11th International Conference "Distributed Computing and Grid Technologies in Science and Education" (GRID'2025)

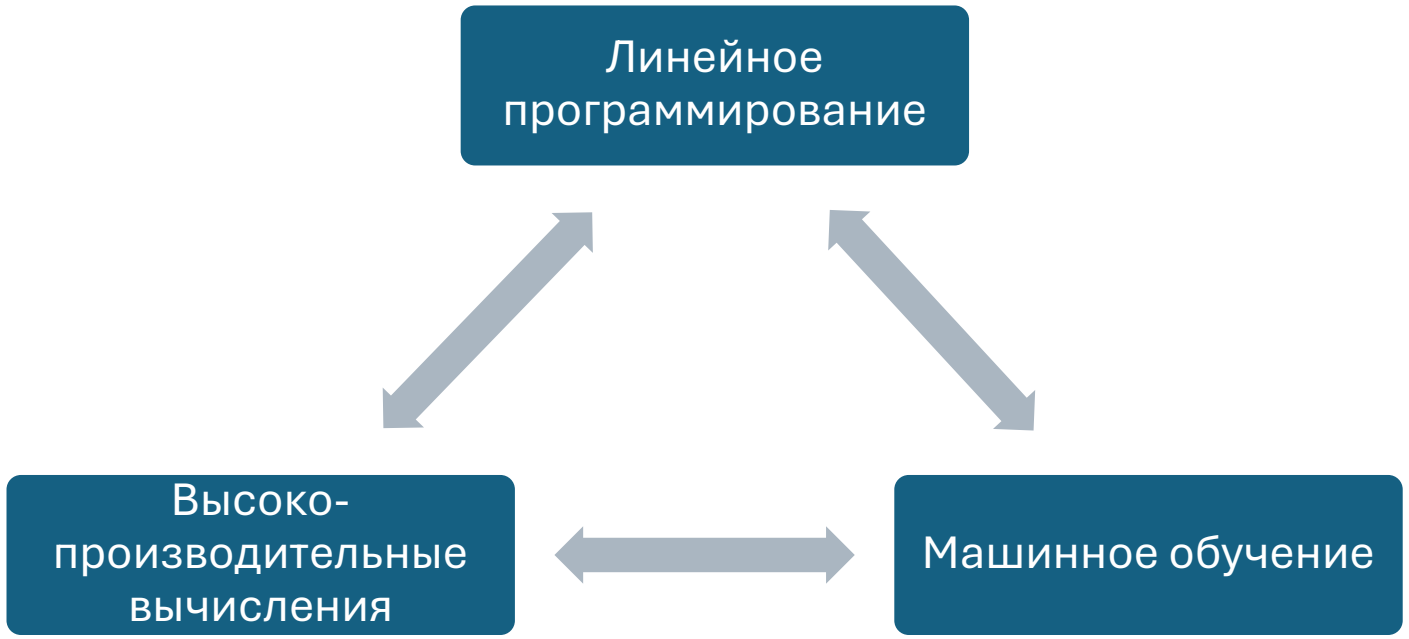
Высокопроизводительные вычисления, нейронные сети и линейное программирование: на пути к гибридному искусственному интеллекту

