Hogwild Double Greedy Submodular Maximization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Hogwild bidirectional greedy submodular maximization adds a term to the approximation. That is, hogwild does worse than the sequential algorithm by an additive, but not multiplicative, term in the approximation.

1 Introduction

The bidirectional greedy algorithm [1] gives an approximation of $E[F(A)] = 1/2 f(OPT)$, where A is the algorithm output, and OPT is an optimal solution. The hogwild algorithm can give an approximation of $E[F(A)] = \frac{1}{2}F(OPT) - \frac{1}{4}\sum_i E[\rho_i]$, where ρ_i is the maximum difference in the marginal gain that may result from not knowing the full information when deciding whether to include or exclude element i.

2 Submodular maximization

3 Algorithm

3.1 Sequential

Algorithm 1: Serial submodular maximization

$A^0 = \emptyset$, $B^0 = V$
for $i = 1$ *to* n **do**
  $\Delta_+(i) = F(A^{i-1} \cup i) - F(A^{i-1})$
  $\Delta_-(i) = F(B^{i-1} \backslash i) - F(B^{i-1})$
  Draw $u_i \sim Unif(0,1)$
  if $u_i < \frac{[\Delta_+(i)]_+}{[\Delta_+(i)]_+ + [\Delta_-(i)]_+}$ **then**
    $A^i := A^{i-1} \cup i$;
    $B^i := B^{i-1}$
  else
    $A^i := A^{i-1}$;
    $B^i := B^{i-1} \backslash i$

Algorithm 2: Hogwild bidirectional greedy

For each $e \in V$, $\hat{A}(e) = 0$, $\hat{B}(e) = 1$
for $p \in \{1, \dots, P\}$ **do in parallel**
  while $\exists$ *element to process* **do**
    e = next element to process
    $\Delta_+^{\max}(e) = F(\hat{A} \cup e) - F(\hat{A})$
    $\Delta_-^{\max}(e) = F(\hat{B} \backslash e) - F(\hat{B})$
    Draw $u_e \sim Unif(0,1)$
    if $u_e < \frac{[\Delta_+^{\max}(e)]_+}{[\Delta_+^{\min}(e)]_+ + [\Delta_-^{\max}(e)]_+}$ **then**
      $\hat{A}(e) \leftarrow 1$
    else
      $\hat{B}(e) \leftarrow 0$

Output: $A = \hat{A}$

The sequential bidirectional greedy [1] algorithm monotonically grows A^i and shrinks B^i.

Algorithm 3: Hogwild for separable sums

For each $e \in V$, $\hat{A}(e) = 0$
For each l, $\hat{\sigma}_l = 0$, $\hat{\tau}_l = \sum_{e \in S_l} w_l(e)$
for $p \in \{1, \ldots, P\}$ **do in parallel**
  while $\exists$ *element to process* **do**
    $e =$ next element to process
    $\Delta_+^{\max}(e) = -\lambda v(e)$
    $\Delta_-^{\max}(e) = +\lambda v(e)$
    for $l : e \in S_l$ **do**
      $\Delta_+^{\max}(e) \leftarrow \Delta_+^{\max}(e) + g(\hat{\sigma}_l + w_l(e)) - g(\hat{\sigma}_l)$
      $\Delta_-^{\max}(e) \leftarrow \Delta_-^{\max}(e) + g(\hat{\tau}_l - w_l(e)) - g(\hat{\tau}_l)$
    Draw $u_e \sim Unif(0, 1)$
    if $u_e < \frac{[\Delta_+^{\max}(e)]_+}{[\Delta_+^{\min}(e)]_+ + [\Delta_-^{\max}(e)]_+}$ **then**
      $\hat{A}(e) \leftarrow 1$
      for $l : e \in S_l$ **do**
        $\hat{\sigma}_l \leftarrow \hat{\sigma}_l + w_l(e)$
    else
      for $l : e \in S_l$ **do**
        $\hat{\tau}_l \leftarrow \hat{\tau}_l - w_l(e)$

Output: $A = \hat{A}$

Figure 1: The serial submodular maximization algorithm and parallel implementation.

