Проблемы аукциона Викри

Сергей Николенко *

16 июня 2008 г.

Содержание

1. VCG-based механизмы

1.1. Постановка задачи

Мы уже обсуждали, что аукцион Викри удовлетворяет четырём свойствам хорошего аукциона:

1) Он правдив и рационален.

2) Он эффективен — максимизирует общее счастье.

3) Он работает с любыми ценностями x_i.

4) Он реализуем за полиномиальное время.

*конспектировали А. Мордвинцев, Д. Трофимов, Р. Наумов

Мы уже обсуждали в той же лекции, что VCG не всегда получается полиномиально реализовать. Сейчас мы рассмотрим похожие соображения в более общей ситуации. И рассмотрим другой подход к решению этой проблемы. Собственно, тогда у нас не было никакого подхода: мы просто зафиксировали, что проблема не решается.

Как водится, напомним определения. N агентов, у агента i есть функция v_i, определяющая его тип. Агент i в частном порядке знает v_i. Ценности квазилинейные: если при результате o механизм платит агенту p_i, то общая польза агента

$$u_i = v_i(o) + p_i.$$

Агент хочет максимизировать u_i. Механизм хочет быть эффективным, то есть максимизировать $V(v, o) = \sum_{i=1}^{N} v_i(o)$.

Прямой механизм — это механизм, где:

- спрашивают типы $\mathbf{w} = (w_1, \ldots, w_N)$,
- а потом вычисляют исход $g(\mathbf{w}) \in \mathcal{O}$
- и платежи каждому участнику $p(\mathbf{w}) = (p_1, \ldots, p_N)$.

Здесь, как и раньше, агенты могут врать, т.е. априори $w_i \neq v_i$.

Механизм правдивый, если каждый агент для каждого возможного своего типа v_i и каждого возможного множества ставок других агентов $\mathbf{w}_{-i}$ максимизирует свою u_i, если объявляет свою настоящую функцию ценности v_i. То есть мы будем говорить о правдивости в доминантных стратегиях.

1.2. VCG и VCG-based

Механизм (g, p) является VCG-механизмом, если:

- $g(\mathbf{w})$ максимизирует общую ценность относительно $\mathbf{w}$, т.е. $g(\mathbf{w}) \in argmax_o V(\mathbf{w}, o)$;
- выплаты вычисляются по VCG-формуле:

  $$p_i(\mathbf{w}) = \sum_{j \neq i} w_j(g(\mathbf{w})) + h_i(\mathbf{w}_{-i}),$$

  где h_i — некая (произвольная) функция от $\mathbf{w}_{-i}$.

Беда в том, что зачастую функцию распределения g за разумное время не подсчитать (т.е. не решить задачу оптимизации); примеры тому мы уже видели.

Механизм (g, p) является VCG-механизмом, основанным на функции g, если

- выплаты вычисляются по VCG-формуле:

$$p_i(\mathbf{w}) = \sum_{j \neq i} w_j(g(\mathbf{w})) + h_i(\mathbf{w}_{-i}),$$

  где h_i — некая (произвольная) функция от $\mathbf{w}_{-i}$.

Т.е. мы просто берём некоторую функцию распределения и строим выплаты соответственно. Важное замечание: выплаты здесь не равны выплатам VCG, т.к. мы подставляем не оптимальную функцию, а нашу g.

Получается, что VCG-based механизм задаётся функцией распределения g и функциями $(h_1, \ldots, h_N)$. Тогда, если у агента настоящая ценность v_i и ставки $\mathbf{w} = (w_1, \ldots, w_N)$, то его utility будет равна

$$v_i(g(\mathbf{w})) + p_i(\mathbf{w}) = v_i(g(\mathbf{w})) + \sum_{j \neq i} w_j(g(\mathbf{w})) + h_i(\mathbf{w}_{-i}) = \\ = V((v_i, \mathbf{w}_{-i}), g(\mathbf{w})) + h_i(\mathbf{w}_{-i}).$$

Ценность агента i равна

$$V((v_i, \mathbf{w}_{-i}), g(\mathbf{w})) + h_i(\mathbf{w}_{-i}).$$

Что это значит? Это значит, что VCG-based механизм даёт правдивому агенту в точности сумму общего счастья (с точностью до h_i). То есть если g распределяет оптимально, то VCG-based механизм (который уже просто VCG) автоматически будет правдивым. Но если g не оптимальна, то VCG-based механизм может потерять правдивость.

