Алгосы, часть ii

Денис Осипов, Иван Ермошин, Егор Нечаев

25 августа 2020 г.

Введение

Этот проект – коллективный **конспект** по второй части курса «Математические основы алгоритмов», впервые прочитанного первокурсникам МКН СПбГУ в первой половине ii семестра 2020 года Эдуардом Алексеевичем Гиршем.

Актуальные исходники: `https://www.overleaf.com/read/hnbkrkyknbpk` и `https://github.com/gogochushij/algosi-hirsch`

Если вы хотите **принять участие** в написании билетов, или же **сообщить об ошибке**, напишите `http://vk.com/gogochushij`. Предполагается, что каждый автор напишет около 4 билетов, но мы будем рады любой посильной помощи. Здесь можно посмотреть, с какими билетами вы можете помочь проекту.

Последнее обновление публичной версии конспекта: 25 августа 2020 г.

Содержание

1 (19) (В РАЗРАБОТКЕ) Алгоритм Борувки для MST. Линейный вероятностный алгоритм для MST. (Осипов Д.)

1.1 Алгоритм Борувки

Шаг Борувки – это алгоритм, который сводит задачу поиска миностова у графа к той же задаче, но с меньшим числом вершин у графа. Алгоритм Борувки – многократное применение шага Борувки. Шаг Борувки базируется на следующей лемме:

Лемма (о безопасных ребрах для алгоритма Борувки)**.** *Для всякой вершины $v \in V$ хотя бы одно смежное с v ребро минимального веса входит в любое минимальное остовное дерево.*

Warning! *Авторское доказательство .* Если все смежные в v ребра имеют одинаковый вес, то доказывать нечего – вершина v должна быть покрыта хоть каким-то смежным с ней ребром. Пусть теперь среди смежных с v ребер есть ребра веса, строго большего, чем минимальный.

Пусть T – какой-то минимальный остов. Предположим, что ни одно из ребер, смежных с v и имеющих среди них минимальный вес, не входит в T. Пусть (v, w) – любое такое ребро. Так как T – остов, то вершина v покрыта более тяжелым ребром из него, пусть (v, u). Добавим (v, w) в T, тогда $P + (w, v) + (v, u)$ есть цикл в $T \cup (v, w)$, проходящий через v. Удаление ребра (u, v) разрушит этот цикл, и полученное множество ребер $T' = T \smallsetminus (v, u) \cup (v, w)$ будет снова остовным деревом. Но вес T' будет строго меньше веса T – противоречие с минимальностью. $\square$

Шаг Борувки.

1. Для каждой вершины $v \in V$ помечаем смежное с ней ребро минимального веса. Если таких ребер несколько, выбираем ребро с наименьшим номером.
2. Определим компоненты связности на помеченных ребрах.
3. Каждую компоненту связности стянем в одну вершину. Некоторые ребра при этом станут петлями или мультиребрами.
4. Все петли уберем, а в мультиребрах оставим только ребра минимального веса.

Корректность алгоритма Борувки заключается в следующем утверждении:

Теорема. *Пусть шаг Борувки получил из графа G граф G'. Тогда миностов графа G есть миностов графа G' плюс помеченные в этом шаге Борувки ребра.*

Для доказательства воспользуемся:

Лемма. *Ребра, отмеченные на шаге Борувки, образуют лес.*

Warning! *Авторское доказательство леммы.* Предположим, что какие-то из отмеченных ребер образовали цикл. Ориентируем ребра этого цикла следующим образом. Пусть в шаге 1 для вершины v было помечено ребро (v, u), тогда ориентируем его как $v \to u$.

Утверждается, что получившийся орграф есть цикл в ориентированном смысле (а не, например, поток). Действительно, для каждой вершины v в цикле верно $out(v) = 1$ по смыслу алгоритма и $in(v) + out(v) = 2$ по смыслу цикла, значит и $in(v) = 1$.

Итак, пусть имеем цикл $v_1 \to v_2 \to \ldots \to v_k \to v_1$. Ребро (v_j, v_{j+1}), ориентированное как $v_j \to v_{j+1}$, означает, что оно было выбрано как минимальное среди всех ребер, смежных с v_j, откуда имеем для весов $w(v_j, v_{j+1}) \le w(v_j, v_{j-1})$. Применив это рассуждение для всех вершин в цикле, имеем:

$$w(v_1, v_2) \le w(v_2, v_3) \le \ldots \le w(v_{k-1}, v_k) \le w(v_k, v_1) \le w(v_1, v_2),$$

откуда следует, что у всех ребер цикла одинаковый вес.

Вспомним, что в случае нескольких смежных ребер с минимальным весом алгоритм выбирает ребро с наименьшим номером (см. шаг 1). Обозначим номер ребра через #. Тогда имеем для всех j $\#(v_j, v_{j+1}) > \#(v_j, v_{j-1}$, или:

$$\#(v_1, v_2) > \#(v_2, v_3) > \ldots < \#(v_{k-1}, v_k) > \#(v_k, v_1) > \#(v_1, v_2),$$

откуда и получаем противоречие. □

Warning! *Авторское доказательство теоремы.* Шаг Борувки в графе G построил лес F, каждому дереву которого соответствует вершина в графе G'. Миностов T' графа G' соединяет все вершины графа G', т.е. все деревья леса F в G, поэтому объединение F и T' есть дерево. По *лемме о безопасных ребрах* все ребра этого дерева входят в какой-то миностов G, ну значит этот миностов и есть $F \cup T'$. □

Теперь несложно получить оценку на время работы.

Лемма. *Время работы шага Борувки есть* $O(E + V)$.

Доказательство. Шаг 1 требует однократного просмотра всех смежных ребер у каждой вершины: $O(E + V)$.

Шаг 2 можно выполнить поиском в глубину, который работает за $O(E + V)$.

Шаг 3 требует переназначения вершин в новые компоненты связности – $O(V)$ – и перераспределения всех ребер на новые вершины – $O(E)$.

Шаг 4 требует просмотра всех ребер – $O(E)$. □

В связном графе $V \leq E + 1 = O(E)$, так что верна и оценка $O(E)$.

Заметим, наконец, что всякий шаг Борувки уменьшает число вершин не менее, чем в два раза. Действительно, шаг 1 соединяет каждую вершину с какой-то, значим образуется не более $n/2$ компонент – вершин в новом графе. Отсюда сразу следует, что применить шаг Борувки до построения полного миностова нужно не более $\log_2 V = O(\log V)$ раз. Общая оценка времени работы алгоритма Борувки есть $O(E \log V)$.

1.2 Линейный вероятностный алгоритм для MST

Coming soon