3.2 Hogwild for arbitrary submodular F

Algorithm 2 is the hogwild parallel bidirectional greedy unconstrained submodular maximization algorithm. We associate with each element e a time T_e at which Algorithm 2 line 7 is executed, and order the elements according to the times T_e. Let $\iota(e)$ be the position of e in this ordering. This total ordering on elements also allows us to define sets A^i, B^i corresponding to that obtained by the serial algorithm; specifically, $A^i = \{e' : e' \in A, \iota(e') < i\}$ and $B^i = A^i \cup \{e' : \iota(e') \geq i\}$.

Note that in Algorithm 2, lines 5 and 6 may be executed in parallel with lines 9 and 11. Hence, $\Delta_+^{\max}(e)$ and $\Delta_-^{\max}(e)$ (lines 5 and 6) may be computed with different values of $\hat{A}(e')$. We denote by $\hat{A}_e$ and $\hat{B}_e$ respectively the vectors of $\hat{A}$ and $\hat{B}$ that are used in the computation of $\Delta_+^{\max}(e)$ and $\Delta_-^{\max}(e)$.

Lemma 3.1. *For any $e \in V$, $\hat{A}_e \subseteq A^{\iota(e)-1}$, $\hat{B}_e \supseteq B^{\iota(e)-1}$.*

Proof. Consider any element $e' \in V$. If $e' \in \hat{A}_e$, it must be the case that the algorithm set $\hat{A}(e')$ to 1 (line 9) before T_e, which implies $\iota(e') < \iota(e)$, and hence $e' \in A^{\iota(e)-1}$. So $\hat{A}_e \subseteq A^{\iota(e)-1}$.

Similarly, if $e' \notin \hat{B}_e$, then the algorithm set $\hat{B}(e')$ to 0 (line 11) before T_e, so $\iota(e') < \iota(e)$. Also, $e' \notin A$ because the execution of line 11 excludes the execution of line 9. Therefore, $e' \notin A^{\iota(e)-1}$, and $e' \notin B^{\iota(e)-1}$. So $\hat{B}_e \subseteq B^{\iota(e)-1}$. $\square$

It's easy to see that

$$\begin{aligned}
\Delta_+(e) &= F(A^{i-1} \cup i) - F(A^{i-1}) \\
\Delta_+^{\max}(e) &= F(\hat{A}_e \cup i) - F(\hat{A}_e) \\
\Delta_-(e) &= F(B^{i-1} \backslash i) - F(B^{i-1}) \\
\Delta_-^{\max}(e) &= F(\hat{B}_e \backslash i) - F(\hat{B}_e)
\end{aligned}$$

Corollary 3.2. *Submodularity of F implies that*

$$\begin{aligned}\Delta_+(e) &\leq \Delta_+^{\max}(e),\\ \Delta_-(e) &\leq \Delta_-^{\max}(e).\end{aligned}$$

3.3 Hogwild for separable sums F

For some functions F, we can maintain sketches / statistics to aid the computation of $\Delta_+^{\max}$, $\Delta_-^{\max}$, and obtain the bounds given in Corollary 3.2. In particular, we consider functions of the form

$$F(X) = \sum_{l=1}^{L} g\left(\sum_{i\in X\cup S_l} w_l(i)\right) - \lambda \sum_{i\in X} v(i),$$

where $S_l \subseteq V$ are (possibly overlapping) groups of elements in the ground set, g is a non-decreasing concave scalar function, and $w_l(i)$ and $v(i)$ are non-negative scalar weights. It is easy to see that

$$F(X\cup e) - F(X) = \sum_{l:e\in S_l}\left[g\left(w_l(e) + \sum_{i\in X\cup S_l} w_l(i)\right) - g\left(\sum_{i\in X\cup S_l} w_l(i)\right)\right] - \lambda v(e).$$

