Даны s полиномов с n переменными степени 2 над полем $\mathbb{F}_q$. Задача определить существует ли у этой системы решение называется MQS (Multivariable Quadratic Systems) и является NP-трудной. Мы будем решать задачу #MQS, то есть считать количество (на самом деле долю) решений. Эта задача очевидно не проще, поэтому мы будем пытаться решать ее приближенно, то есть мы разрешаем себе ошибаться не более, чем на ε. Мы хотим построить алгоритм, который будет детерминированным (то есть рандома нет, это важно!), и будет работать за время $poly(n, s, \log q)/\varepsilon^2$.

Теорема 1. *При $s = 1$ задача решается точно за полиномиальное время.*

Эта теорема без доказательства, мы ей только пользовались.

Определение 1. *Множество $S \subset \mathbb{F}_q^n$ называется ε-biased (ε-скошенным, наверное), если выполняется следующее условие*

$$\forall u \in \mathbb{F}_q^n, r \in \mathbb{F}_q \; |Pr_{v \in S}\{\langle u, v\rangle = r\} - Pr_{v \in \mathbb{F}_q^n}\{\langle u, v\rangle = r\}| \leq \varepsilon$$

Теорема 2. *Можно построить ε-biased множество размера $O(\frac{n^2}{\varepsilon^2})$ за время $poly(n, \log q)/\varepsilon^2$*

Эта теорема тоже без доказательства.

Теорема 3. *Можно получить ε-приближение для задачи #MQS за время $poly(n, s, \log q)/\varepsilon^2$*

Для доказательства нужно построить ε-biased подмножество множества $\mathbb{F}_q^s$ (внимание, здесь s!) и рассмотреть формулы $P_i(y) := \sum v_i p_i(y)$, где v_i - элементы построенного множества, а p_i - полиномы из задачи. Тогда если y - решение, то $P_i(y) = 0$, а иначе $P_i(y)$ принимает значения 0 и 1 с вероятностью примерно (с точностью до ε) $\frac{1}{q}$. Тогда приближением количества решений #MQS будет разность количеств решений уравнений $P_i(y) = 0$ и $P_i(y) = 1$. Эти количества мы считать умеем по теореме 1.

Следствие 1. *Если $\varepsilon > \frac{1}{q^n}$ и система имеет хотя бы εq^n решений, то за время $poly(n, s, \log q)/\varepsilon^2$ можно найти решение явно.*

Значение переменных выбираем, подставив все возможные значения и выбрав то, которое дает большее число решений, а когда мы уже больше не сможем сделать следующий шаг так, чтобы гарантировать, что решения остались, просто переберем значения всех оставшихся переменных.

Теорема 4. *Можно свести #k-SAT с n переменными к вычислению количества решений полинома степени $q(k/\varepsilon)^{O(k)}$ с n переменными за время $O(q^{\varepsilon n})$.*

Нам хочется попросить, чтобы в формуле было не более $(k/\varepsilon)^{O(k)}n$ клозов. Это сделает sparsification lemma (на самом деле нет, она сделает нечто другое, но нам сейчас это не важно). Теперь КНФ достаточно просто сводится к системе полиномов: or можно симулировать через умножение, а and - через and (нам же нужно, чтобы все полиномы выполнились). Также, надо дополнительно добавить n полиномов вида $x(1-x)$, которые запишут условие, что наши переменные булевы. Получили систему уравнений G. Разобъем ее на εn систем уравнений размера не более $(k/\varepsilon)^{O(k)}$. Теперь из каждой G_j построим полином P_j.

$$P_j(x) := 1 - \prod_{i=1}^{(k/\varepsilon)^{O(k)}} (1 - (p_i(x))^{q-1})$$

P_j имеет нужную степень, но полинимов много, а нам нужен один. Но мы уже учились с этим бороться в теореме 3. Разница в том, что там мы только приближали, а нам нужно посчитать точно. Но мы знаем одно 0-biased множество (все линейные комбинации), его и возьмем.