

Оптимальное разделение данных в распределенной оптимизации для машинного обучения

Глеб Молодцов

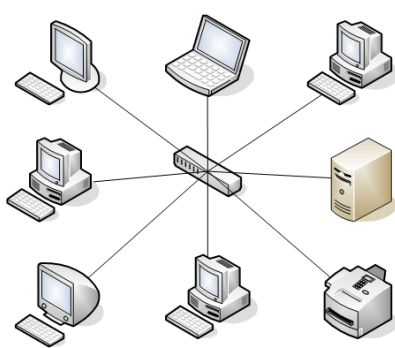
Московский физико-технический институт

Даниил Медяков

Аннотация

Рассматривается распределение данных между различными устройствами с учетом их мощностей. Сеть представляет собой звездную топологию.

Целью данной работы является получение оптимального соотношения для распределения данных между сервером и локальными устройствами.



Вступление

Сеть состоит из агентов, образующих звёздную топологию. Первый является главным узлом, а остальные связаны только с ним и нужны для дополнительной мощности. Вычисления производятся с помощью сторонних узлов, а копии всех оптимизационных переменных и связь между узлами осуществляется через сервер. Задача сводится к минимизации следующей функции:

$$r(x) = \frac{1}{n} \sum_{i=1}^n f_i(x)$$

Полученную задачу можно решить, применяя алгоритм градиентного спуска для L -гладкой и μ -сильно выпуклой функции r . Для некоторых классов задач, где $k = L/\mu$ не является большим числом, этот метод приемлем. Однако при очень больших k алгоритм неэффективен. Это происходит, например, при высоких коммуникационных затратах. Представим функцию $g(x)$ в виде:

$$r(x) = \underbrace{f_1(x)}_{:=q(x)} + \underbrace{\frac{1}{n} \sum_{i=1}^n [f_i(x) - f_1(x)]}_{:=p(x)}$$

Для этой задачи была предложена модификация градиентного скользящего алгоритма из [1]. Все существующие алгоритмы градиентного скольжения были неприменимы к поставленной задаче, поскольку требовали выпуклости обеих функций и не обеспечивали оптимальных нижних оценок сложности как для локального вычисления градиента, так и для коммуникационных затрат. Однако данная модификация алгоритма восполняет пробелы. Приведём её ниже:

Algorithm 1 Accelerated Extragradient

```
1: Input:  $x^0 = x_f^0 \in \mathbb{R}^d$ 
2: Parameters:  $\tau \in (0, 1)$ ,  $\eta, \theta, \alpha > 0$ ,  $K \in \{1, 2, \dots\}$ 
3: for  $k = 0, 1, 2, \dots, K-1$  do
4:    $x_g^k = \tau x^k + (1-\tau)x_f^k$ 
5:    $x_f^{k+1} \approx \arg \min_{x \in \mathbb{R}^d} [A_g^k(x) := p(x_g^k) + \langle \nabla p(x_g^k), x - x_g^k \rangle + \frac{1}{2\theta} \|x - x_g^k\|^2 + q(x)]$ 
6:    $x^{k+1} = x^k + \eta \alpha (x_f^{k+1} - x^k) - \eta \nabla r(x_f^{k+1})$ 
7: end for
8: Output:  $x^K$ 
```

Постановка проблемы

Работаем с алгоритмом 1. Найдём количество операций алгоритма за одну итерацию. В строке 5 происходит одна коммуникация продолжительностью τ_{comm} , одно локальное вычисление длительностью $\max(\tau_1, \tau_2, \dots, \tau_n)$, одно вычисление в центральном узле длиной τ_1 и дополнительное вычисление в центральном узле. В строке 6 – одна коммуникация, одно локальное вычисление и одно вычисление в центральном узле. Пусть общее число итераций – K . Запишем общее время работы алгоритма в виде:

$$T_{sum} = 2 \cdot \max(\tau_1, \tau_2, \dots, \tau_n) \cdot K + 2 \cdot K \cdot \tau_{comm} + \tau_1 \cdot k_{some}$$

Итоговая задача – минимизировать данную функцию. Обозначим за b_i – объём данных на i -ом устройстве, τ_i^{loc} – время на обработку единицы данных на i -ом устройстве. Учитывая зависимость $\tau_i = b_i \cdot \tau_i^{loc}$, а также, принимая во внимание оценки, приведенные в [1], получаем следующую оптимизационную задачу:

$$\min_{\sum_{i=1}^n b_i = N, \delta = \frac{L}{\mu}} [(\max(\tau_1^{loc} \cdot b_1, \tau_2^{loc} \cdot b_2, \dots, \tau_n^{loc} \cdot b_n) + \tau_{comm}) * \mathcal{O}(\max\{1, \sqrt{\frac{L_p}{\mu}} \log(\frac{1}{\varepsilon})\}) + \mathcal{O}(\max\{1, \sqrt{\frac{L_q}{L_p}} \cdot \sqrt{\frac{L_p}{\mu}} \cdot \sqrt{\frac{L_q}{\mu}} \log(\frac{1}{\varepsilon})\})]$$

В работе рассматриваются два случая: (1) : $\delta = \frac{L}{b_1}$; (2) : $\delta = \frac{L}{\sqrt{b_1}}$.

