Core-Sparse Monge Matrix Multiplication

Пресняков Сергей

2 мая 2025 г.

1 Мотивация

В данной статье речь пойдет про быстрое тропическое умножение монжевых матриц. Сперва скажу пару слов о том что такое тропическое произведение матриц и зачем нам оно нужно.

Определение 1. *Пусть $A \in M_{p\times r}, B \in M_{r\times q}$ — матрицы. Опеределим* тропическое умножение $\otimes$

$$A \otimes B[i,j] := \min_{k\in[0,r)} (A[i,k] + B[k,j])$$

Зачем же нужно так умножать? Давайте рассмотрим два графа G' и G'':

- В графе G' выделим p вершин-источников и r вершин-стоков.
- В графе G'' выделим r вершин-источников и q вершин-стоков.
- Рассмотрим матрицу A кратчайших путей между выделенными вершинами в графе G'.
- Рассмотрим матрицу B кратчайших путей между выделенными вершинами в графе G''.

Тогда мы можем склеить эти графы и посчитать матрицу кратчайших путей в полученном графе с помощью тропического умножения матриц. Это видно по определению

$$C = A \otimes B[i,j] = \min_{k\in[0,r)} (A[i,k] + B[k,j])$$

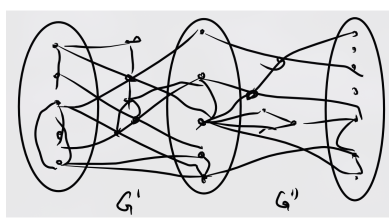

Рис. 1: Склеивание графов

2 Монжевы матрицы

А если граф планарный, то какое ограничение на матрицу кратчайших путей можно наложить? Такие матрицы называются *матрицами Монжа.*

Определение 2. *Матрица называется* матрицей Монжа*, если*

$$A[i,j] + A[i+1,j+1] \leqslant A[i,j+1] + A[i+1,j]$$

Замечание 1. *Монжевость равносильна следующему условию*

$$A[a,c] + A[b,d] \leqslant A[a,d] + A[b,c]$$

для $0 \leqslant a < b < p$, $0 \leqslant c < d < q$

Это условие возникает из планарности графа, а именно из-за «неравенства треугольника»

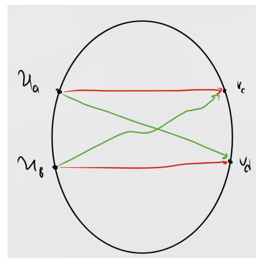

Рис. 2: Иллюстрация монжевости в планарных графах

Факт 1. *Монжевы матрицы замкнуты относительно тропического произведения. То есть*

$$A, B - \text{монжевы} \implies C := A \otimes B - \text{монжева матрица}$$

Определение 3. *Также нам нужны дополнительные определения для понимания результата этой статьи.*

- Матрица плотностей $A^{\square}$ *для монжевой матрицы* A.

$$A^{\square}[i,j] = A[i,j+1] + A[i+1,j] - A[i,j] - A[i+1,j+1] \quad i \in [0, p-1),\ j \in [0, q-1)$$

- Ядро A

$$core(A) := \{(i, j, A^{\square}[i,j]) | A^{\square}[i,j] \neq 0\}$$

- Размер ядра A

$$\delta(A) := |core(A)|$$

- Сумма ядра A

$$\delta^{\Sigma}(A) := \sum_{A^{\square}[i,j] \neq 0} A^{\square}[i,j]$$

- $A[a..b][c..d]$ *— смежная подматрица по индексам* $[a..b][c..d]$

- Матрица свидетелей $\mathcal{W}^{A,B}$

$$\mathcal{W}^{A,B}_{i,j} := \min \arg\min_j (A[i,j] + B[j,k])$$

- Сжатое представление матрицы A (CR A) *— это структура данных, которая содержит три объекта: самый левый столбец матрицы* A*, самая верхняя строка матрицы* A *и* $core(A)$

Основной результат данной статьи такой:

Теорема 1. *Существует алгоритм, который по сжатым представлениям матриц* $A \in M_{p\times q}$, $B \in M_{q\times r}$ *позволяет получить сжатое представление матрицы* $C := A \otimes B$ *за* $O(p + q + (r + \delta(A) + \delta(B)) \cdot \log(2 + \delta(A) + \delta(B)))$

3 Соотношения на δ и δ^Σ

Утверждение 1. *Верны следующие соотношения.*

- *Для любой целочисленной матрицы A: $\delta(A) \leqslant \delta^\Sigma(A)$.*
- $A_{a,c} + A_{b,d} + \delta^\Sigma([a..b][c..d]) = A_{a,d} + A_{b,c}$.
- *A' — подматрица A монжевой матрицы (не обязательно смежная), тогда A' также монжева матрица.*
- *A, B — монжевы матрицы, тогда $\mathcal{W}^{A,B}$ — не убывает по столбцам и по строкам.*

Лемма 1. *Для монжевых матриц A, B*

$$\delta^\Sigma(A \otimes B) \leqslant \min(\delta^\Sigma(A), \delta^\Sigma(B))$$

$A \in Monge_{p\times q},\ B \in Monge_{q\times r}$. $C := A \otimes B$. Пусть $i \in [0, p-1),\ k \in [0, r-1):\ C^\square_{i,k} \neq 0$. Тогда

$$\exists j_A, j_B \in [\mathcal{W}^{A,B}_{i,k}..\mathcal{W}^{A,B}_{i+1,k+1}):\ A^\square_{i,j_A} \neq 0,\ B^\square_{j_B,k} \neq 0.$$

То есть любому элементу ядра C соответствуют элемент ядра A и элемент ядра B, что дает возможность доказать неравенство на размер ядра C.

