Графы. Поиск в глубину.

Разбор задач

А. Конвертер графов

- Дан граф в одном из трех форматов (список смежности, список ребер, матрица смежности)
- Необходимо вывести этот граф в одном из этих же трех форматов (заданном)

А. Конвертер графов

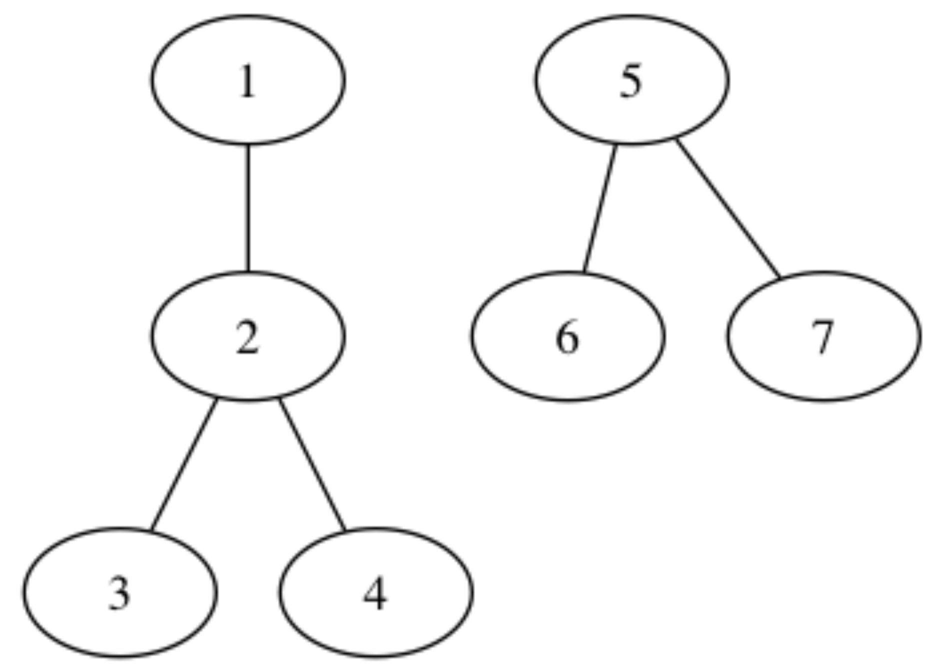
- Удобно считать граф в матрицу смежности, поскольку из нее достаточно легко получить остальные представления графа

B. Модули

- Представим, что все модули программ - вершины некоторого графа. Тогда между вершинами i и j есть ребро, если i-му модулю нужен j-й для функционирования (и наоборот, j-му модулю нужен i-й).

Пример для данного списка ребер:

1 2
2 3
2 4
5 6
5 7



B. Модули

- Получим неориентированный граф. Теперь чтобы посчитать сколько минимальных программ можно собрать из модулей, нужно посчитать количество компонент связности в этом графе.

- Для того, чтобы найти количество компонент связности, воспользуемся поиском в глубину

B. Модули

После того, как мы построим граф, запустим поиск в глубину из каждой еще непосвященной вершины

В переменной ans будет храниться количество компонент связности

```
procedure Explore(v: Integer);
var
    i: Integer;
begin
    used[v] := true;
    for i := 0 to High(adj[v]) do
        if not used[adj[v][i]] then
            Explore(adj[v][i]);
end;


procedure Dfs;
var
    v: Integer;
begin
    for v := 1 to MaxV do
        if not used[v] then begin
            ans += 1;
            Explore(v);
        end;
end;
```

C. КПК

- Дан неориентированный граф. Необходимо добавить и удалить некоторое минимальное количество ребер таким образом, чтобы граф стал ациклическим и связным

- Для начала разберемся с тем, какие ребра необходимо удалить

С. КПК

- Запустим поиск в глубину из некоторой вершины. Заметим, что мы пройдем не по всем ребрам текущей компоненты связности, а только по некоторым.

