Первый курс, осенний семестр 2020/21
Практика по алгоритмам #2

Голодные венгры

13 сентября

Собрано 18 сентября 2021 г. в 09:00

Содержание

Голодные венгры

1. **Татт, Ловас, Рабин и их друзья**

   Дан произвольный граф. Найдите размер максимального паросочетания. Найдите само максимальное паросочетание.

2. **Взвешенное покрытие циклами**

   Дан взвешенный орграф. Покрыть все его вершины простыми непересекающимися циклами минимального суммарного веса.

3. **Покрытие минимальным числом циклов**

   За сколько вы можете покрыть все вершины орграфа минимальным числом простых циклов?

4. **Теорема Дилворта**

   Есть множество, на его элементах задан частичный порядок, описанный как некий орграф. Найти и предъявить максимальное по размеру множество попарно несравнимых элементов.

5. **Стабильные паросочетания**

   a) Приведите пример, когда стабильное паросочетание не единственно.
   b) Найдите стабильное паросочетание, которое максимизирует $\sum$ профит для мальчиков.
   c) Проверить единственность стабильного паросочетания.

6. **Венгерка и понимание**

   a) Работает ли венгерский алгоритм в матрицах, где могут быть отрицательные веса?
   b) Умеем искать паросочетание минимального веса. Как найти максимального?
   c) Модифицируйте венгерский алгоритм для поиска по прямоугольной матрице паросочетания размера $\min(n, m)$. Оцените сложность.

7. **Уравнивание деревьев (генеология)**

   Даны два корневых дерева. У каждого дерева можно отрезать вершины с их поддеревьями, менять порядок детей у вершины. Сделать данные деревья равными, максимизировав число оставшихся вершин.

   (*). А за сколько работает? Говорят, $\mathcal{O}(V^3)$.

8. **Венгры в разреженном графе**

   Дан двудольный граф. Найти **максимальное** паросочетание минимального веса.

   a) $\mathcal{O}(VE \log V)$,
   b) $\mathcal{O}(VE + V^2 \log V)$.

9. **Кредитные операции – 2**

   По заданной матрице a_{ij} найти такие вектора x и y, что $x_i + y_j \geqslant a_{ij}$, а $\sum x_i + \sum y_j \to \min$. Дополнительно известно, что матрица и квадратная, и неотрицательная.

10. **(*) Покрытие циклами неорграфа**
    Дан произвольный **не**ориентированный граф.
    Покрыть все вершины простыми непересекающимися циклами.

11. **(*) Хитрая матрица**

    Дана матрица A, придумайте матрицу $B\colon \forall i\, j\; b_{ij} \geqslant a_{ij}, \sum(b_{ij} - a_{ij}) \to \min$,
    при этом для матрицы B должно выполняться свойство: $\forall i, j \quad b_{i,j+1} + b_{i+1,j} = b_{i,j} + b_{i+1,j+1}$.

12. **(*) Быстрая рёберная раскраска**

    Пусть d – максимальная степень вершины. Идея алгоритма раскраски рёбер графа в d цветов: добавляем рёбра по очереди, у концов есть свободные цвета a и b...

    Задача: придумайте алгоритм покраски рёбер двудольного графа в d цветов за $\mathcal{O}(VE)$. Почему так можно только для двудольного?

Разбор задач практики

1. **Татт, Ловас, Рабин и их друзья**

   Решение без линейной алгебры.
   Бинпоиском ищем $\max|M| = k$, внутри бинпоиска добавляем $n - 2k$ фиктивных вершин, соединённых со всеми, проверяем наличие совершенного паросочетания.
   Осталось восстановить совершенное паросочетание.
   Перебираем p_1 – с кем соединена первая вершина, удаляем 1 и p_1 из графа, проверяем наличие совершенного (альтернатива: перебирать ребро, удалять ребро). В обоих случаях нам нужно будет E раз считать определитель. Итого: $\mathcal{O}(V^3E)$.
   Можно заметить, что при переборе p_1 матрицы, от которых мы считаем определители, отличаются лишь одной строкой, поэтому за $\mathcal{O}(V^3)$ можно найти их всех $\Rightarrow$ суммарное время работы $\mathcal{O}(V^4)$.