Приведём пример, когда всё совсем плохо (мы его уже упоминали раньше). Пусть играют три игрока, и функция распределения немножко неоптимальная. А именно — механизм распределяет вещь игроку, ставка которого вторая сверху. Какие тогда будут VCG-выплаты и доминантные стратегии?

VCG-выплаты будут такими: механизм будет брать с того, кому достанется вещь (второго сверху) ставку третьего сверху. А вот доминантных стратегий вообще не будет... Например, агентам зачастую будет выгодно понижать свою ставку, причём в зависимости от ставок других агентов. В общем, получится крайне неэффективный аукцион. А ведь на самом деле мы всего лишь заменили точную функцию распределения на (не такую уж плохую) приближённую.

Можно чуть обобщить VCG механизмы и максимизировать некие аффинные преобразования ценностей. Рассмотрим оценочную функцию v и набор положительных констант $\mathbf{a} = (a_1, \ldots, a_N)$. Тогда можно определить взвешенную функцию общественной выгоды $g_{v,\mathbf{a}}(\mathbf{w}, o)$ исхода o:

$$g_{v,\mathbf{a}}(\mathbf{w}, o) = v(0) + \sum_{i>0} a_i w_i(o).$$

Будем писать просто $g_{\mathbf{a}}$, когда $v = 0$.

Механизм, основанный на аффинном (affine-based) — это функция распределения g и набор положительных констант $\mathbf{a} = (a_1, \ldots, a_N)$. Affine-based механизм устанавливает игроку i выплату

$$p_i(\mathbf{w}) = \frac{1}{a_i} \left(\sum_{j \neq i} a_j w_j(g(\mathbf{w})) + h_i(\mathbf{w}_{-i}) \right),$$

где h_i — некоторые (произвольные) функции.

Affine-based механизм даёт агенту i прибыль

$$\begin{aligned} v_i(g(\mathbf{w}) + p_i(\mathbf{w})) = \frac{1}{a_i}(a_i v_i(g(\mathbf{w}))) + p_i(\mathbf{w}) = \\ = \frac{1}{a_i} \left(g_{\mathbf{a}}((v_i, \mathbf{w}_{-i}), g(\mathbf{w})) + h_i(\mathbf{w}_{-i}) \right). \end{aligned}$$

То есть если g максимизирует $g_{\mathbf{a}}(\mathbf{w}, o)$ (по o), то механизм становится правдивым.

Оказывается, affine-based механизмы — это в каком-то смысле единственные правдивые механизмы. Больше ничего правдиво не реализуешь. Мы этим тоже займёмся, чуть позже.

Мы уже разбирались с тем, что VCG никогда толком не вычислишь. Второй шаг VCG — определение победителя (winner determination, WD). Задача WD для VCG даже с целеустремлёнными агентами NP-трудна. Для этого мы сводили к ней WIS — Weighted Independent Set; см. лекцию 7.

Более того, WIS — задача не просто NP-трудная, а очень NP-трудная. Если NP$\not\subseteq$ZPP, то для всякого $\epsilon > 0$ не существует $O(n^{1-\epsilon})$-приближённого алгоритма для WIS, где n — число вершин в графе. Следствие: для всякого $\epsilon > 0$ не существует $O(M^{\frac{1}{2}-\epsilon})$-оптимального алгоритма для WD, где M — количество вещей. Иначе говоря, даже самый лучший полиномиальный алгоритм сможет распределить не лучше, чем в $O(\sqrt{M})$ раз хуже оптимума. Печально, но ничего не поделаешь.