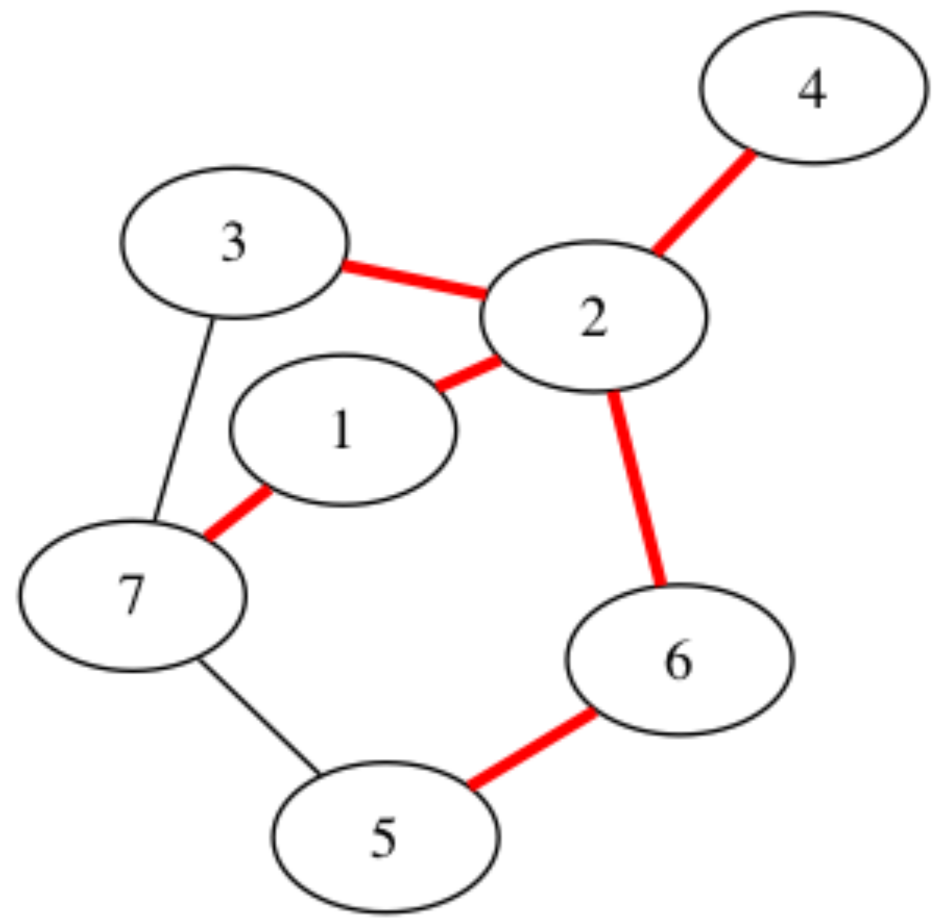
- Ребра, по которым мы прошли образуют связный ациклический подграф (как раз то, что нам нужно).

- Тогда все те ребра, по которым мы не прошли, нужно выбросить

C. КПК

Красным отмечены ребра, по которым мы прошли.