Для случая (1) получается следующая задача минимизации:

$$\min_{\sum_{i=1}^n b_i = N} [(\max\{\tau_1^{loc} \cdot b_1, \tau_2^{loc} \cdot b_2\} + \tau_{comm}) \cdot \mathcal{O}(\sqrt{\frac{L}{\mu b_1}} \log(\frac{1}{\varepsilon})) + \tau_1^{loc} \cdot b_1 \cdot \mathcal{O}(\sqrt{\frac{L}{\mu}} \log(\frac{1}{\varepsilon}))]$$

Для (2) аналогично получаем:

$$\min_{\sum_{i=1}^n b_i = N} [(\max\{\tau_1^{loc} \cdot b_1, \tau_2^{loc} \cdot b_2\} + \tau_{comm}) \cdot \mathcal{O}(\sqrt{\frac{L}{\mu \sqrt{b_1}}} \log(\frac{1}{\varepsilon})) + \tau_1^{loc} \cdot b_1 \cdot \mathcal{O}(\sqrt{\frac{L}{\mu}} \log(\frac{1}{\varepsilon}))]$$

Методология решения

- В случае (1) получение аналитической формулы сводится к поиску корней уравнения третьей степени следующего вида: $y^3 + \beta \cdot y + \gamma = 0$. Решение такого уравнения ищется с помощью формулы Кардано, причём в нашем случае получается лишь единственный вещественный корень, который и будет искомым.

- В (2) не удаётся найти аналитическое решение в общем случае, то есть при любом времени коммуникаций, так как в ходе поиска возникает уравнение высоких степеней. Поэтому для данной задачи решение искалось в двух предельных ситуациях:

$$2.1. \forall i \hookrightarrow \tau_{comm} \ll \tau_i^{loc}, \forall i \neq j \hookrightarrow \tau_i^{loc} \neq \tau_j^{loc}$$

$$2.2. \forall i \hookrightarrow \tau_{comm} \gg \tau_i^{loc}, \forall i \neq j \hookrightarrow \tau_i^{loc} = \tau_j^{loc}$$

Результаты

Рассмотрим отдельно случаи (1) и (2). Результатом в обоих будет количество данных на сервере, причём данные между остальными устройствами распределяются равномерно относительно их мощностей.

Далее для обоих случаев будем рассматривать две ситуации:

$$(a) : 0 < b_1 \leq b_1^0, \quad (b) : b_1^0 < b_1 \leq N$$
$$b_1^0 = \frac{N(\sum_{i=2}^n \frac{1}{\tau_i^{loc}})^{-1}}{\tau_1^{loc} + (\sum_{i=2}^n \frac{1}{\tau_i^{loc}})^{-1}}$$

С учётом этого, получим решения для (1), (2).

- Точное решение в случае (1):

$$b_1 = \frac{a^2}{3c^2} + \frac{\sqrt[3]{2a^6 + 3\sqrt{3}\sqrt{4a^3b^3c^6 + 27b^4c^8 + 18a^3bc^2 + 27b^2c^4}}}{3\sqrt[3]{2c^2}} - \frac{\sqrt[3]{2}(-a^4 - 6abc^2)}{3c^2\sqrt[3]{2a^6 + 3\sqrt{3}\sqrt{4a^3b^3c^6 + 27b^4c^8 + 18a^3bc^2 + 27b^2c^4}}}$$

$$\begin{cases} (a) : a = \frac{1}{2}c_1\sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon})(\sum_{i=2}^n \frac{1}{\tau_i^{loc}})^{-1}; c = \tau_1^{loc} \cdot c_2\sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon}); \\ b = -\frac{1}{2}c_1[N(\sum_{i=2}^n \frac{1}{\tau_i^{loc}})^{-1} + \tau_{comm}] \cdot \sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon}); \\ (b) : a = \frac{1}{2}c_1\sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon})\tau_1^{loc}; c = \tau_1^{loc} \cdot c_2\sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon}); \\ b = -\frac{1}{2}c_1\tau_{comm} \cdot \sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon}); \end{cases}$$

- Случай (2)

2.1. Малые коммуникации

$$b_1 = b_1^0$$

2.2. Большие коммуникации

$$b_1 = (\frac{\alpha\tau_{comm}}{4\beta\tau})^{\frac{4}{5}}$$

$$\alpha = c_1 \cdot \sqrt{\frac{L}{\mu}} \cdot \log(\frac{1}{\varepsilon}), \beta = c_2 \cdot \sqrt{\frac{L}{\mu}} \cdot \log(\frac{1}{\varepsilon})$$

Заметим, что задача свелась к решению уравнения относительно b_1 . Мы не всегда можем найти аналитическое решение, например, как в случае (2). Однако всегда можно решить задачу численно, поскольку функция на рассматриваемых промежутках является монотонной и непрерывной.