Оказывается, далеко не всё на свете можно сделать правдивыми VCG. Мы ещё поговорим о теореме Робертса, которая говорит, что в случае неограниченных коэффициентов квазилинейных предпочтений правдиво можно реализовать только аффинные максимизаторы. А сейчас характеризуем класс правдивых VCG-based механизмов для комбинаторных аукционов. У нас получится, что неоптимальные правдивые VCG-based механизмы вообще какие-то ущербные. А, значит, правдивые полиномиальные тоже ущербные, потому что оптимальные NP-трудны.

1.3. Правдивость и максимальность на образе

Рассмотрим функцию распределения g и множество возможных типов $\Theta = \prod_{i=1}^{N} \Theta_i$. Рассмотрим подмножество $\Theta' \subseteq \Theta$ и обозначим $\mathcal{O} = m_{\Theta'} g$, т.е. $\mathcal{O} = \{g(\mathbf{w}) \mid \mathbf{w} \in \Theta'\}$. g максимально на своём образе на Θ', если для каждого типа $\mathbf{w} \in \Theta'$ $g(\mathbf{w})$ максимизирует g на $\mathcal{O}$. g максимально на своём образе, если g максимально на своём образе на Θ.

Например, рассмотрим комбинаторный аукцион, который отдаёт весь набор вещей S агенту, у которого максимально $v_i(S)$. Очевидно, что он, вообще говоря, не эффективен — можно распределить вещи между несколькими агентами и добиться большего. Но *на своём образе* он максимален: нельзя лучше отдать все вещи одному агенту, чем агенту $argmax v_i(S)$. Оказывается, что такие аукционы характеризуют правдивые.

Теорема. *VCG-based механизм правдив (почти) iff его функция распределения максимальна на своём образе.*

Справа налево очевидно. Механизм с правдивой на образе функцией распределения — это просто VCG, если ограничить образом набор возможных исходов. Значит, как и всякий VCG, он правдив.

Наоборот — тут не совсем так, но почти. Обозначим через $\tilde{\Theta}$ множество всех таких типов $\theta = (\theta_1, \ldots, \theta_N)$, что для любых двух различных распределений $x, y \in \mathcal{O}$ $g(\theta, x) \neq g(\theta, y)$ $\forall \theta \in \tilde{\Theta}$. Мы так охватим почти все типы (проверьте, что $\Theta \setminus \tilde{\Theta}$ имеет меру 0).

Мы докажем, что если VCG-based механизм для комбинаторного аукциона правдив, то его функция распределения максимальна на своём образе на $\tilde{\Theta}$. Предположим противное: пусть (g, p) правдив, но g не максимальна на своём образе на $\tilde{\Theta}$.

Функции h_i не влияют на правдивость — положим их равными нулю. Т.е. для всех i $p_i(\mathbf{w}) = \sum_{j \neq i} w_j(g(\mathbf{w}))$. Это значит, что полезность для каждого агента равна

$$v_i(g(\mathbf{w})) + \sum_{j \neq i} w_j(g(\mathbf{w})) = g((v_i, \mathbf{w}_{-i}), g(\mathbf{w})).$$

Обозначим через $\mathcal{O}$ образ g на $\tilde{\Theta}$; пусть для $\theta \in \tilde{\Theta}$ g не оптимальна, т.е. $y = argmax_{o \in \mathcal{O}} V(\theta, o) \neq g(\theta)$. По определению $\tilde{\Theta}$ y единственный. Обозначим также $\mathbf{w} \in \tilde{\Theta}$ тип, для которого $y = g(\mathbf{w})$. Он существует, т.к. $y \in \mathcal{O}$.

Теперь будем строить новый вектор типов $\mathbf{z}$.

$$z_i(s) = \begin{cases} v_i(s), & s \not\supset y_i, \\ \infty, & s \supset y_i. \end{cases}$$

То есть новый агент i очень хочет y_i, а в остальном совпадает с v_i. Будем предполагать, что $z \in \tilde{\Theta}$ (иначе добавим маленький шум).