Соколинский Леонид Борисович
доктор физ.-мат. наук

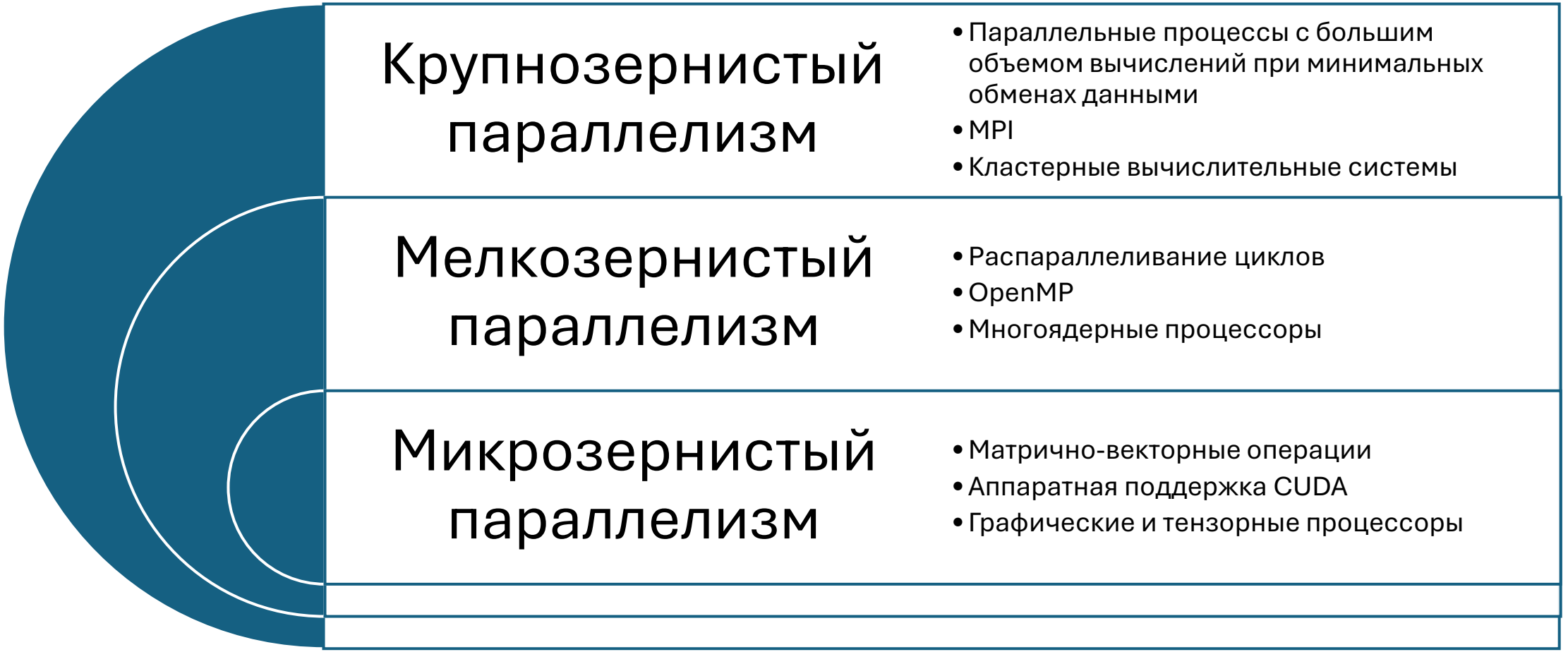
Южно-Уральский государственный университет
(национальный исследовательский университет)

Гибридный искусственный интеллект

- Гибридный ИИ (Hybrid AI): разработка методов объединения сильных сторон машинного обучения, точных математических моделей и высокопроизводительных вычислений