Красные ребра нужно оставить, а черные - удалить



С. КПК

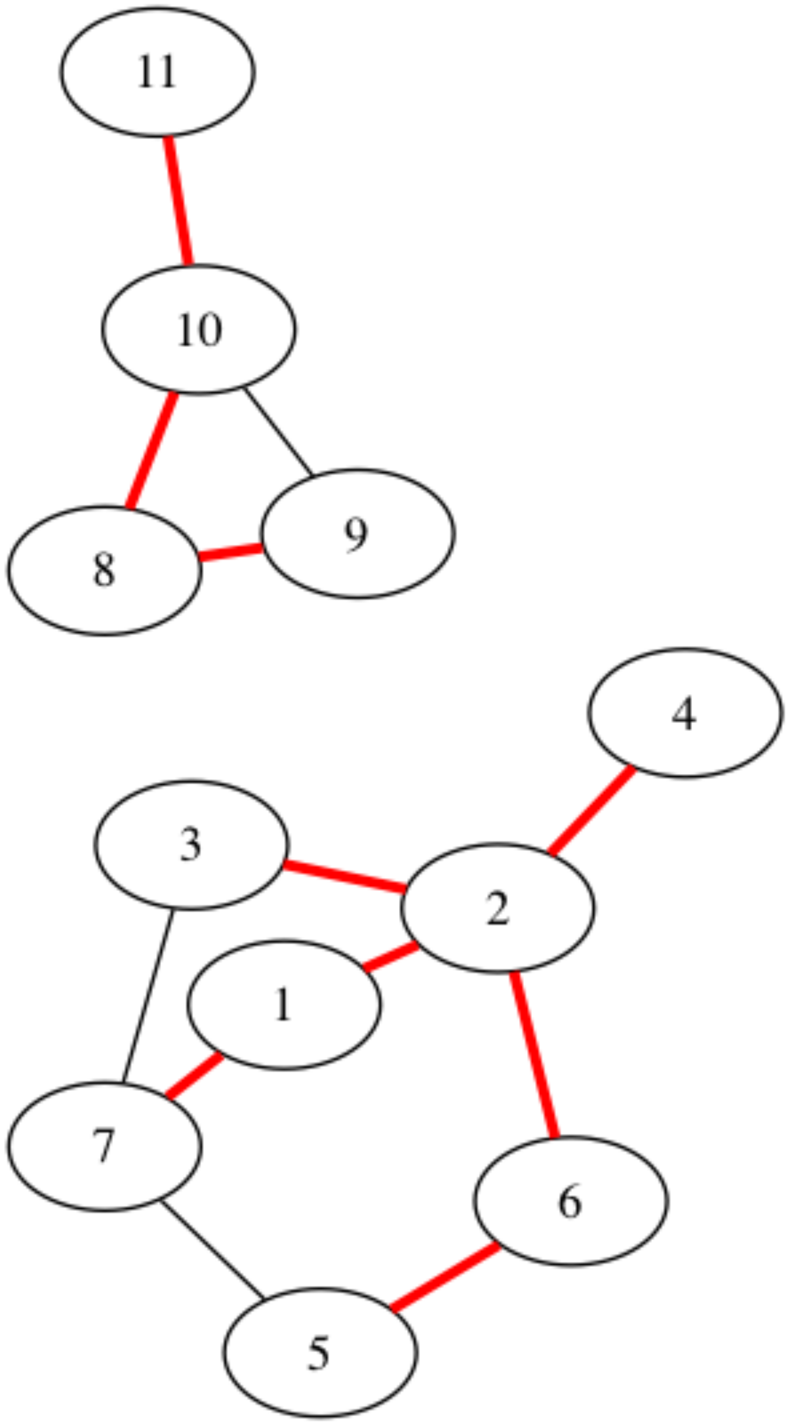
- Таким образом, запустив поиск в глубину из каждой непосещенной вершины, мы поймем, какие ребра нужно удалить.

- Добавлять ребра нужно в том случае, если компонент связности несколько, то есть исходный граф был несвязным