Мы сначала докажем, что на z алгоритм выдаёт y. А затем докажем, что если он выдаёт y на z, он должен выдавать y и на θ.

Рассмотрим последовательность векторов

$$\begin{array}{rcl} \mathbf{w}^0 & = & (\theta_1, \theta_2, \ldots, \theta_N), \\ \mathbf{w}^1 & = & (z_1, \theta_2, \ldots, \theta_N), \\ \ldots & & \ldots, \\ \mathbf{w}^N & = & (z_1, \ldots, z_N). \end{array}$$

То есть вектор по одной компоненте превращается из θ в $\mathbf{z}$.

Докажем, что $g(\mathbf{w}^1) = y$. Пусть не так, т.е. по определению $\tilde{\Theta}$ $V(\mathbf{w}^1, g(\mathbf{w}^1)) \neq V(\mathbf{w}^1, y)$. Рассмотрим ситуацию $(z_1, \theta_2, \ldots, \theta_N)$. Агент 1 может объявить θ_1 и заставить механизм выбрать y. Механизм правдив, поэтому $V(\mathbf{w}^1, g(\mathbf{w}^1)) > V(\mathbf{w}^1, y)$.

Должно быть верно, что $g_1(\mathbf{w}^1) \supseteq y_1$, т.к. агент 1 должен получить всё то, что он получает в случае объявления $\mathbf{w}^1$ (там у него ∞). Поэтому верно, что $\infty + \sum_{j=2}^N w_j(g(\mathbf{w}^1)) > \infty + \sum_{j=2}^N w_j(\mathbf{y})$. Но $\theta_1(g(\mathbf{w}^1)) \geq \theta_1(y)$, т.к. free disposal.

Значит, $w_1(g(\mathbf{w}^1)) \sum_{j=2}^N w_j(g(\mathbf{w}_1)) > w_1(y) + \sum_{j=2}^N w_j(\mathbf{y})$. Значит, $V(\mathbf{w}^0, g(\mathbf{w}^1)) > V(\mathbf{w}^0, y)$. А значит, для типа первого агента θ_1 ему лучше объявить z_1. Но механизм правдив. Противоречие.

Аналогично можно продолжать и показать шаг за шагом, что $g(z) = y$. Доказали первый шаг. Теперь покажем, что из $g(z) = y$ следует, что $g(v) = y$, а это уже будет противоречие.

Рассмотрим последовательность векторов

$$\begin{array}{rcl} \mathbf{v}^0 & = & (v_1, v_2, \ldots, v_N), \\ \mathbf{v}^1 & = & (z_1, v_2, \ldots, v_N), \\ \ldots & & \ldots, \\ \mathbf{v}^N & = & (z_1, \ldots, z_N). \end{array}$$

То есть вектор по одной компоненте превращается из $\mathbf{v}$ в $\mathbf{z}$.

Докажем, что для всех $\mathbf{v}^j$ y максимизирует V на $\mathcal{O}$. Тогда мы сможем спуститься от $\mathbf{v}^N$ к $\mathbf{v}^0$, доказывая, что они все равны (а иначе механизм будет неправдивый, как раньше).

Итак, почему же максимизируется V? Пусть V для v^1 максимизируется на $x \neq y$. Тогда, т.к. ∞ большое, $x_1 \supseteq y_i$, и ценность для первого агента равна ∞.

Но y максимизирует V на $\mathcal{O}$ для v^0. То есть для всех $x \neq y$

$$v_1(y) + \sum_{j=2}^{N} v_j(y) > v_1(x) + \sum_{j=2}^{N} v_j(x),$$

$$V(v^1, y) = \infty + \sum_{j=2}^{N} v_j(y) > \infty + \sum_{j=2}^{N} v_j(x) = V(v^1, x).$$

Противоречие.