2. **Взвешенное покрытие циклами**

   Так же, как невзвешенное, раздвоим граф и ищем совершенный парсоч. Каждое ребро парсоча войдет в покрытие, значит, надо парсоч минимального веса. $\mathcal{O}(n^3)$.

3. **Покрытие минимальным числом циклов**

   Если есть Гамильтонов цикл, то ответ – это ровно он. Значит, задача не проще Гамильтонова цикла. Динамикой по подмножествам за $\mathcal{O}^*(2^n)$ она решается.

4. **Теорема Дилворта**

   Алгоритм. Транзитивно замкнем граф. Раздваиваем вершины v на v_L и v_R, проводим рёбра исходного графа из левой доли в правую. В полученном графе находим min вершинное покрытие, выкидываем из исходного графа вершину v, если хоть одна из v_L или v_R вошла туда.

   Корректность. Для любого ребра (v, u) хоть одна из v_L, u_R вошла в покрытие, значит, выкинем хоть одну из v или u, значит, в исходном графе осталось независимое множество вершин («антицепь»).

   Оптимальность. max антицепь $\leqslant$ min число путей, не более элемента на каждом пути. Если найти ту, которая $\geqslant$ числа путей, то победа.
   Если размер паросочетания k, то путей $n - k$. Размер покрытия тоже k, после выкидывания осталось $\geqslant n - k$ вершин исходного графа.
   Теорема Дилворта как раз и гласит, что максимальный размер антицепи равен минимальному размеру разбиения на пути.

5. **Стабильные паросочетания**

   a) Пример, когда не единственное: списки мальчиков $A_1 = \{1, 2\}, A_2 = \{2, 1\}$, а списки девочек наоборот $B_1 = \{2, 1\}, B_2 = \{1, 2\}$. Оба паросочетания стабильны.

   b) Алгоритм «мальчики предлагает, девочки отказывают» выкидывает только те рёбра, которые не входят ни в одно стабильное паросочетание.
   Действительно: пусть $A_i.\texttt{front} = A_j.\texttt{front} = k$ и девочка k отказала мальчику j.
   Рассмотрим любое паросочетание, где они образуют пару: k больше хочет к i, а i больше хочет к k, т.к. она для него самая лучшая $\Rightarrow$ паросочетание не стабильно.
   В итоге каждый мальчик будет в паре с лучшей для себя из гипотетически возможных

девочек. То есть оптимизирован даже каждый отдельгный мальчик.

c) Алгоритм «девочки предлагают, мальчики отказываются» дает оптимальный для каждой девочки ответ.

Запустим оба алгоритма. Если ответы не совпали, то не единственное. Если совпали, то любое другое паросочетание не стабильно: если в паросочетании нет какой-то пары из нашего, то добавим ее, этой паре стало лучше.

6. **Венгерка и понимание**

a) *Отрицательные веса.* Сперва нужно понять, что проблема в том, что мы заранее не вычитаем минимум из столбцов. Но в конце все числа точно не отрицательные, а парсоч не нулях $\Rightarrow$ ок.

b) *Паросочетание максимального веса.* max паросочетание на A = min паросочетание на $-A$.

c) *Прямоугольная матрица.* Пусть $n \leqslant m$, тогда развернём матрицу так, чтобы было ровно n строк и n раз ищем дополняющую цепь.

7. **Уравнивание деревьев**

Решение за $\mathcal{O}(n^5)$: динамика $f[v, u]$ – максимальный размер дерева, который можно получить, уравняв деревья с корнями в вершинах v и u. Пересчет – максимальное по весу паросочетание на детях.

Венгерка на матрице размера $n \times m$ работает за время $nm \cdot \min(n, m) \Rightarrow$ наша динамика работает за $\sum_{ij}(\deg_i \deg_j \min(\deg_i \deg_j)) \leqslant (\sum_i \deg_i)(\sum_j \deg_j) \min(n, m) \leqslant nm \cdot \min(n, m)$.

8. **Венгры в разреженном графе**

a) $\mathcal{O}(VE \log V)$. При поиске дополняющего пути нужно $\mathcal{O}(V)$ раз найти вершину правой доли, в которую ведет ребро $e = (v, u)$ с минимальным $a[e]+row_v+col_u$, причем v использовано, u нет. Делаем это кучей: храним в ней минимальное ребро для всех неиспользованных u. Теперь надо пересчитать row и col.

