Графы 2

Простыми словами, обход графа — это переход от одной его вершины к другой в поисках свойств связей этих вершин. Связи (линии, соединяющие вершины) называются направлениями, путями, гранями или ребрами графа. Вершины графа также именуются узлами.  
  
Двумя основными алгоритмами обхода графа являются поиск в глубину (Depth-First Search, DFS) и поиск в ширину (Breadth-First Search, BFS).  
  
Несмотря на то, что оба алгоритма используются для обхода графа, они имеют некоторые отличия. Начнем с DFS.

### Поиск в глубину

DFS следует концепции «погружайся глубже, головой вперед» («go deep, head first»). Идея заключается в том, что мы двигаемся от начальной вершины (точки, места) в определенном направлении (по определенному пути) до тех пор, пока не достигнем конца пути или пункта назначения (искомой вершины). Если мы достигли конца пути, но он не является пунктом назначения, то мы возвращаемся назад (к точке разветвления или расхождения путей) и идем по другому маршруту.

\*нарисовать пример DFS\*

Давайте проанализируем этот алгоритм. Поскольку мы обходим каждого «соседа» каждого узла, игнорируя тех, которых посещали ранее, мы имеем время выполнения, равное O(V + E).  
  
Краткое объяснение того, что означает V+E:  
  
V — общее количество вершин. E — общее количество граней (ребер).  
  
Может показаться, что правильнее использовать V\*E, однако давайте подумаем, что означает V\*E.  
  
V\*E означает, что применительно к каждой вершине, мы должны исследовать все грани графа безотносительно принадлежности этих граней конкретной вершине.  
  
С другой стороны, V+E означает, что для каждой вершины мы оцениваем лишь примыкающие к ней грани. Возвращаясь к примеру, каждая вершина имеет определенное количество граней и, в худшем случае, мы обойдем все вершины (O(V)) и исследуем все грани (O(E)). Мы имеем V вершин и E граней, поэтому получаем V+E.  
  
Далее, поскольку мы используем рекурсию для обхода каждой вершины, это означает, что используется стек (бесконечная рекурсия приводит к ошибке переполнения стека). Поэтому пространственная сложность составляет O(V).

\*реализовываем DFS на произвольном графе\*

### Поиск в ширину

Вместо того, чтобы двигаться по определенному пути до конца, BFS предполагает движение вперед по одному соседу за раз. Это означает следующее. Вместо следования по пути, BFS подразумевает посещение ближайших к s соседей за одно действие (шаг), затем посещение соседей соседей и так до тех пор, пока не будет обнаружено t.  
  
Далее возникает вопрос: как узнать, каких соседей следует посетить первыми?

Для этого мы можем воспользоваться концепцией «первым вошел, первым вышел» (first-in-first-out, FIFO) из очереди (queue). Мы помещаем в очередь сначала ближайшую к нам вершину, затем ее непосещенных соседей, и продолжаем этот процесс, пока очередь не опустеет или пока мы не найдем искомую вершину.

\*нарисовать пример BFS\*

Может показаться, что BFS работает медленнее. Однако если внимательно присмотреться к визуализациям, можно увидеть, что они имеют одинаковое время выполнения.  
  
Очередь предполагает обработку каждой вершины перед достижением пункта назначения. Это означает, что, в худшем случае, BFS исследует все вершины и грани.  
  
Несмотря на то, что BFS может казаться медленнее, на самом деле он быстрее, поскольку при работе с большими графами обнаруживается, что DFS тратит много времени на следование по путям, которые в конечном счете оказываются ложными. BFS часто используется для нахождения кратчайшего пути между двумя вершинами.  
  
Таким образом, время выполнения BFS также составляет O(V + E), а поскольку мы используем очередь, вмещающую все вершины, его пространственная сложность составляет O(V).

\*реализовываем BFS на произвольном графе\*

На практике эти два алгоритма отличаются областями задач, в которых их применяют. Так например, для поиска конкретной вершины в графе используется DFS, BFS же обычно используется для подсчета расстояний между вершинами.

Далее решаем задачки:

[Обход графа](https://informatics.msk.ru/mod/statements/view.php?id=256#1) , [Баобаб](https://informatics.msk.ru/mod/statements/view.php?id=256&chapterid=111541#1) , [Банкет](https://informatics.msk.ru/mod/statements/view.php?id=256&chapterid=165#1) , [Удаление клеток](https://informatics.msk.ru/mod/statements/view.php?id=256&chapterid=652#1) , [Путь в графе](https://informatics.msk.ru/mod/statements/view.php?id=255#1) , [Числа](https://informatics.msk.ru/mod/statements/view.php?id=255&chapterid=2001#1) (жесть)