity)
  - $F$ is concave along any *non-negative* direction vector

- If $f$ is monotone: $\frac{\partial F}{\partial x_i} \geq 0$ for all $i$, and $F(y) \geq F(x)$ if $y \geq x$
Key property of $F$

$F$ is concave along any *non-negative* direction vector $V$

$$\langle F'(x), d \rangle \geq F(x + d) - F(x)$$
Sequential Greedy

\[ \max F(x) \]

\[ x_1 + x_2 + \cdots + x_n \leq k \]

- \( x \leftarrow 0 \)
- While (\( \sum x_i \leq k \)) do
  - \( j \leftarrow \text{argmax}_n F'_h(x) \)
  - \( x \leftarrow x + \delta e_j \)
- Output \( x \)

[Theorem: For monotone \( f \), \( F(x) \geq (1-1/e) \text{OPT} \)]

[Wolsey’82, Vondrak’08]
Let $x^*$ be optimum fractional soln.

$x$ is current point

$$\langle F'(x), (x \lor x^* - x) \rangle \geq F(x \lor x^*) - F(x) \geq OPT - F(x)$$
Analysis

\[ \langle F'(x), (x \lor x^* - x) \rangle \geq F(x \lor x^*) - F(x) \geq \text{OPT} - F(x) \]

By monotonicity \( F'_j(x) \geq 0 \) for all \( j \)

\[ \sum_j F'_j(x) x_j^* \geq \langle F'(x), (x \lor x^* - x) \rangle \geq \text{OPT} - F(x) \]

And \( \sum_j x_j^* \leq k \) and hence

\[ \max_n F'_n(x) \geq \frac{\text{OPT} - F(x)}{k} \]
Analysis

\[ F'_{j}(x) \geq \frac{OPT - F(x)}{k} \]

\[ F(x + \delta e_{j}) - F(x) = F'_{j}(x) \delta \geq \delta \frac{OPT - F(x)}{k} \]

Let \( t = \sum_{i} x_{i} \)

\[ \frac{dF(x)}{dt} \geq \frac{OPT - F(x)}{k} \]

Solving differential equation with initial condition \( F(0) = 0 \), at \( t = k \),

\[ F(x) \geq \left( 1 - \frac{1}{e} \right) OPT \]
Analysis

\[
\frac{dF(x_t)}{dt} \geq \frac{OPT - F(x_t)}{k}
\]

\[y_t = OPT - F(x_t)\]

\[
\frac{dy_t}{dt} \leq -\frac{y_t}{k} \Rightarrow \int^t_0 \frac{dy_t}{y_t} \leq \int^t_0 -\frac{dt}{k}
\]

\[
\ln \frac{y_t}{y_0} \leq -\frac{t}{k} \Rightarrow y_t \leq y_0 \exp\left(-\frac{t}{k}\right)
\]

\[
\Rightarrow F(x_t) \geq OPT \left(1 - \exp\left(-\frac{t}{k}\right)\right)
\]
Continuous Greedy

- For cardinality constraint continuous-greedy boils down to standard greedy ([Wolsey’82])
- Power comes from its version for arbitrary polytope ([Vondrak’08])
Parallel Algorithm

\[ \max \ F(x) \]

\[ x_1 + x_2 + \cdots + x_n \leq k \]

**Simple Idea:** *Simultaneously* increase all coordinates with best gradient as much as possible
Parallel Algorithm

- $x \leftarrow 0$
- While $(\sum_j x_j \leq k)$ do
  - $\lambda \leftarrow \max_{h} F'_h(x)$  // current best gradient
  - repeat
    - $S \leftarrow \{ j \mid F'_j(x) \geq (1 - \epsilon) \lambda \}$  // good co-ords
    - $x \leftarrow x + \delta S$  // increase all of them uniformly
  - Until ($S = \emptyset$)
- Output $x$