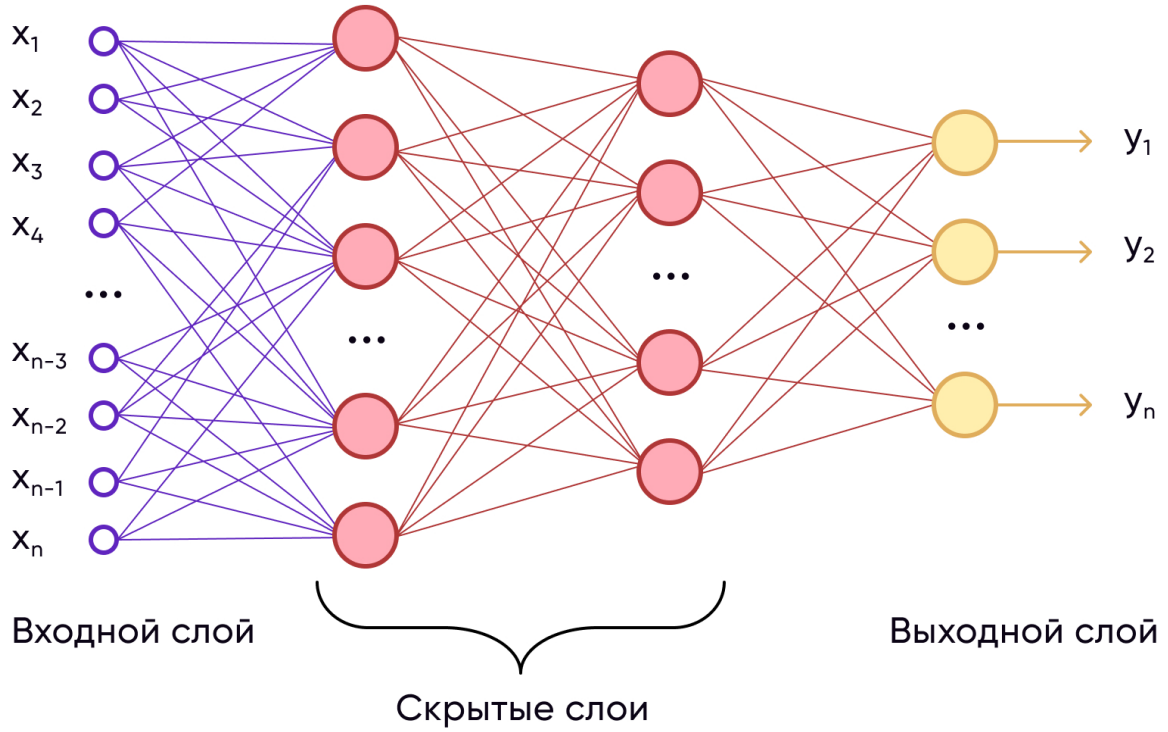
Высокопроизводительные вычисления



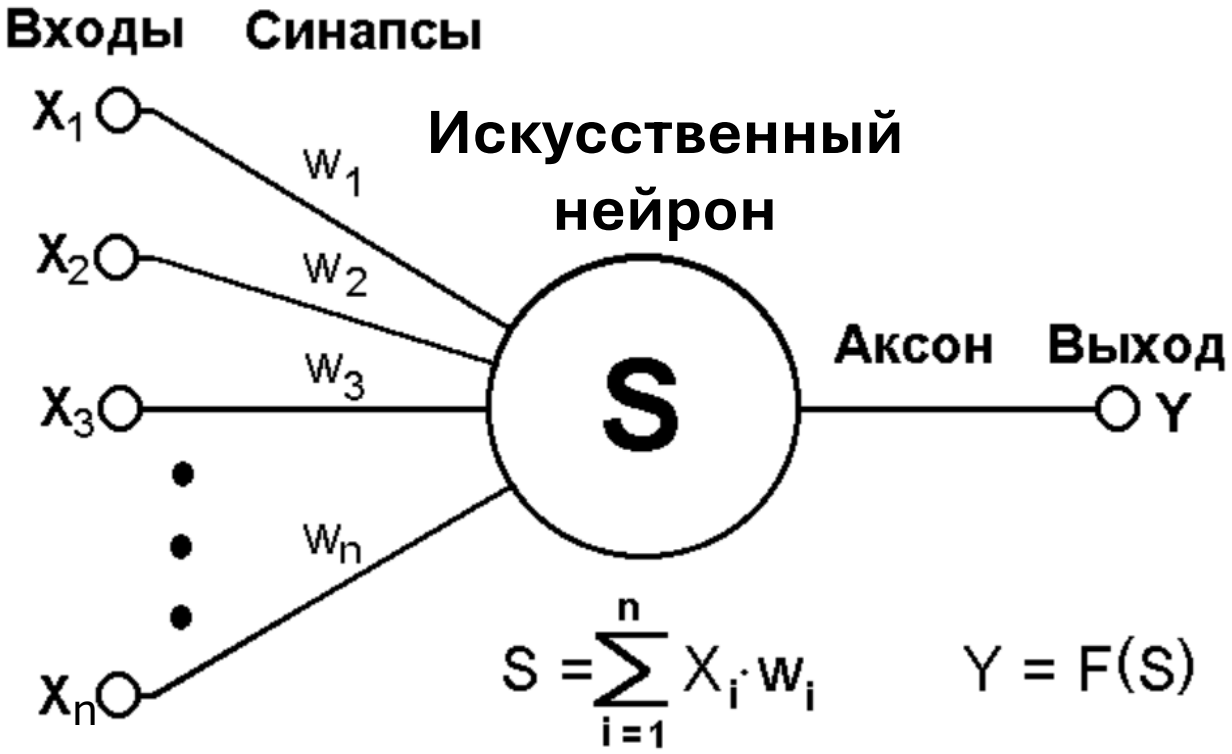
Искусственный интеллект

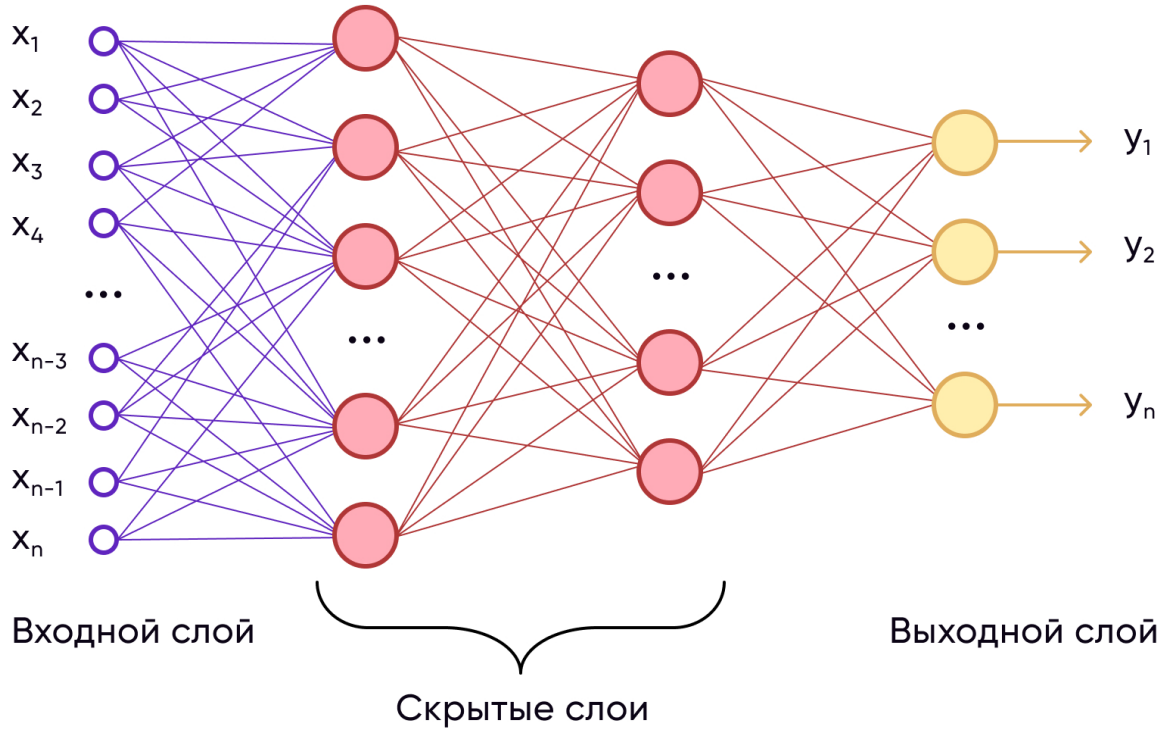
Глубокая нейронная сеть прямого распространения

+ Универсальный инструмент для аппроксимации сложных функций
+ Реальный режим времени
– Необходимы наборы обучающих данных
– Трудности с верификацией



Принцип работы искусственного нейрона



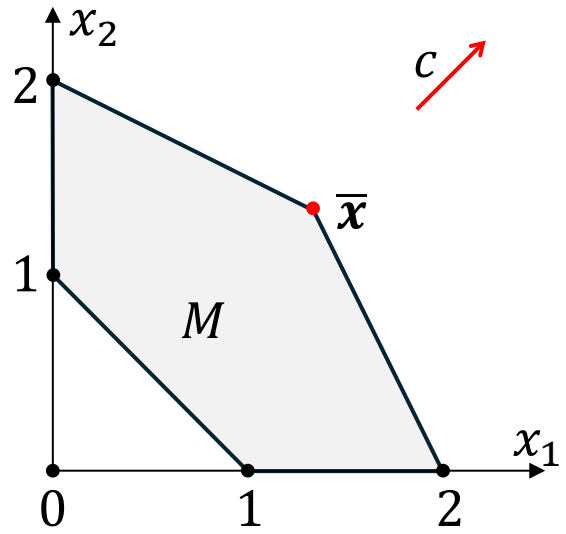


Линейное программирование

- Одна из самых востребованных математических оптимизационных моделей
- Области применения:
  - Экономика
  - Производство
  - Логистика
  - Квантовая физика (теорема Белла)
  - Высокопроизводительные вычисления

$$f(x_1, x_2) = x_1 + x_2 \to max$$

$$\begin{cases} x_1 + 2x_2 \leq 2 \\ 2x_1 + x_2 \leq 2 \\ x_1 + x_2 \geq 1 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$



