АиСД у2023. Третий семестр

Домашние задания M3232-M3233

⟨Версия от 24 сентября 2024 г.⟩

Темы

Неделя 1. Обход в глубину

1.1. В произвольном неориентированном графе за время $\mathcal{O}(n+m)$

(a) найдите для каждой вершины количество достижимых из нее

(b) найдите для каждой вершины минимальную по номеру вершину, достижимую из нее

1.2. В произвольном ориентированном графе

(a) найдите для каждой вершины минимальную по номеру вершину, достижимую из нее, за время O(n + m)

(b) постройте транзитивное замыкание за время O(nm)

1.3. Дан ориентированный ацикличный граф. Проверьте, что данный граф имеет ровно одну топологическую сортировку. Время работы: $\mathcal{O}(n+m)$.

1.4. Для ориентированного графа найдите наименьшее множество вершин, из которых достижимы все остальные, за время $\mathcal{O}(n+m)$

1.5. На ориентированном графе запустили обход в глубину, получили лес (множество деревьев) обхода. Может ли вершина с $deg_{in}(v) > 0$ и $deg_{out} > 0$ образовывать в этом лесу дерево из одной вершины?

1.6. Раскраской графа в k цветов называется назначение каждой вершине одного из k цветов. Раскраска называется правильной, если нет пары одноцветных вершин, соединенных ребром. Дан неориентированный граф. Покрасьте его правильным образом в два цвета, либо определите, что это невозможно. Графы, которые можно раскрасить в 2 цвета, называются двудольными. Время работы — $\mathcal{O}(n+m)$.

1.7. Дан граф, проверьте существование нечетного цикла. Подсказка: эта задача связана с предыдущей.

1.8. Есть n человек. Некоторые пары людей дружат. Требуется разбить всех людей на две команды из $\frac{n}{2}$ человек таким образом, чтобы каждая пара друзей попала в разные команды. Время работы — $\mathcal{O}(n^2)$.

1.9. В произвольном неориентированном графе за время $\mathcal{O}(n+m)$, если на каждом ребре написана буква, найдите простой путь из s в t, задающий лексикографически минимальную строку (гарантируется, что ни из какой вершины не выходит два ребра с одной буквой).

1.10. Рассмотрим следующий алгоритм:

```
visited = 0

dfs(v):
    visited += 1
    for (v, u) in E:
        if (visited < n):
            dfs(u)
```

```
for v = 1 ... n:
    if (visited < n):
        dfs(v)
```

Докажите или опровергните, что данный фрагмент кода, запущенный на ацикличном ориентированном графе, работает за время $\mathcal{O}(n+m)$.

1.11. Докажите или опровергните, что фрагмент кода из предыдущей задачи, запущенный на произвольном ориентированном графе, работает за время $\mathcal{O}(n+m)$.

1.12. Дан ацикличный ориентированный граф. Найдите в нем простой путь, состоящий из максимально возможного количества ребер. Простым называется путь, который проходит по каждой вершине и по каждому ребру не более одного раза. Время работы: $\mathcal{O}(n+m)$.

1.13. Дан ацикличный ориентированный граф. Найдите в нем длину кратчайшего пути из s в t, а также количество кратчайших путей из s в t. Время работы: $\mathcal{O}(n+m)$.

1.14. Придумайте алгоритм, который строит лексикографически минимальную топологическую сортировку за $\mathcal{O}((n+m)\log n)$.

1.15. Есть n человек, которых нужно рассадить в два автобуса. Также известны m ограничений. Каждое ограничение может быть одного из двух типов:

- Люди с номерами x_i и y_i **должны** сидеть в одном автобусе;
- Люди с номерами x_i и y_i **не должны** сидеть в одном автобусе.

Найдите любой план рассадки людей по двум автобусам, удовлетворяющий всем ограничениям, либо выясните, что его не существует. Время работы: $\mathcal{O}(n+m)$.

1.16. Есть n гирек, веса которых попарно различны. Вам дана последовательность из m взвешиваний на чашечных весах: в каждом взвешивании участвовали ровно две гирьки, по одной на каждой чаше весов. Результат каждого взвешивания дает информацию, которая из двух гирек легче. Возможно, результаты некоторых взвешиваний некорректны. Определите, после какого минимального количества взвешиваний полученная информация гарантированна стала противоречивой. Время работы: $\mathcal{O}((n+m)\log n)$.

1.17. У вас есть n дел, некоторые дела зависят от других дел: их нельзя сделать раньше тех, от которых они зависят. Всего есть m зависимостей. Граф зависимостей ацикличен. Выполнение каждого дела занимает один час. Дело под номером t является очень важным. Определите порядок выполнения дел, чтобы очень важное дело t было выполнено как можно раньше. Время работы: $\mathcal{O}(n+m)$.

1.18. *Задача о кротах.* Дан неориентированный граф с n вершинами и m ребрами. Требуется разбить множество вершин V на три множества A, B и C: $V = A \sqcup B \sqcup C$, таким образом, чтобы были выполнены следующие условия:

(a) Множество вершин A является путем (можно выбрать $|A| - 1$ ребер, соединяющих некоторые пары вершин A, чтобы получился путь).