Пусть в какой-то момент мы знаем, что надо было row_v `+=` D для всех $v \in A^-$, и col_u `-=` D для всех $u \in B^-$. Не будем этого делать явно, вместо этого делаем S `+=` D – запоминаем отложенную сумму.

Все ребра $A^+ \to B^-$ лежат в куче, их вес уменьшен на S, но относительный порядок не изменился.

Когда вынимаем минимальное ребро $e = (v, u)$, нужно обработать ребра из $p[u]$.

Их веса $(A^- \to B^-)$ мы не меняли, как и явно хранимые в куче. Но и те, и другие должны измениться на новое S. Так что просто делаем `DecreaseKey` для соседей $p[u]$.

Пересчитаем S. Теперь можно явно сделать col_u `-=` S и $row_{p[u]}$ `+=` S.

Теперь для u и $p[u]$ все корректно пересчитано. И так будет со всеми, кто попадает из A^- в A^+ и из B^- в B^+.

После того, как нашли дополняющую цепь из очередной v, делаем row_v `+=` D для оставшихся $v \in A^-$, и col_u `-=` D для оставшихся $u \in B^-$.

$\mathcal{O}(V)$ внешних итераций, каждая за $\mathcal{O}(V \log V + E \log V + V = \mathcal{O}(V + E) \log V)$: V извлечений из кучи, E `DecreaseKey`, за V пересчет оставшихся row_v и col_u.

b) $\mathcal{O}(VE + V^2 \log V)$. То же самое, но куча Фибоначчи.

9. **Кредитные операции - 2**

Венгерка на матрице $-A$ ищет такие x и y, что есть совершенное паросочетание по нулям, а остальные $-a_{ij} + x_i + y_j \geqslant 0$.

Тогда выполнено $x_i + y_j \geqslant a_{ij}$.
При любом паросочетании $x_i + y_j \geqslant a_{ij} \Rightarrow \sum x_i + \sum y_j = \sum(x_i + y_{p[i]}) \geqslant \sum a_{i,p[i]}$ (тут нам пригодилась квадратность).
У нас паросочетание по нулям $\Rightarrow -a_{i,p[i]} + x_i + y_{p[i]} = 0 \Rightarrow \sum x_i + \sum y_j = \sum a_{i,p[i]}$. Значит, минимально.

10. **(*) Покрытие циклами неорграфа**

Пусть у вершины v степень d. Заведем вершины $v_{L,1}, \ldots, v_{L,d}, v_{R,1}, v_{R,(d-2)}$, строим полный двудольный граф на наборах v_L, v_R.
Пусть соседи каждой вершины как-то упорядочены. Если есть ребро (v, u), где v – i-й сосед u, а u – j-й сосед v, то проводим ребро $(v_{L,j}, u_{L,i})$. В полученном графе ищем совершенное паросочетание.
В совершенном покрыты все v_R, значит, из левой половины v во внешний мир выходит ровно два ребра. Получилось как раз покрытие циклами.

Если изначально двудольный гаджет каждой вершины инициализировать паросочетанием размера $(d-2)$, то останется набрать $2V$ ребер, $\mathcal{O}(V)$ запусков поиска дополняющего пути. Ребер в новом графе $\mathcal{O}(E + \sum \deg_v^2)$. Но можно в каждой двудольном гаджете провести Cd ребер так, чтобы для $\forall$ подмножества v_L существовало паросочетание, покрывающее его. Здесь C – константа. Например, если из каждой вершины левой доли провести 4 случайных ребра в правую, с высокой вероятностью нам повезет. Так можно сделать $\mathcal{O}(E + \sum \deg_v) = \mathcal{O}(E)$ ребер.
Итого можно решить задачу за $\mathcal{O}(\texttt{Matching}(V, E))$.

11. **(*) Хитрая матрица**

$b_{i,j+1} + b_{i+1,j} = b_{i,j} + b_{i+1,j+1} \Leftrightarrow b_{i,j+1} - b_{i,j} = b_{i+1,j+1} - b_{i+1,j}$.
Отсюда видим, что каждый столбец получается из следующего прибавлением к нему одного и того же числа.
Значит, если x – первый столбец матрицы B, то есть некий вектор y: $b_{ij} = x_i + y_j$.
Получили задачу «Кредитные операции - 2».