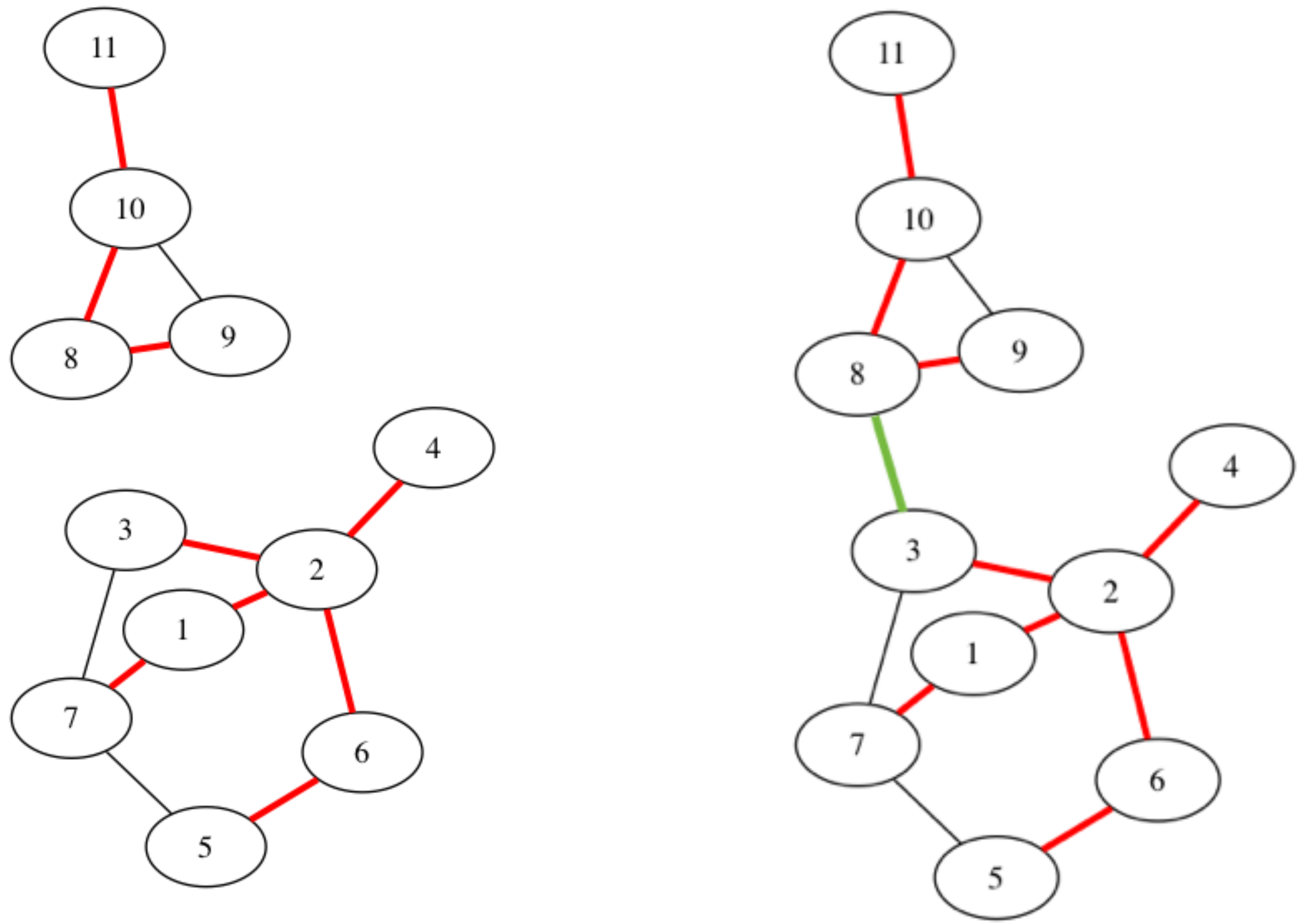
C. КПК

- В этом примере, после удаления черных ребер, итоговый граф будет несвязным.

- Теперь нужно добавить ребра между компонентами связности



C. КПК



Зеленым обозначено добавленное ребро

C. КПК

Почему полученный ответ оптимален?

D. Водовод на о. Русский

- Дан неориентированный 4х-связный граф.
- Необходимо определить есть ли путь из “начала” этого графа в “конец”, и вывести его. Если таких путей несколько, то нужно вывести 2 пути.

“конец” - последний

```
..###.
###.##
..###.
```

“Начало” - первый столбик

D. Водовод на о. Русский

- Для того чтобы определить, существует ли хотя бы один путь, достаточно запустить поиск в глубину из всех вершин “начала”. Если таким образом мы смогли добраться до хотя бы одной вершины “конца”, то такой путь (хотя бы один) существует.

- Если путь существует, теперь необходимо определить, единственный ли он

D. Водовод на о. Русский

- Для этого можно снова запустить поиск в глубину, но перебирать ребра в другом порядке (например, с конца)
- Если таким образом мы получим другой путь - то путей несколько, а если такой же - то путь всего один