(b) В графе не существует ребра (u, v), такого что $u \in B$ и $v \in C$.

(c) $|B| = |C|$.

Разрешается делать некоторые из множеств A, B и C пустыми. Время работы: $\mathcal{O}(n + m)$.

Интересный факт: требуемое разбиение всегда существует.

1.19. Петя решил реализовать алгоритм построения топологической сортировки следующим образом:

```
dfs(v):
    ans.add(v)
    mark[v] = true

    for (v, u) in E:
        if !mark[u]:
            dfs(u)

for v = 1 ... n:
    if !mark[v]:
        dfs(v)
```

Докажите или опровергните, что данный алгоритм на произвольном ациклическом ориентированном графе построит топологическую сортировку.

1.20. Дан неориентированный граф. Найдите минимальный по размеру набор ребер, который нужно удалить из графа, чтобы граф стал ацикличным. Время работы: $\mathcal{O}(n + m)$.

1.21. Дан ориентированный ацикличный граф. Посчитайте количество различных топологических сортировок данного графа. Так как топологическая сортировка — это перестановка вершин, будем считать, что сортировки различны, если отличаются соответствующие им перестановки. Время работы: $\mathcal{O}(2^n \cdot n^2)$.

1.22. Проверьте в графе существование четного и нечетного пути из вершины s во все остальные вершины за время $\mathcal{O}(n + m)$.

Неделя 2. КСС, конденсация графа, 2-SAT

2.23. Какое минимальное и максимальное количество ребер может быть в графе на n вершинах с k компонентами сильной связности? Считайте, что петли и кратные ребра запрещены.

2.24. Приведите контрпример для каждой из следующих версий алгоритма поиска компонент сильной связности. Пояснение: Здесь tin и tout – указание на то, сортировать вершины по времени входа или времени выхода, $\uparrow$ и $\downarrow$ — в порядке возрастания или убывания, а G и G^{rev} — делать dfs по прямым или обратным ребрам.

(a) dfs2 в порядке tin $\uparrow$ по G

(b) dfs2 в порядке tout $\uparrow$ по G

2.25. (a) dfs2 в порядке tin $\downarrow$ по G

(b) dfs2 в порядке tout $\downarrow$ по G

2.26. (a) dfs2 в порядке tin $\uparrow$ по G^{rev}

(b) dfs2 в порядке tout $\uparrow$ по G^{rev}

(c) dfs2 в порядке tin $\downarrow$ по G^{rev}

2.27. Дан ориентированный граф. Определите, какое минимальное число ребер нужно добавить в него, чтобы он стал сильно связным. Время работы: $\mathcal{O}(n+m)$.

2.28. Дан ориентированный граф. Постройте граф, с таким же множеством вершин и минимальным числом ребер, чтобы его конденсация совпадала с конденсацией данного графа. Время работы: $\mathcal{O}(n+m)$.

2.29. (a) Дан ориентированный граф. Найдите минимальное по размеру множество вершин, таких что любая вершина графа была достижима хотя бы из одной вершины из множества. Время работы: $\mathcal{O}(n+m)$.

(b) Та же задача, но теперь выбрать вершину номер i стоит c_i денег. Требуется выбрать самое дешевое множество вершин, из которых достижимы все оставшиеся.

2.30. Дан ориентированный граф. По некоторым ребрам можно перемещаться, только имея пропуск, а по остальным ребрам можно перемещаться, даже не имея пропуска. Изначально вы находитесь в вершине s, и у вас есть пропуск. Выясните, существует ли возможность потерять пропуск в графе: это значит, что вы должны пройти некоторый путь из вершины s, затем оставить пропуск в какой-то вершине, после этого пройти еще некоторый путь из этой вершины, после чего у вас не должно быть возможности вернуться в вершину, в которой вы оставили пропуск. Время работы: $\mathcal{O}(n+m)$.

2.31. *Турниром* называется ориентированный граф, в котором каждая пара вершин соединена ребром (в одну или другую сторону). *Гамильтонов путь* — это путь в графе, проходящий по каждой вершине ровно один раз. Докажите, что в любом турнире существует гамильтонов путь.

2.32. *Гамильтонов цикл* — это цикл в графе, проходящий по каждой вершине ровно один раз. Докажите, что в любом сильно связном турнире существует гамильтонов цикл.

2.33. Докажите, что если в каждой вершине турнира число входящих и исходящих ребер одинаково, то турнир сильно связен.

2.34. Дана задача 2-SAT, причем гарантируется, что для каждой скобки вида $x \vee y$ существует парная скобка $\bar{x} \vee \bar{y}$, а также для каждой скобки вида $\bar{x} \vee y$ существует парная скобка $x \vee \bar{y}$. Найдите решение такой задачи 2-SAT, в котором будет минимальное количество переменных, равных единице. Время работы: полином от n.