Define

$$\begin{aligned}\hat{\sigma}_l &= \sum_{j\in\hat{A}\cup S_l} w_l(j), & \hat{\sigma}_{l,e} &= \sum_{j\in\hat{A}_e\cup S_l} w_l(j), & \sigma_l^{\iota(e)-1} &= \sum_{j\in A^{\iota(e)-1}\cup S_l} w_l(j).\\ \hat{\tau}_l &= \sum_{j\in\hat{B}\cup S_l} w_l(j), & \hat{\tau}_{l,e} &= \sum_{j\in\hat{B}_e\cup S_l} w_l(j), & \tau_l^{\iota(e)-1} &= \sum_{j\in B^{\iota(e)-1}\cup S_l} w_l(j).\end{aligned}$$

We can update $\hat{\sigma}_l$ and $\hat{\tau}_l$ according to Algorithm 3. Following arguments analogous to that of Lemma 3.1, we can show the following:

Lemma 3.3. *For each l and $e \in V$, $\hat{\sigma}_{l,e} \leq \sigma_l^{\iota(e)-1}$ and $\hat{\tau}_{l,e} \geq \tau_l^{\iota(e)-1}$.*

Corollary 3.4. *Concavity of g implies*

$$\begin{aligned}\Delta_+^{\max}(e) &= \sum_{l:e\in S_l}\left[g(\hat{\sigma}_{l,e} + w_l(e)) - g(\hat{\sigma}_{l,e})\right] - \lambda v(e)\\ &\geq \sum_{l:e\in S_l}\left[g(\hat{\sigma}_l^{\iota(e)-1} + w_l(e)) - g(\hat{\sigma}_l^{\iota(e)-1})\right] - \lambda v(e)\\ &= \Delta_+(e),\\ \Delta_-^{\max}(e) &= \sum_{l:e\in S_l}\left[g(\hat{\tau}_{l,e} - w_l(e)) - g(\hat{\tau}_{l,e})\right] + \lambda v(e)\\ &\geq \sum_{l:e\in S_l}\left[g(\hat{\tau}_l^{\iota(e)-1} - w_l(e)) - g(\hat{\tau}_l^{\iota(e)-1})\right] + \lambda v(e)\\ &= \Delta_-(e),\end{aligned}$$

4 Approximation of hogwild bidirectional greedy

Theorem 4.1. *Let F be a non-negative (monotone or non-monotone) submodular function. The hogwild bidirectional greedy algorithm solves the unconstrained problem $\max_{A\subset V} F(A)$ with approximation*

$$E[F(A)] \geq \frac{1}{2}F^* - \frac{1}{4}\sum_{i=1}^{n} E[\rho_i],$$

where A is the output of the algorithm, F^* is the optimal value, and ρ_i is a random variable such that $\rho_i \geq \Delta_+^{\max}(i) - \Delta_+(i)$ and $\rho_i \geq \Delta_-^{\max}(i) - \Delta_-(i)$.

4.1 Assumption

F is submodular and non-negative.

We assume that we can bound

$$\Delta_+^{\max} - \rho_i \leq \Delta_+ \leq \Delta_+^{\max} \leq \Delta_+ + \rho_i$$
$$\Delta_-^{\max} - \rho_i \leq \Delta_- \leq \Delta_-^{\max} \leq \Delta_- + \rho_i$$

This is possible, for example, by defining

$$\begin{aligned}
\rho_i &= \max_{S,T \subseteq V} \{[F(S \cup i) - F(S)] - [F(S \cup T \cup i) - F(S \cup T)]\} \\
&\leq F(i) - F(\emptyset) - F(V) + F(V \backslash i) \\
&\leq F(i) \left(1 - \frac{F(V) - F(V \backslash i)}{F(i)}\right) \\
&= F(i)\kappa_F
\end{aligned}$$

where S plays the role of A^j and T plays the role of $\{j+1, \ldots, i-1\}$, and κ_F is the total curvature of F. Summing over i then gives us $\sum_i \rho_i \leq \kappa_F \sum_i F(i)$.

[XP: Is there theory along these lines?]

