Остовные деревья. Пусть нам дан взвешенный граф. Мы хотим оставить в нем подмножество ребер минимального суммарного веса так, чтобы граф оставался связным.

Лемма о безопасном ребре: Для подмножества вершин A есть ребро наименьшего веса, ведущее в дополнение к A. А именно, ведет из $u \in A$ в $v \notin A$. Существует остов, содержащий это ребро. Доказательство: От противного. Пусть мы не взяли ребро $u \leftrightarrow v$. Посмотрим на путь из u в v в дереве. В какой-то момент там нашлось ребро из A в $V \setminus A$. Тогда можно удалить это ребро и заменить его на $v \leftrightarrow u$ — стоимость не увеличится.

Алгоритм Прима. По аналогии с алгоритмом Дейкстры будем поэтапно строить множество A и добавлять в него минимальное ребро из леммы. Но тут важно, что лемма не гарантирует нам, что итоговый ответ хороший, потому что она говорила нам только о том, что ребро принадлежит какому-то ответу. Но есть усиленная версия леммы — если ребро меньше по весу, чем все остальные, то тогда оно обязательно будет в минимальном остове. Поэтому надо как-то неявно задать веса ребер, чтобы они были различны (это в реальной жизни не нужно, только для доказательства), после чего алгоритм Прима нам уже будет гарантировать минимальный ответ.

Система непересекающихся множеств. Структура данных, которая поддерживает следующие операции:

- $get(v)$ — узнать, в каком множестве находится элемент v
- $union(v, u)$ — объединить множество, содержащее v с множеством, содержащим u.
- $check(v, u)$ — проверить, находятся ли элементы в одном и том же множестве. Выражается через два get-а.

Алгоритм Краскала. В начале упорядочим ребра по весу. Будем поддерживать какое-то множество компонент связности. Если для очередного ребра компоненты вершин различны, то можно сделать $union$ в dsu. Работает за $O(sort(m) + union(n) \cdot n + check(n) \cdot m)$

Алгоритм Борувки. Выделим для каждой вершины минимальное ребро. Теперь добавим все эти ребра в остов, и сожмем граф. Дальше сделаем аналогичную операцию. Повторяем, пока не сойдется до одной компоненты. Каждый раз вершины соединяются в компоненты хотя бы по две вершины. Это значит, что итоговая оценка сложности будет $O(m \log n)$. Заметим, что на плотных графах Борувка работает за $O(m)$ — каждый раз число ребер уменьшается в два раза.

Чтобы объединить две асимптотики, получим:

$$\frac{n^2}{4^k} \leq m$$

$$4^k \geq \frac{n^2}{m}$$

$$k \approx \frac{1}{2} \log \frac{n^2}{m}$$

Значит, через k операций мы получим граф, который можем считать полным (а на нем мы работаем за линию!). Тогда асимптотика это $O(m \log \frac{n^2}{m})$.