2.35. Дана задача 2-SAT, требуется найти ее лексикографически минимальное решение. Решением считается последовательность значений переменных $x_1, x_2, \ldots, x_n$, минимизировать нужно ее. Время работы: полином от n.

2.36. Дана задача 2-SAT и ее решение. Найдите следующее решение в лексикографическом порядке. Время работы: полином от n.

2.37. Есть n различных целых чисел $a_1, a_2, \ldots, a_n$, а также два целых числа x и y. Требуется разделить все числа a_i на два множества A и B таким образом, чтобы были выполнены следующие условия:

- ▷ Если $t \in A$, то $(t - x) \in A$
- ▷ Если $t \in B$, то $(t - y) \in B$

Время работы: полином от n.

2.38. Найдите правильную раскраску неориентированного графа в три цвета, если для каждой вершины дан один цвет, в который ее запрещено красить.

2.39. Есть n городов, расположенных на окружности по часовой стрелке в порядке возрастания номеров. Каждые два соседних города (i и $i+1$ или 1 и n) соединены дорогой. Требуется построить еще m дорог между определенными городами (v_i, u_i), при чем каждую из них можно провести либо внутри окружности, либо снаружи. Определите, можно ли построить все необходимые дороги, чтобы никакие две не пересекались, за время $\mathcal{O}(n^2 + m)$.

Неделя 3. Мосты и точки сочленения. Двусвязность

3.40. Строго докажите, почему при поиске мостов или точек сочленения можно использовать как $depth[v]$, так и $time_{in}[v]$ с одинаковой эффективностью.

3.41. Дан неориентированный граф. Необходимо определить, сколько ребер нужно добавить в него, чтобы он стал реберно двусвязным? Время работы: $\mathcal{O}(n+m)$.

3.42. Дан неориентированный связный граф. Необходимо ориентировать его ребра так, чтобы получился сильно связный граф. Время работы: $\mathcal{O}(n+m)$.

3.43. Какое максимальное число мостов может быть в графе с n вершинами?

3.44. Какое максимальное число точек сочленения может быть в графе с n вершинами?

3.45. Дан неориентированный граф. Необходимо за $\mathcal{O}(\log n)$ отвечать на запросы: «для заданных вершин u и v сколько существует ребер, по которым мы обязательно пройдем, если будем идти из u в v». Препроцессинг: $\mathcal{O}(n\log n+m)$.

3.46. Дан неориентированный граф. Необходимо за $\mathcal{O}(\log n)$ отвечать на запросы: «для заданных вершин u и v сколько существует вершин, по которым мы обязательно пройдем, если будем идти из u в v». Препроцессинг: $\mathcal{O}(n\log n+m)$.

3.47. Дан неориентированный граф. На каждом ребре записано число. Необходимо за $\mathcal{O}(\log n)$ отвечать на запросы: «для заданных вершин u и v какое максимальное число можно встретить на реберно простом пути из u в v». Препроцессинг: $\mathcal{O}(n\log n+m)$.

3.48. Дан неориентированный граф. На каждом ребре записано число. Необходимо за $\mathcal{O}(\log n)$ отвечать на запросы: «для заданных вершин u и v какое максимальное число можно встретить на вершинно простом пути из u в v». Препроцессинг: $\mathcal{O}(n\log n+m)$.

3.49. Есть n компьютеров, некоторые m пар компьютеров соединены проводами. Также имеется очень важный файл, который можно сохранить на некоторых компьютерах, причем должно быть возможно доставить этот файл на любой компьютер, используя провода. Найдите минимальное количество компьютеров, на которые следует сохранить файл, при условии, что любой **один** провод может сломаться и перестать передавать данные, чтобы было достигнуто описанное выше условие. Время работы: $\mathcal{O}(n+m)$.

3.50. Для данного ориентированного графа найдите все ребра, лежащие на

(a) каком-то

(b) любом

пути из v в u.

3.51. Докажите или опровергните следующие утверждения:

▷ Все ребра, исходящие из точки сочленения, являются мостами

▷ Из каждой точки сочленения исходит хотя бы один мост

- ▷ Если все ребра, исходящие из вершины, являются мостами, то эта вершина является точкой сочленения

3.52. Докажите или опровергните следующие утверждения:

- ▷ Если из вершины исходит хотя бы один мост, то она является точкой сочленения
- ▷ Если из вершины исходят хотя бы два моста, то она является точкой сочленения
- ▷ Если вершина лежит на простом цикле, то она не может быть точкой сочленения

3.53. Дан связный неориентированный граф. Найдите количество способов удалить из графа два ребра таким образом, чтобы он стал несвязным. Время работы: $\mathcal{O}(n^2)$.

3.54. Назовем граф реберным кактусом, если каждое его ребро лежит не более, чем на одном простом цикле. По данному графу за $\mathcal{O}(n+m)$ определите, является ли он реберным кактусом. Если да, то найдите все его циклы.

3.55. Дан реберный кактус. Научитесь отвечать на запросы: «найти количество простых путей из вершины u в вершину v» за $\mathcal{O}(\log n)$. Препроцессинг: $\mathcal{O}(m + n \log n)$.