Alternatively, we could assume that $|A^{\iota(e)-1} \backslash \hat{A}_e| \leq \xi$, which would be the case if the lag between processors is bounded. Letting $T = A^{\iota(e)-1} \backslash \hat{A}_e$, we could define

$$\rho_i = F(i) - F(\emptyset) - F(T \cup i) + F(T).$$

[XP: Need to check above statement.]

4.2 Proof

We follow the proof outline of [1].

Let OPT be an optimal solution to the problem. Define $O^i := (OPT \cup A^i) \cap B^i$. Note that O^i coincides with A^i and B^i on elements $1, \ldots, i$, and O^i coincides with OPT on elements $i+1, \ldots, n$. Hence,

$$O^i \backslash i + 1 \supseteq A^i$$
$$O^i \cup i + 1 \subseteq B^i.$$

Lemma 4.2. *For every* $1 \leq i \leq n$, $\Delta_+(i) + \Delta_-(i) \geq 0$.

Proof. This is just Lemma II.1 of [1]. □

Lemma 4.3. *For every* $1 \leq i \leq n$,

$$E[F(O^{i-1}) - F(O^i)] \leq \frac{1}{2} E[f(A^i) - f(A^{i-1}) + f(B^i) - f(B^{i-1}) + \rho_i].$$

Proof. We follow the proof outline of [1]. First, note that it suffices to prove the inequality conditioned on knowing A^{i-1} and j, then applying the law of total expectation. Under this conditioning, we also know B^{i-1}, O^{i-1}, $\Delta_+(i)$, $\Delta_+^{\max}(i)$, $\Delta_-(i)$, $\Delta_-^{\max}(i)$, and ρ_i.

We consider the following 9 cases.

Case 1: $\Delta_+(i) \leq \Delta_+^{\max}(i) \leq 0$, $\Delta_-(i) \leq \Delta_-^{\max}(i) \leq 0$. This is not possible, by Lemma 4.2.

Case 2: $\Delta_+(i) \leq \Delta_+^{\max}(i) \leq 0$, $\Delta_-(i) \leq 0 < \Delta_-^{\max}(i)$. This is not possible, by Lemma 4.2.

Case 3: $\Delta_+(i) \le \Delta_+^{\max}(i) \le 0$, $0 < \Delta_-(i) \le \Delta_-^{\max}(i)$. In this case, the algorithm always choses to exclude i, so $A^i = A^{i-1}$, $B^i = B^{i-1}\backslash i$ and $O^i = O^{i-1}\backslash i$:

$$\begin{aligned}
E[F(A^i) - F(A^{i-1})|A^{i-1}, j] &= F(A^{i-1}) - F(A^{i-1}) = 0 \\
E[F(B^i) - F(B^{i-1})|A^{i-1}, j] &= F(B^{i-1}\backslash) - F(B^{i-1}) = \Delta_-(i) > 0 \\
E[F(O^{i-1}) - F(O^i)|A^{i-1}, j] &= F(O^{i-1}) - F(O^{i-1}\backslash i) \\
&\le \begin{cases} F(A^{i-1} \cup i) - F(A^{i-1}) & \text{if } i \in OPT \\ 0 & \text{if } i \notin OPT \end{cases} \\
&= \begin{cases} \Delta_+(i) & \text{if } i \in OPT \\ 0 & \text{if } i \notin OPT \end{cases} \\
&\le 0 \\
&< \frac{1}{2} E[f(A^i) - f(A^{i-1}) + f(B^i) - f(B^{i-1}) + \rho_i | A^{i-1}, j]
\end{aligned}$$

where the first inequality is due to submodularity: $O^{i-1}\backslash i \supseteq A^{i-1}$.

Case 4: $\Delta_+(i) \le 0 < \Delta_+^{\max}(i)$, $\Delta_-(i) \le \Delta_-^{\max}(i) \le 0$. This is not possible, by Lemma 4.2.

Case 5: $\Delta_+(i) \le 0 < \Delta_+^{\max}(i)$, $\Delta_-(i) \le 0 < \Delta_-^{\max}(i)$. This is not possible, by Lemma 4.2.