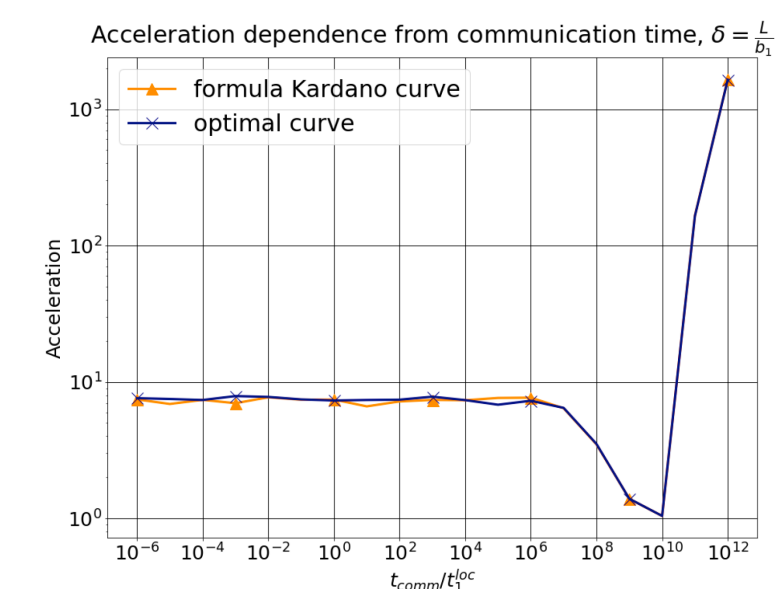
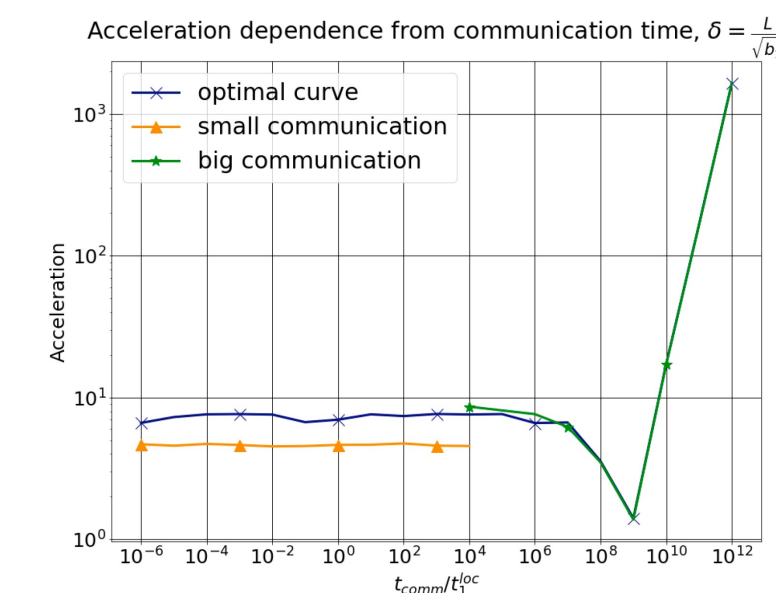
Эксперимент

Для экспериментов рассмотрим задачу "Ridge Regression":

$$\min_{\omega} [\frac{1}{2N} X\omega - y^2 + \frac{\lambda}{2}\omega^2], q(\omega) = \frac{1}{2N} X\omega - y^2, p(\omega) = \frac{\lambda}{2}\omega^2$$

Была проведена симуляция работы Алгоритма 1 на датасете размером 100'000 строк. Для поиска $argmin$ был использован итеративный метод OGM-C из [2]. После экспериментального расчёта количества итераций для достижения определённой точности было проведено сравнение времени работы на равномерном распределении данных с временем на полученном выше распределении.

Итоговый результат представлен ниже.



Формула для случая больших коммуникаций и формула Кардано практически совпали с поиском оптимального решения. Для случая малых коммуникаций большая разница объясняется тем, что формула была получена в грубом приближении. Однако, если учесть порядок констант α, β , то можно получить результат, приближённый к оптимальному:

$$b_1 \cong \frac{4 \cdot 10^3 \tau_1^{loc}}{N \left(\sum_{i=2}^n (\tau_i^{loc})^{-1} \right)^{-1}}$$

Источники

[1] Dmitry Kovalev, Aleksandr Beznosikov, Ekaterina Borodich, Alexander Gasnikov, and Gesualdo Scutari. Optimal gradient sliding and its application to optimal distributed optimization under similarity. *Advances in Neural Information Processing Systems*, 35:33494–33507, 2022.

[2] Donghwan Kim and Jeffrey A Fessler. Optimizing the efficiency of first-order methods for decreasing the gradient of smooth convex functions. *Journal of optimization theory and applications*, 188(1):192–219, 2021.