Теорема 2.

$$\delta(A \otimes B) \leqslant 2(\delta(A) + \delta(B))$$

Эта теорема является ключевой для доказательства сложности алгоритма, который будет описан ниже.

4 Непосредственно алгоритм.

Определение 4.

- *$core_{i,\cdot} = \{(i, j, A^\square_{i,j}) | A^\square_{i,j} \neq 0\}$, то есть сужение ядра на координату.*
- $\delta_{i,\cdot}(A) = |core_{i,\cdot}(A)|$

Замечание 2.

$$\delta(A) = \sum_i \delta_{i,\cdot}(A) = \sum_j \delta_{\cdot,j}(A)$$

Лемма 2. *Существует структура данных $LCO(A)$ (local core oracle), которую можно построить за $O(p + q + \delta(A))$ имея сжатое представление матрицы A. $LCO(A)$ должна поддерживать следующий интерфейс:*

- *Boundary access (BA): получение $A_{0,j}$ или $A_{i,0}$ за $O(1)$.*
- *Vertically adjacent recomputation (VAR): получение $A_{i+1,j} - A_{i,j}$ за $O(\delta_{i,\cdot}(A) + 1)$.*
- *Horizontally adjacent recomputation (HAR): получение $A_{i,j+1} - A_{i,j}$ за $O(\delta_{\cdot,j}(A) + 1)$.*

Доказательство. Сохраним в эту структуру данных: левый столбец, верхнюю строчку и два списка списков $[core_{i,\cdot}(A)]$ и $[core_{\cdot,j}(A)]$. Это делается за $O(p + q + \delta(A))$.

Проверим, что данная структура имплиментирует описанный интерфейс.

- BA: очевидно.
- VAR: Мы знаем, что $A_{i,j} + A_{i+1,0} = A_{i+1,j} + A_{i,0} + \delta^\Sigma(A[i..i+1][0..j])$. $\delta^\Sigma(A[i..i+1][0..j])$ можно получить быстро за счет итерирования по $core_{i,\cdot}(A)$.

- HAR: Аналогично VAR.

□

Лемма 3. *По сжатому представлению A можно построить сжатое представление смежной подматрицы A' за $O(p+q+\delta(A))$.*

Доказательство.

1. Строим $LCO(A)$.
2. $core(A')$ строим простой фильтрации $core(A)$.
3. Столбец и строчку получаем с помощью VAR и HAR.

□

Лемма 4 (Матричное сжатие и разжатие)**.** *Для монжевых матриц можно построить две операции.*

Compress *: По CR $A^* \in M_{p^* \times q}$ и CR $B^* \in M_{q \times r^*}$ можно построить CR $A \in M_{p \times q}$ и CR $B \in M_{q \times r}$ за $O(p^* + q + r^* + \delta(A^*) + \delta(B^*))$:*

- $p \leqslant \delta^{\Sigma}(A^*) + 1$ *и* $r \leqslant \delta^{\Sigma}(B^*) + 1$.
- $\delta(A^*) = \delta(A)$ *и* $\delta(B^*) = \delta(B)$

Decompress *: По CR $A \otimes B$ можно построить CR $A^* \otimes B^*$ за $O(p^* + q + r^* + \delta(A^*) + \delta(B^*))$.*

Доказательство. Если у нас матрица и так достаточно плотная (то есть размер ядра не меньше одной из размерностей), то compress ничего не делает. Если же матрица разреженная (то есть размер ядра меньше одной из размерностей), тогда у нас есть размерности в которых нет элемента ядра, а тогда мы можем просто их удалить, а после удаления их восстановить в $A^* \otimes B^*$. □

Теперь мы можем перейти к доказательству теоремы (1).

Доказательство. Алгоритм будет рекурсивный.

1. **Сжатие**: Уменьшаем размер матриц, сохраняя ядро.
2. **Разделение**: Рекурсивно разбиваем матрицы на подматрицы.
$$A = [A^L \ A^R] \quad B = \begin{bmatrix} B^L \\ B^R \end{bmatrix}$$
3. **Рекурсивный вызов**: Посчитаем $C^L = A^L \otimes B^L$, $C^R = A^R \otimes B^R$.
4. **Вычисление границы**: Мы знаем, что $C = \min(C^L, C^R)$ (поэлементный минимум). Так как $\mathcal{W}^{A,B}$ — монотонна по столбцам и строкам, то с одной стороны у нас должны быть элементы матрицы C^L, а с другой элементы C^R. Значит нужно лишь посчитать границу между ними, это можно сделать за линейное время.

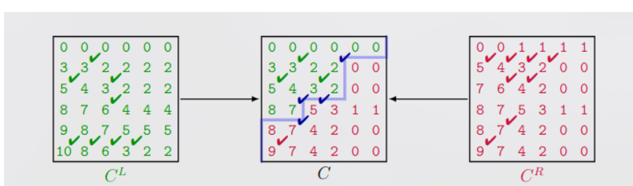

Рис. 3: Склейка матриц C^L и C^R.

То есть мы получили рекурсивный алгоритм, где одна итерация линейна, значит итоговый алгоритм будет за условные $O(n \log n)$. □