Рассмотрим VCG-based механизм для комбинаторного аукциона с функцией распределения g. Если механизм правдив, то существует такая функция распределения $\tilde{g}$, что она максимальна на своём образе и для каждого v $V(v, g(v)) = V(v, \tilde{g}(v))$. Для доказательства рассмотрим механизм, который оптимален на образе g и эффективен. Тогда полученный VCG будет, по теореме, правдив. Но полезность для каждого агента определяется общим счастьем, т.е. $V(v, g(v))$ и $V(v, \tilde{g}(v))$ непрерывны и совпадают на плотном подмножестве.

2. Механизмы неразумные и вырожденные

2.1. Разумные и неразумные механизмы

Механизм для комбинаторного аукциона называется *разумным* (reasonable), если тогда, когда существуют вещь i и агент j, удовлетворяющие:

- для всех S, если $j \notin S$, то $v_i(S \cup \{j\}) > v_i(S)$, и

- для всех $k \neq j$ и всех S $v_k(S \cup \{j\}) = v_k(S)$,

вещь i достаётся агенту j. То есть если агенту j вещь всегда нужна, и больше никому она не нужна вообще, то j получит эту вещь. Кажется очень разумным.

Теорема. *Любой неоптимальный правдивый VCG-based механизм для комбинаторных аукционов не является разумным.*

Рассмотрим механизм M. Мы тут выясняли, что существует эквивалентный механизм $\tilde{M}$, который оптимален на своём образе. Но тогда $\tilde{M}$ тоже будет неоптимальным. Значит, у него образ не полный.

Значит, существует разбиение $\mathbf{S} = (S_1, \ldots, S_N)$, не лежащее в образе механизма. Определим вектор типов

$$v_i(X) = \begin{cases} 1, & X \supset S_i, \\ 0, & \text{в противном случае.} \end{cases}$$

Каждый чего-то хочет, и все хотят разного.

Но $\mathbf{S}$ не в образе, и, значит, $\tilde{g}(v) \neq \mathbf{S}$. Поскольку S строго оптимально, $g(v)$ должно быть неоптимальным. Значит, кто-то не получит своё S_j. Но каждая вещь при таком разбиении нужна только одному. Значит, механизм неразумный.

Следствие: если P≠NP, любой полиномиальный правдивый VCG-based механизм для комбинаторных аукционов неразумен. Это потому, что поиск оптимального разбиения NP-труден.

2.2. Multicast transmissions

Рассмотрим задачу: граф $G = (V, E)$, каждое ребро e принадлежит кому-то, цена пересылки t_e известна только владельцу. По данному истоку $s \in V$ и набору терминалов $T \subseteq V$, механизм должен выбрать поддерево с корнем в s, захватывающее все терминалы. Ни один агент не владеет целым сечением графа.

Цель — минимизировать $\sum_{e \in R} t_e$. Цель агента — максимизировать свой доход: $p_i - \sum_{e \in R, e \text{ принадлежит } i} t_e$. Получается задача дизайна механизмов — кстати, небесполезная (для peer-to-peer, например).

CMAP (cost minimization allocation problem) состоит из: *Множество типов* агента i — векторы $(v_i^1, \ldots, v_i^{m_i})$ (у нас $v_i^e = -t_e$). $m = \sum_i m_i$. *Множество исходов* — вектор битов $\mathbf{x} = (x_1^1, \ldots, x_1^{m_1}, \ldots, x_N^{m_N}) \in \{0,1\}^m$. Обозначим $x_i = (x_i^1, \ldots, x_i^{m_i})$. Могут быть дополнительные ограничения (это дерево).

Должны выполняться следующие условия: *Неограниченные стоимости*: если $v_i = (v_i^1, \ldots, v_i^{m_i})$ — тип агента i, и $w_i \leq v_i$ покомпонентно, то w_i

тоже тип агента i. *Независимость и монотонность*: каждая v_i зависит только от x_i; если $w_i^j \leq v_i^j$ для всех j, то для всех x $w_i(x_i) \leq v_i(x_i)$.