$\boldsymbol{c} = (1; 1)$ – градиент целевой функции

Система линейных ограничений и целевая функция

$$\begin{cases} a_{1,1}x_1 + \cdots + a_{1,n}x_n \le b_1 \\ \cdots \ \cdots \ \cdots \ \cdots \ \cdots \\ a_{m,1}x_m + \cdots + a_{m,n}x_m \le b_m \\ \\ a_{m+1,1}x_1 + \cdots + a_{m+1,n}x_{m+1} = b_{m+1} \\ \cdots \ \cdots \ \cdots \ \cdots \ \cdots \\ a_{m+k,1}x_{m+k} + \cdots + a_{m+k,n}x_{m+k} = b_{m+k} \end{cases} \Rightarrow \begin{cases} \langle \boldsymbol{a}_1, \boldsymbol{x} \rangle \le b_1 \\ \cdots \\ \langle \boldsymbol{a}_m, \boldsymbol{x} \rangle \le b_m \\ \\ \langle \boldsymbol{a}_{m+1}, x \rangle = b_{m+1} \\ \cdots \\ \langle \boldsymbol{a}_{m+k}, \boldsymbol{x} \rangle = b_{m+k} \end{cases} \Rightarrow \begin{cases} \hat{A}\boldsymbol{x} \le \widehat{\boldsymbol{b}} \\ \bar{A}\boldsymbol{x} = \overline{\boldsymbol{b}} \end{cases}$$