Case 6: $\Delta_+(i) \le 0 < \Delta_+^{\max}(i)$, $0 < \Delta_-(i) < \Delta_-^{\max}(i)$. Since both $\Delta_+^{\max}(i) > 0$ and $\Delta_-^{\max}(i) > 0$, the probability of including i is just $\Delta_+^{\max}(i)/(\Delta_+^{\max}(i) + \Delta_-^{\max}(i))$, and the probability of excluding i is $\Delta_-^{\max}(i)/(\Delta_+^{\max}(i) + \Delta_-^{\max}(i))$.

$$\begin{aligned}
E[F(A^i) - F(A^{i-1})|A^{i-1}, j] &= \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} (F(A^{i-1} \cup i) - F(A^{i-1})) \\
&= \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} \Delta_+(i) \\
&\ge \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} (\Delta_+^{\max}(i) - \rho_i) \\
E[F(B^i) - F(B^{i-1})|A^{i-1}, j] &= \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} (F(B^{i-1}\backslash i) - F(B^{i-1})) \\
&= \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} \Delta_-(i) \\
&\ge \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} (\Delta_-^{\max}(i) - \rho_i)
\end{aligned}$$

$$
\begin{aligned}
&E[F(O^{i-1}) - F(O^i)|A^{i-1}, j] \\
=&\frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)}(F(O^{i-1}) - F(O^{i-1} \cup i)) \\
&+ \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)}(F(O^{i-1}) - F(O^{i-1} \backslash i)) \\
=&\begin{cases} \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}(F(O^{i-1}) - F(O^{i-1} \cup i)) & \text{if } i \notin OPT \\ \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}(F(O^{i-1}) - F(O^{i-1} \backslash i)) & \text{if } i \in OPT \end{cases} \\
\leq&\begin{cases} \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}(F(B^{i-1} \backslash i) - F(B^{i-1})) & \text{if } i \notin OPT \\ \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}(F(A^{i-1} \cup i) - F(A^{i-1})) & \text{if } i \in OPT \end{cases} \\
=&\begin{cases} \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}\Delta_-(i) & \text{if } i \notin OPT \\ \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}\Delta_+(i) & \text{if } i \in OPT \end{cases} \\
\leq&\begin{cases} \frac{\Delta_+^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}\Delta_-^{\max}(i) & \text{if } i \notin OPT \\ \frac{\Delta_-^{\max}(i)}{\Delta_+^{\max}(i)+\Delta_-^{\max}(i)}\Delta_+^{\max}(i) & \text{if } i \in OPT \end{cases} \\
=&\frac{\Delta_+^{\max}(i)\Delta_-^{\max}(i)}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)}
\end{aligned}
$$

where the first inequality is due to submodularity: $O^{i-1} \backslash i \supseteq A^{i-1}$ and $O^{i-1} \cup i \subseteq B^{i-1}$.

Putting the above inequalities together:

$$
\begin{aligned}
&E[F(O^{i-1}) - F(O^i)|A^{i-1}, j] - \frac{1}{2}E[f(A^i) - f(A^{i-1}) + f(B^i) - f(B^{i-1}) + \rho_i|A^{i-1}, j] \\
\leq& \frac{1/2}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)}\Big[2\Delta_+^{\max}(i)\Delta_-^{\max}(i) - \Delta_-^{\max}(i)(\Delta_-^{\max}(i) - \rho_i) \\
&\qquad - \Delta_+^{\max}(i)(\Delta_+^{\max}(i) - \rho_i)\Big] - \frac{1}{2}\rho_i \\
=& \frac{1/2}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)}\Big[- (\Delta_+^{\max}(i) - \Delta_-^{\max}(i))^2 + \rho_i(\Delta_+^{\max}(i) + \Delta_-^{\max}(i))\Big] - \frac{1}{2}\rho_i \\
\leq& \frac{\frac{1}{2}\rho_i(\Delta_+^{\max}(i) + \Delta_-^{\max}(i))}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} - \frac{1}{2}\rho_i \\
=& 0.
\end{aligned}
$$

Case 7: $0 < \Delta_+(i) \leq \Delta_+^{\max}(i)$, $\Delta_-(i) \leq \Delta_-^{\max}(i) \leq 0$. Analogous to Case 3.