How should we choose $\delta$? Does it have any advantages?
Parallel Algorithm

- $x \leftarrow 0$
- While ($\sum j x_j \leq k$) do
  - $\lambda \leftarrow \max F'_h(x)$ // current best gradient
  - repeat
    - $S \leftarrow \{ j \mid F'_j(x) \geq (1 - \epsilon) \lambda \}$ // good co-ords
    - Choose max $\delta$ such that
      - $F(x + \delta S) - F(x) \geq (1 - 2\epsilon) \lambda \delta |S|$
      - $x \leftarrow x + \delta S$ // increase all of them uniformly
  - Until ($S = \emptyset$)
- Output $x$

Choose greedy step $\delta$ as large as possible such that $S$ remains good through the step
Consequences of greedy step size

• All coordinates that are simultaneously increased are good *throughout* greedy step: analysis of approximation ratio is similar to that of sequential algorithm: \((1-1/e - \varepsilon)\)

• Main issue: # of iterations/depth
Consequences of greedy step size

**Claim:** After greedy choice $|S|$ goes down by $(1 - \varepsilon)$ factor

- $\lambda$ decreases by $(1 - \varepsilon)$ factor each time
- $\lambda$ starts at $\text{OPT}$ and can stop when $\lambda < \text{OPT}/n$

Thus total # of iterations is $O((\log n)^2 / \varepsilon^2)$

Can improve to $O((\log n) / \varepsilon^2)$
Consequences of greedy step size

**Claim:** After greedy choice $|S|$ goes down by $(1 - \epsilon)$ factor

$$ S = \{ j \mid F_j'(x) \geq (1 - \epsilon)\lambda \} \quad y = x + \delta S $$

$$ S' = \{ j \mid F_j'(y) \geq (1 - \epsilon)\lambda \} $$

If $|S'| > (1 - \epsilon) |S|$ then for some $\delta' > \delta$ we have

$$ F(x + \delta'S') - F(x) \geq (1 - \epsilon)\lambda \delta'|S'| \geq (1 - \epsilon)^2 \lambda \delta'|S| $$

$$ > (1 - 2\epsilon)\lambda \delta'|S| $$

Submodularity and monotonicity, contradicting choice of $\delta$. 
• Algorithm is deterministic assuming $F'$ is available
• Greedy step can implemented via binary search
• Can convert continuous algorithm into randomized combinatorial algorithm relatively easily
• Generalizes relatively easily to knapsack constraint. Implies parallel algorithm for submodular set cover
A Combinatorial Algorithm

- \( I \leftarrow \emptyset \)
- \( \text{While } ( |I| \leq k ) \text{ do} \)
  - \( \lambda \leftarrow \max_{h} f_{Q}(h) \)
  - \( \text{repeat} \)
    - \( S \leftarrow \{ j \mid f_{I}(j) \geq (1 - \epsilon) \lambda \} \)
    - \( I \leftarrow I \cup Q \) where \( Q \) samples \( \delta S \)
      where \( \delta \) is greedy choice
    - \( \text{Until } ( S = \emptyset ) \)
- Output \( I \)
Generalizing

- Multiple Packing Constraints
- Matroid Constraint(s)
Multiple Packing Constraints

\[
\begin{align*}
\text{max } & \ F(x) \\
Ax & \leq b
\end{align*}
\]

\[
\begin{align*}
\text{max } & \ F(x) \\
A_{11}x_1 + A_{12}x_2 + \cdots + A_{1n}x_n & \leq 1 \\
\vdots & \\
A_{m1}x_1 + A_{m2}x_2 + \cdots + A_{mn}x_n & \leq 1
\end{align*}
\]

Captures many constraints of interest:
- Knapsack constraints
- Partition, Laminar matroids
- Matchings and generalizations
Multiple Packing Constraints

- Adapt multiplicative weight update (MWU) techniques to “essentially” reduce to single cardinality constraint

- [C-Jayram-Vondrak 2015] adapted MWU in sequential setting to multilinear relaxation