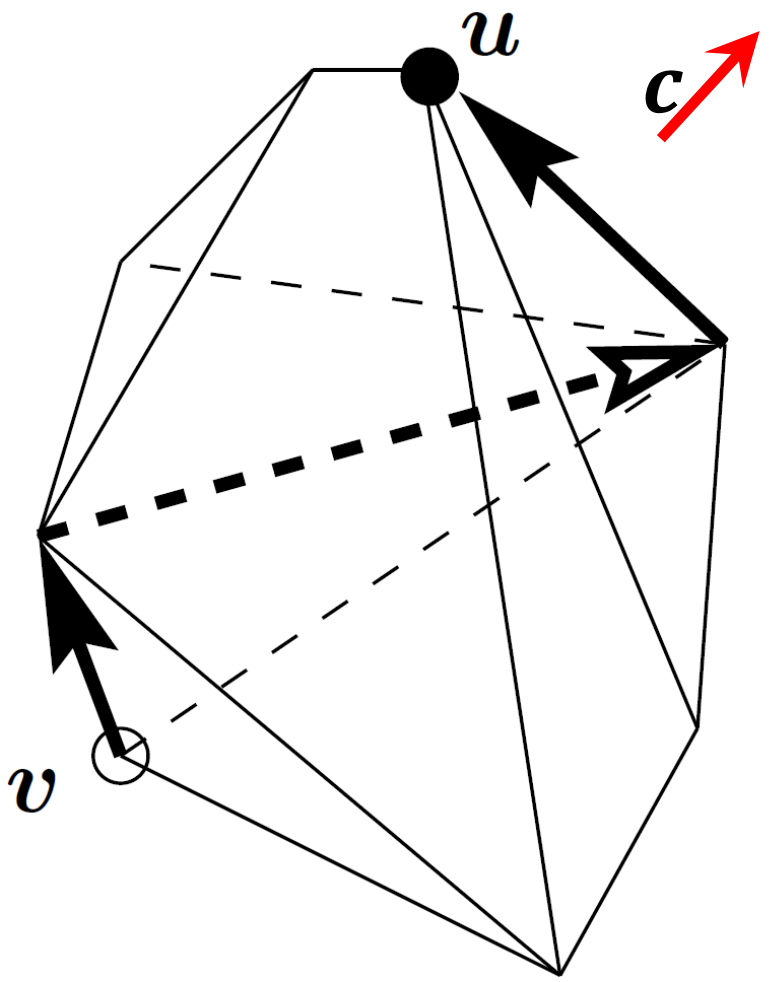
$$f(x_1, \dots, x_n) = c_1x_1 + \cdots + c_nx_n$$

$$f(x) = \langle \boldsymbol{c}, \boldsymbol{x} \rangle$$

$$f(x) = \langle \boldsymbol{c}, \boldsymbol{x} \rangle$$

Симплекс-метод

- Основной метод решения задач ЛП на практике
  - Экспоненциальная сложность
  - Потеря точности
  - Ограниченная масштабируемость (16-32 процессорных узла)