Case 8: $0 < \Delta_+(i) \leq \Delta_+^{\max}(i)$, $\Delta_-(i) \leq 0 < \Delta_-^{\max}(i)$. Analogous to Case 6.

Case 8: $0 < \Delta_+(i) \leq \Delta_+^{\max}(i)$, $0 < \Delta_-(i) \leq \Delta_-^{\max}(i)$. In the proof of Case 6, we only required that $\Delta_+^{\max}(i) > 0$ and $\Delta_-^{\max}(i) > 0$, but did not use the fact that $\Delta_+^{\max}(i) \leq 0$. Thus the proof of Case 6 also holds for Case 8.

□

([XP: Note] If we weaken the assumption of $\Delta_+(i) \leq \Delta_+^{\max}(i)$ to $\Delta_+(i) \leq \Delta_+^{\max}(i) + \epsilon_i$, then in Case 6 above, we can instead bound

$$
\begin{aligned}
E[F(O^{i-1}) - F(O^i)|A^{i-1}, j] &\leq \frac{\Delta_+^{\max}(i)\Delta_-^{\max}(i) + \epsilon \max(\Delta_+^{\max}(i), \Delta_-^{\max})}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)} \\
&\leq \frac{\Delta_+^{\max}(i)\Delta_-^{\max}(i) + \epsilon(\Delta_+^{\max}(i) + \Delta_-^{\max})}{\Delta_+^{\max}(i) + \Delta_-^{\max}(i)}.
\end{aligned}
$$

The bound of Lemma 4.3 becomes

$$E[F(O^{i-1}) - F(O^i)] \leq \frac{1}{2}E[f(A^i) - f(A^{i-1}) + f(B^i) - f(B^{i-1}) + \rho_i + 2\epsilon_i],$$

and the bound of Theorem 4.1 becomes $E[F(A)] \geq \frac{1}{2}F^* - \frac{1}{4}\sum_i E[\rho_i + 2\epsilon_i]$.)

We will now prove Theorem 4.1.

Theorem 4.1. Summing up the statement of Lemma 4.3 for all i gives us a telescoping sum, which reduces to:

$$\begin{aligned} E[F(O^0) - F(O^n)] &\leq \frac{1}{2}E[F(A^n) - F(A^0) + F(B^n) - F(B^0)] + \frac{1}{2}\sum_{i=1}^{n} E[\rho_i] \\ &\leq \frac{1}{2}E[F(A^n) + F(B^n)] + \frac{1}{2}\sum_{i=1}^{n} E[\rho_i]. \end{aligned}$$

Note that $O^0 = OPT$ and $O^n = A^n = B^n$, so $E[F(A^n)] \geq \frac{1}{2}F(OPT) - \frac{1}{4}\sum_i E[\rho_i]$. □

4.3 Example: max graph cut

Let $C^{ji} = \{j+1, \ldots, i-1\}$, $D^i = \{i+1, \ldots, n\}$. Denote $\tilde{A}^j = V \backslash A^j \backslash C^{ji} \backslash D^i = \{1, \ldots, j\} \backslash A^j$ be the elements up to j that are not included in A. Let $w_i(S) = \sum_{j \in S, (i,j) \in E} w(i,j)$. For the max graph cut function, it is easy to see that

$$\begin{aligned} \Delta_+ &\geq -w_i(A^j) - w_i(C^{ji}) + w_i(D^i) + w_i(\tilde{A}^j) \\ \Delta_+^{\max} &= -w_i(A^j) + w_i(C^{ji}) + w_i(D^i) + w_i(\tilde{A}^j) \\ \Delta_- &\geq +w_i(A^j) - w_i(C^{ji}) + w_i(D^i) - w_i(\tilde{A}^j) \\ \Delta_-^{\max} &= +w_i(A^j) + w_i(C^{ji}) + w_i(D^i) - w_i(\tilde{A}^j) \end{aligned}$$

Thus, we can set $\rho_i = 2w_i(C^{ji})$.