Forcing condition: для каждого типа v, исхода x и $\alpha \in \mathbb{R}$ можно определить тип

$$v[\alpha]_i^j = \begin{cases} v_i^j, & x_i^j = 1, \\ \alpha, & \text{в противном случае} \end{cases}.$$

Forcing condition выполняется, если для каждого $y \neq x$

$$\lim_{\alpha \to -\infty} V(t(\alpha), y) = -\infty.$$

То есть надо минимизировать цену при данных ограничениях.

Обозначим $V_{opt}(v)$ оптимальное значение V. Обозначим $V(v, g(v))$ через $V_g(v)$. Алгоритм распределения g вырожденный, если отношение

$$r_g(v) = \frac{V_g(v) - V_{opt}(v)}{|V_{opt}(v)| + 1}$$

неограничено, т.е. для некоторой последовательности v $r_g(v) \to \infty$.

2.3. Вырожденные механизмы

Теорема. *Любой правдивый VCG-based механизм для CMAP либо оптимален, либо вырожден.*

Идея в нашем примере очень простая: выбираем вектор типов, для которых решение неоптимально. Если мы увеличим цену ребра, доход его владельца не увеличится (по правдивости). Мы постепенно увеличим цену всех рёбер, кроме тех, которые в оптимальном решении. В итоге алгоритм выбирает неоптимально, но цена любого субоптимального дерева становится сколь угодно большой.

Теперь поформальнее. Пусть (g, p) — неоптимальный механизм, пусть $p_i(\mathbf{w}) = \sum_{j \neq i} w_j(g(\mathbf{w}))$. Пусть v — вектор, на котором g неоптимальна, пусть y — оптимальный исход.

Определим новый вектор типов z:

$$z_i^j = \begin{cases} v_i^j, & y_i^j = 1, \\ -\alpha, & \text{в противном случае} \end{cases}.$$

Рассмотрим последовательность типов

$$\begin{array}{rcl} \mathbf{v}^0 & = & (v_1, v_2, \ldots, v_N), \\ \mathbf{v}^1 & = & (z_1, v_2, \ldots, v_N), \\ \ldots & & \ldots, \\ \mathbf{v}^N & = & (z_1, \ldots, z_N). \end{array}$$

Докажем, что для всех j $y = \text{opt}(\mathbf{v}^j)$. По определению, y оптимален для $v^{\mathbf{0}}$. Пусть $x \neq y$ — некое распределение. По независимости, $V(v^j, y) = V(v^0, y)$. По монотонности, $V(v^j, x) \leq V(v^0, x)$. Итого $V(v^j, x) \leq V(v^0, x) \leq V(v^0, y) = V(v^j, y)$.

Теперь докажем, что $V(v^1, g(v^1)) < V(v^1, y)$. Пусть не так. Тогда, т.к. y оптимален для v^1, $V(v^1, g(v^1)) = V(v^1, y)$. По независимости, $V(v^0, y) = V(v^1, y)$, а $g(v^0)$ субоптимальна. По монотонности (мы ведь только делаем хуже агенту 1), $V(v^0, g(v^1)) \geq V(v^1, g(v^1))$.

Итого, $V(v^0, g(v^1)) > V(v^0, g(v^0))$. Рассмотрим случай, когда агент 1 имеет тип v^1, остальные v_i. По правдивости, общая полезность равна $V(v^0, g(v^0))$. Но при лжи получается больше. Противоречие.

Аналогично получим, что $V(v^N, g(v^N)) < V(v^N, y) = V(v^0, y)$. По forcing condition, $V(v^N, g(v^N)) \to -\infty$ при $\alpha \to \infty$. Получается, что алгоритм вырожденный.

Значит, если только P$\neq$NP, всякий полиномиальный правдивый VCG-based механизм для NP-трудной CAMP вырожден. Это же, кстати, верно для всякого механизма с доминантными стратегиями (по принципу выявления).

Что же делать? Об этом будет следующая лекция.