12. **(*) Быстрая рёберная раскраска**

Хотим добавить ребро (v, u), $\exists a, b$: у вершины v нет ещё цвета a, у u нет b. Если $a = b$, красим, всё ок. Иначе пытаемся сделать, чтобы у u тоже не было a, берём соседнее ребро цвета a и перекрашиваем в b, а дальше перекрашиваем $b \to a$ и т.д. пока не упрёмся. Если умеем делать переход вперёд за $\mathcal{O}(1)$, всё перекрашивание займёт $\mathcal{O}(V)$.

Домашнее задание

3.1. Обязательная часть

1. **(2) Самолёты**

   Есть самолёты, у каждого самолёта есть время вылета t_i и вместимость $1 \leqslant k_i \leqslant 10$. Есть пассажиры, у каждого есть допустимый отрезок времён вылета $[l_i, r_i]$. Все самолёта и пассажиры летят/хотят из одного и того же пункта A в один и тот же пункт B. Нужно отправить максимальное число пассажиров.

2. **(2.5) Сведение**

   Дан двудольный граф и чёрный ящик `maxMatching(minVertexCover)`. Пусть чёрный ящик корректен и работает за $T(V, E)$. Найдите **совершенное** паросочетание за $\mathcal{O}(T(V, E)+V+E)$.

   По сути мы доказываем, что задача «найти совершенное паросочетание» не сложнее, чем задача «по данному минимальному вершинному покрытию найти максимальное паросочетание».

3. **(1.5) Венгерский алгоритм**

   Предъявите матрицу размера $n \times n, n = 3$, на которой $k = 2$ итераций внешнего цикла венгерского алгоритма найдут паросочетание размера k, не являющееся минимальным среди всех паросочетаний размера k. Ответ поясните.

   Венгерский алгоритм ищет дополняющие пути от вершин в порядке $0, 1, \ldots, n-1$.

4. **(3) Сделаем венгерский алгоритм полезнее**

   a) **(1.5)** Найдите максимальное по весу паросочетание произвольного размера.
   b) **(1.5)** Найдите максимальное по весу паросочетание размера ровно k.

5. **(3) Подпоследовательности и** $\gcd$

   Разбить массив на подпоследовательности так, что сумма $\gcd(a, b)$ по всем парам $\langle a, b\rangle$ соседних элементов всех последовательностей максимальна. $\mathcal{O}(n^3)$. Каждый элемент массива должен попасть ровно в одну подпоследовательность.

6. **(3) ATSP**

   Дан полный граф. Граф задан несимметричной матрицей неотрицательных весов. Задача maxATSP – найти в нём максимальный по весу гамильтонов цикл. Веса и неотрицательны и удовлетворяют неравенству треугольника. Найдите 2-OPT приближение для maxATSP за полиномиальное время.

3.2. Дополнительная часть

1. **(3+1+1) Покраска дерева**

   Дано дерево. Вершину v можно покрасить в цвет $c \in [1, k]$ за $\texttt{cost}[v, c]$. Покрасить за минимальную цену все вершины так, чтобы расстояние в рёбрах между вершинами одинакового цвета было строго больше двух.

   a) $\mathcal{O}(nk^5)$.
   b) $\mathcal{O}(nk^4)$.
   c) (**) $\mathcal{O}(nk^3)$.

2. **(3) Ящики**

   Есть k ящиков и поле $n \times n$ с дырками. Нам известны начальные позиции ящиков и k точек, в которые должны попасть ящики в конце. Любой ящик можно поставить на любую из данных позиций. За один шаг ящик можно сдвинуть в соседнюю по стороне клетку, если там нет дырки или другого ящика. Найти минимальное число действий, за которое можно переместить ящики из начальных позиций в конечные. Найдите последовательность действий, чтобы получить именно такой ответ.

3. **(3) Тяжёлые самолёты**

   Решите задачу про самолёты для случая 10^5 самолётов, 10^5 пассажиров, вместимость каждого самолёта до 10^5, среди пассажиров есть те, кого обязательно отправить (например, студенты, участники олимпийских игр).

4. **(4) Трисочетание**

   Рассотрим трёх дольный граф. Трисочетание – множество непересекающихся по вершинам треугольников. Найти совершенное трисочетание минимального веса трудно. Предложите хорошее приближение. В каждой доли $n \leqslant 50$ вершин. Часть баллов за идею, часть баллов за оценку приближения.