Suppose we have P processors, so $|C^{ji}| = (1+\alpha)P$ for some small $\alpha \geq 0$. Then $w_i(C^{ji})$ has a hypergeometric distribution with mean $\frac{\deg(i)}{n}(1+\alpha)P$, and $E[\rho_i] = 2(1+\alpha)P\frac{\deg(i)}{n}$. The approximation of the hogwild algorithm is then $E[F(A^n)] \geq \frac{1}{2}F(OPT) - \frac{1}{4}(1+\alpha)P\sum_i \frac{\deg(i)}{n} = \frac{1}{2}F(OPT) - (1+\alpha)P\frac{\#\text{edges}}{2n}$. In sparse graphs, the hogwild algorithm is off by a small additional term, which albeit grows linearly in P.

5 Additional comments

I was unable to work through the analysis for the algorithm that includes i with probability $\Delta_+^{\min}/(\Delta_+^{\min} + \Delta_-^{\max})$, excludes i with probability $\Delta_+^{\min}/(\Delta_+^{\min} + \Delta_-^{\max})$, and otherwise takes a random action. Although this is closer to the spirit of the OCC algorithm, it seems that the random action can break the bounds quite badly. Intuitively, the probability of taking the wrong action could be higher than that presented in the hogwild analysis above.

In any case, the ability to run hogwild as-above allows us to do without computing the lower bounds $\Delta_+^{\min}$, $\Delta_-^{\min}$, which can be significantly because we only need to maintain sketches of A^j and B^j on-the-fly.

We are also able to be more 'order-agnostic', i.e. do away with the need of a pre-determined global ordering of elements. Each processor can complete its computations based only on A^j, without knowing C^{ji}. Hence, processors can read a snapshot of the current global state, perform their computations, and update their indicator variables independently.

However, it might take more effort to extend the hogwild approach to a constrained setting. The concurrency control approach has, ultimately, a serialization ordering, which can ensure that we're always feasible at any point of time.

6 OCC

6.1 F given by value oracle

The serialization order is given by $\iota(e)$, which is the value of ι at line 11 of Algorithm 4.

Lemma 6.1. *$\hat{A}_e \subseteq A^{\iota(e)-1} \subseteq \tilde{A}_e$, and $\hat{B}_e \supseteq B^{\iota(e)-1} \supseteq \tilde{B}_e$.*

Proof. There are 4 parts to this proof.

1. $e' \in \hat{A}_e \implies e' \in A^{\iota(e)-1}$.
2. $e' \in A^{\iota(e)-1} \implies e' \in \tilde{A}_e$.
3. $e' \notin \hat{B}_e \implies e' \notin B^{\iota(e)-1}$.
4. $e' \notin B^{\iota(e)-1} \implies e' \notin \tilde{B}_e$.

□

Corollary 6.2. *By submodularity of F, $\Delta_+^{\min}(e) \leq \Delta_+(e) \leq \Delta_+^{\max}(e)$, and $\Delta_-^{\min}(e) \leq \Delta_-(e) \leq \Delta_-^{\max}(e)$.*

Theorem 6.3. *OCC bidirectional greedy is serially equivalent to bidirectional greedy.*

Proof. Outline of proof: We need to show 2 things. Firstly, that the sampling using Δ_+^{min}, Δ_+^{max}, Δ_-^{min}, Δ_-^{max} is 'safe', i.e. is equivalently to sampling using Δ_+ and Δ_-. Secondly, that the validation process is correct – specifically that when the validation is executed, it is in fact the case that $\hat{A} = A^{\iota(e)-1}$ and $\hat{B} = B^{\iota(e)-1}$. □

6.2 Separable sums F

We maintain $\tilde{\sigma}_l$, $\hat{\sigma}_l$, $\tilde{\tau}_l$, $\hat{\tau}_l$.