- [C-Quanrud’18] combine ideas from parallel algorithms for positive LP solving due to [Luby-Nisan’94, Young’00] and combine with [C-J-V’15] and cardinality constraint insights

**Theorem:** $O\left( \frac{\log m \log n}{\varepsilon^2} \right)$ rounds for $(1 - 1/e - \varepsilon)$ approximation for monotone $f$
Greedy gives $\frac{1}{2}$ approximation

Continuous greedy plus rounding gives $(1-1/e)$
Matroid Constraint

\[
\max f(S)
\]
\(S \text{ is independent in matroid } \mathcal{M}\)

\[
\max F(x)
\]
\(x \in P(\mathcal{M})\)

Can we parallelize?

**Assumption:** parallel rank/span oracle for matroid available
Can we parallelize?

**Idea:** *Simultaneously* increase all coordinates with best gradient as much as possible *but coordinates need to be independent in matroid!*

\[ \max f(S) \]

\[ S \text{ is independent in matroid } \mathcal{M} \]

\[ \max F(x) \]

\[ x \in P(\mathcal{M}) \]
Combinatorial Algorithm inspired by continuous approach

- $I \leftarrow \emptyset$, $R \leftarrow \emptyset$, $\lambda = \max_j f(j)$
- While ($\lambda \geq \text{OPT}/\text{poly}(n)$) do
  - repeat
    - $S \leftarrow \{ j \in N \setminus \text{span}(R) \mid f_R(j) \geq (1 - \varepsilon) \lambda \}$
    - $Q \leftarrow \text{sample } \delta S \text{ for greedy step size } \delta$
    - $Q' \leftarrow \{ j \in Q \mid j \text{ not in span}(R + Q - j) \}$
    - $I \leftarrow I \cup Q'$
    - $R \leftarrow R \cup Q$
  - Until ($S = \emptyset$)
  - $\lambda \leftarrow (1 - \varepsilon) \lambda$

- Output $I$
Greedy Step

Choose max \( \delta \) such that

- \( E[\|\{ j \in S \mid f_{RUQ}(j) \leq (1 - \epsilon)\lambda\}\|] \leq \epsilon |S| \)
- \( E[|span(Q \cup R)|] \leq \epsilon |S| \)
Algorithm explained

• Maintain two sets \( I \) and \( R \) where \( I \) is independent, \( I \subseteq R \), and \( |I| \geq (1 - \epsilon)|R| \) in expectation.

• Try to add many good elements to \( I \) at the same time. \( Q \) is the set of good elements in terms of marginal value. Suppose we sample \( Q \) with very small \( \delta \). Prune \( Q \) to make it independent to add.

• Pruning may make \( |Q| \) is small. As \( \delta \) increases, marginal values may not be simultaneously good.

• Greedy step ensures that we make good progress in reducing \( \lambda \) or span a lot of good elements for current \( \lambda \)
Theorem: [C-Quanrud’19] Algorithm yields \( \left( \frac{1}{2} - \epsilon \right) \) approximation in expectation in \( \mathcal{O} \left( \frac{\log^2 n}{\epsilon^2} \right) \) adaptive rounds.

Amplification using multilinear extension: Using approach of [Badanidiyuru-Vondrak’14] can amplify \( \left( \frac{1}{2} - \epsilon \right) \) to near-optimal \( (1 - 1/e - \epsilon) \) approx. to value in \( \mathcal{O} \left( \frac{\log^2 n}{\epsilon^3} \right) \) adaptive rounds.
Other Results

- Can generalize from monotone to non-negative functions using some ideas from sequential setting
- Handle multiple matroids and matchoid constraints etc.

Many results from sequential setting can be generalized to parallel setting with comparable approximation ratios.
Continuous approach for submodular function maximization has proved to be very successful. This talk showed utility in parallelization as well.

**Some open directions:**

- Is randomization necessary?
- Can we extend continuous approach to relaxed notions submodularity and other classes of functions?
- Multilinear relaxation approach can be slow due to sampling. Fast algorithms that can avoid this bottleneck?
THANKS