It can be shown that $\hat{\sigma}_{l,e} \leq \sigma^{\iota(e)-1} \leq \tilde{\sigma}_{l,e} - w_l(e)$ and $\hat{\sigma}_{l,e} \geq \tau^{\iota(e)-1} \geq \tilde{\tau}_{l,e} + w_l(e)$, which then allows us to compute our bounds for Δ's.

References

[1] Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. In *Proceedings of the 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, FOCS '12, pages 649–658, Washington, DC, USA, 2012. IEEE Computer Society. ISBN 978-0-7695-4874-6. doi: 10.1109/FOCS.2012.73. URL `http://dx.doi.org/10.1109/FOCS.2012.73`.

Algorithm 4: OCC bidirectional greedy

For each $e \in V$, set $\hat{A}(e) = \tilde{A}(e) = 0$, $\hat{B}(e) = \tilde{B}(e) = 1$
For each $i = 1, \dots, |V|$, set result$(i) = 0$, processed$(i) = false$
For each $p = 1, \dots, P$, set maxResult$(p) = 0$, maxProcessed$(p) = 0$
$\iota = 0$
for $p \in \{1, \dots, P\}$ **do in parallel**
  while $\exists$ *element to process* **do**
    $e =$ next element to process
    `// Aggressively grow` $\tilde{A}$ `and shrink` $\tilde{B}$.
    $\tilde{A}(e) \leftarrow 1$
    $\tilde{B}(e) \leftarrow 0$
    $i = \iota$; $\iota \leftarrow \iota + 1$ `// Get serialization order.`
    $\Delta_+^{\min}(e) = F(\tilde{A} \cup e) - F(\tilde{A})$
    $\Delta_+^{\max}(e) = F(\hat{A} \cup e) - F(\hat{A})$
    $\Delta_-^{\min}(e) = F(\tilde{B} \backslash e) - F(\tilde{B})$
    $\Delta_-^{\max}(e) = F(\hat{B} \backslash e) - F(\hat{B})$
    Draw $u_e \sim Unif(0, 1)$
    `// Try to determine result for` e.
    if $u_e < \frac{[\Delta_+^{\min}(e)]_+}{[\Delta_+^{\min}(e)]_+ + [\Delta_-^{\max}(e)]_+}$ **then**
      result$(i) \leftarrow 1$
    else if $u_e > \frac{[\Delta_+^{\max}(e)]_+}{[\Delta_+^{\max}(e)]_+ + [\Delta_-^{\min}(e)]_+}$ **then**
      result$(i) \leftarrow -1$
    `// Ensure no earlier ordered element needs validation.`
    while *maxResult*$(p) < i$ **do**
      if *result(maxResult(p))* $\neq 0$ **then**
        maxResult$(p) \leftarrow$ maxResult$(p) + 1$
    if *result*$(i) = 0$ **then**
      validate(p, e, i) `// Validate if necessary.`
    `// Can process` e `now.`
    if *result*$(i) = 1$ **then**
      $\hat{A}(e) \leftarrow 1$ `// Unroll wrong update to` $\tilde{B}$.
      $\tilde{B}(e) \leftarrow 1$
    else
      $\tilde{A}(e) \leftarrow 0$ `// Unroll wrong update to` $\tilde{A}$.
      $\hat{B}(e) \leftarrow 0$
    processed$(i) = true$

Algorithm 5: validate(p, e, i)

`// Wait for all earlier results to be processed.`
while *maxProcessed*$(p) < i$ **do**
  if *processed(maxProcessed(p))* **then**
    maxProcessed$(p) \leftarrow$ maxProcessed$(p) + 1$
`// Can assign` e `now:` $A^{i-1} = \hat{A}$ `and` $B^{i-1} = \hat{B}$.
$\Delta_+(e) = F(\hat{A} \cup e) - F(\hat{A})$
$\Delta_-(e) = F(\hat{B} \backslash e) - F(\hat{B})$
if $u_e < \frac{[\Delta_+(e)]_+}{[\Delta_+(e)]_+ + [\Delta_-(e)]_+}$ **then**
  result$(i) \leftarrow 1$
else
  result$(i) \leftarrow -1$