Геометрический подход с использованием HPC и искусственной нейронной сети

Неравенства порождают полупространства

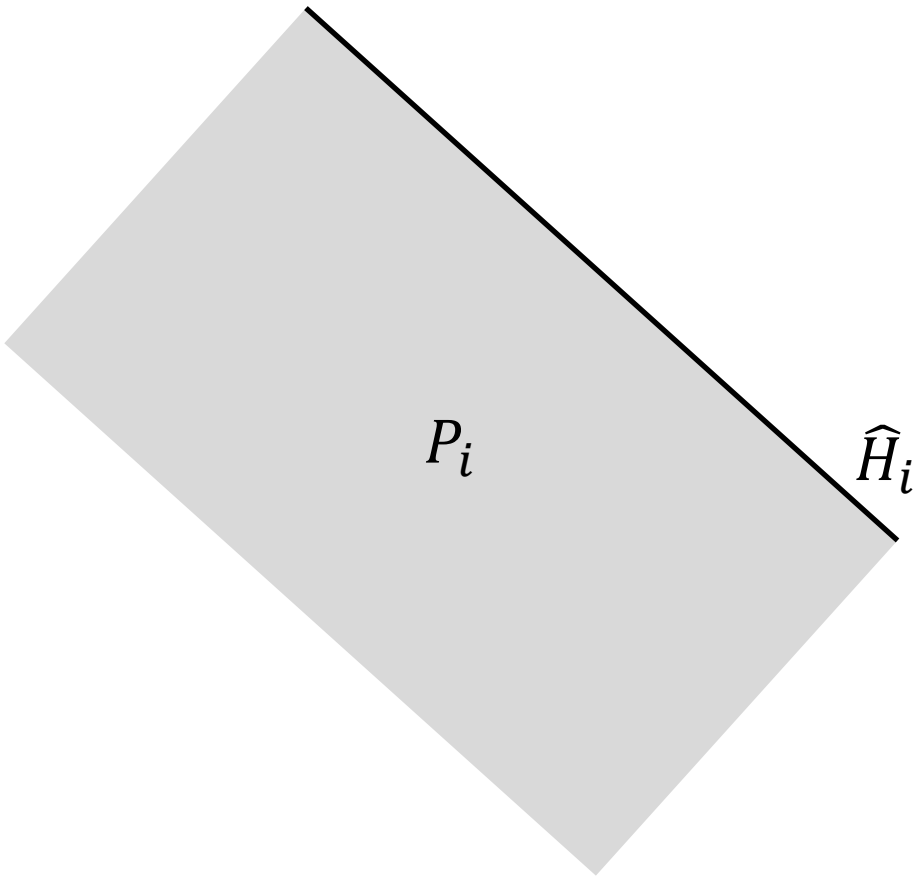
Полупространство:

$$P_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \leq b_i\}$$

Граничная гиперплоскость:

$$\widehat{H}_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle = b_i\}$$

$$i = 1, \dots, m$$

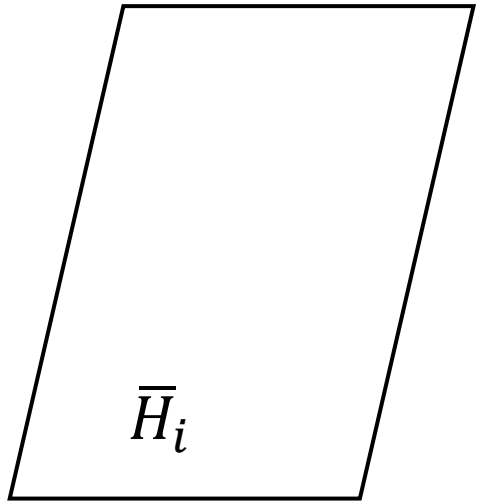


Уравнения порождают опорные гиперплоскости

Опорная гиперплоскость:

$$\overline{H}_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle = b_i\}$$

$$i = m + 1, \ldots, m + k$$



Допустимый многогранник

Граничный многогранник:

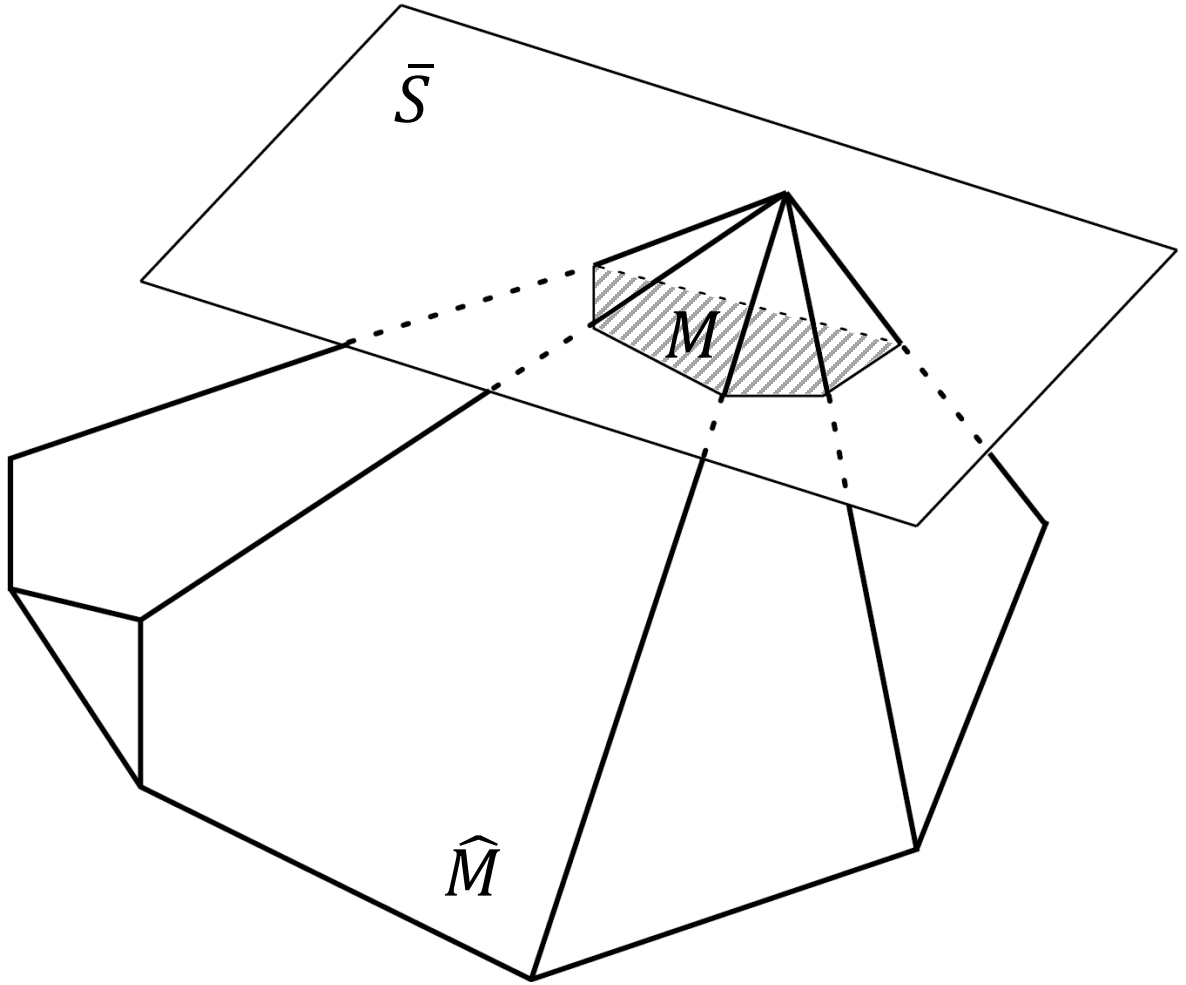
$$\widehat{M} = \bigcap_{i=1}^{m} P_i$$

Опорное полупространство:

$$\bar{S} = \bigcap_{i=m+1}^{m+k} \bar{H}_i$$

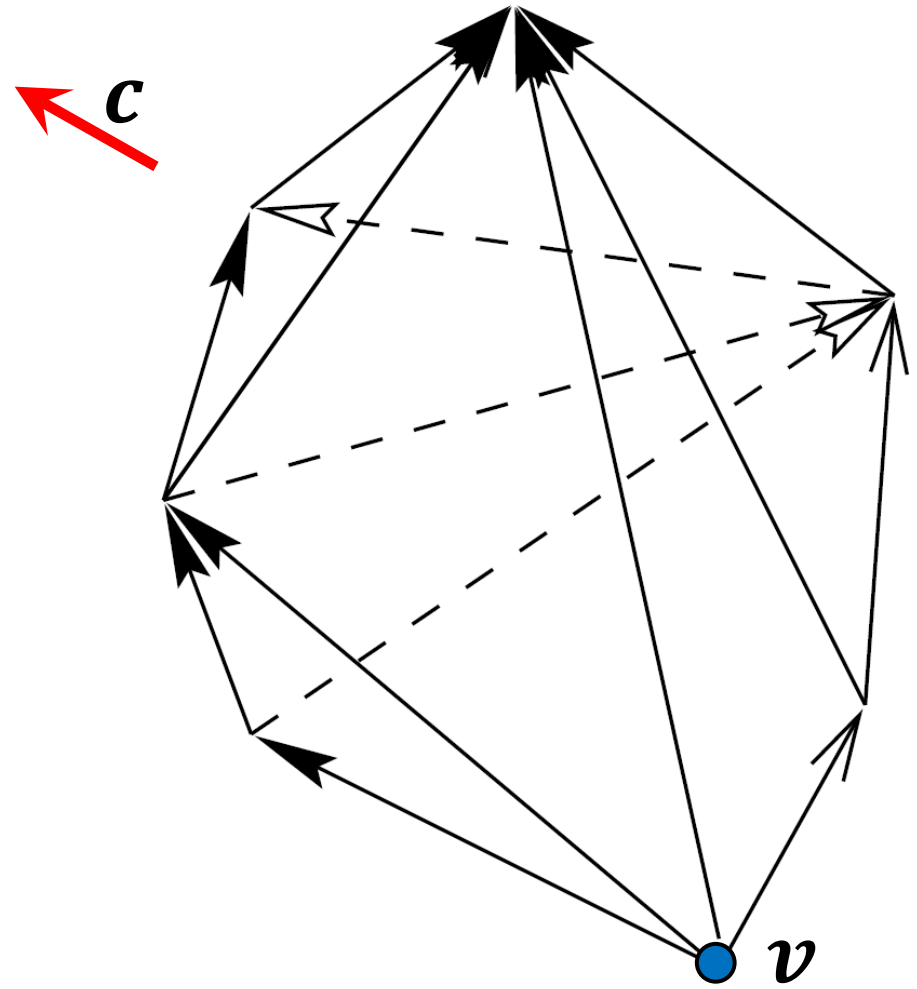
Допустимый многогранник:

$$M = \widehat{M} \cap \bar{S}$$



Идея метода AlEM (Along Edges Movement)

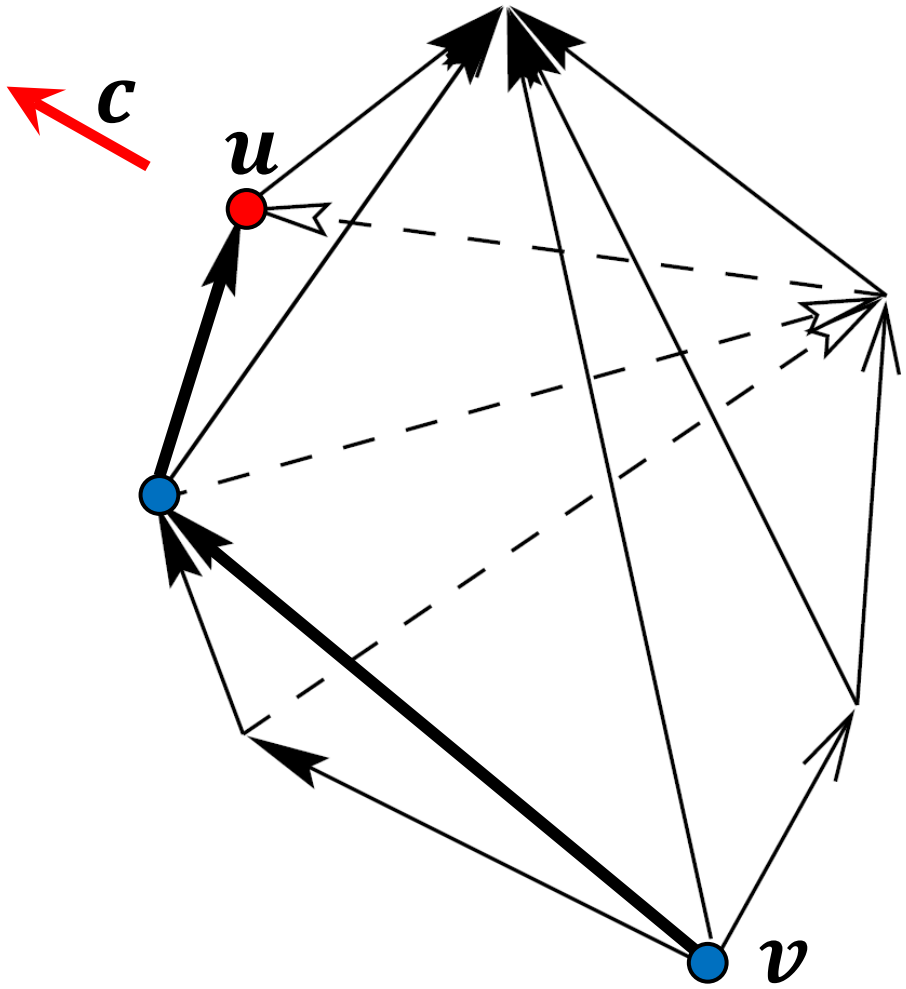
1. Находясь в вершине v, выбираем ребро, ведущее к увеличению целевой функции
2. Выполняем переход к следующей вершине
3. Останавливаемся в вершине, у которой нет ребер, ведущих к увеличению целевой функции



Идея метода AlEM (Along Edges Movement)

1. Находясь в вершине, выбираем ребро, ведущее к увеличению целевой функции
2. Выполняем переход к следующей вершине
3. Останавливаемся в вершине, у которой нет ребер, ведущих к увеличению целевой функции

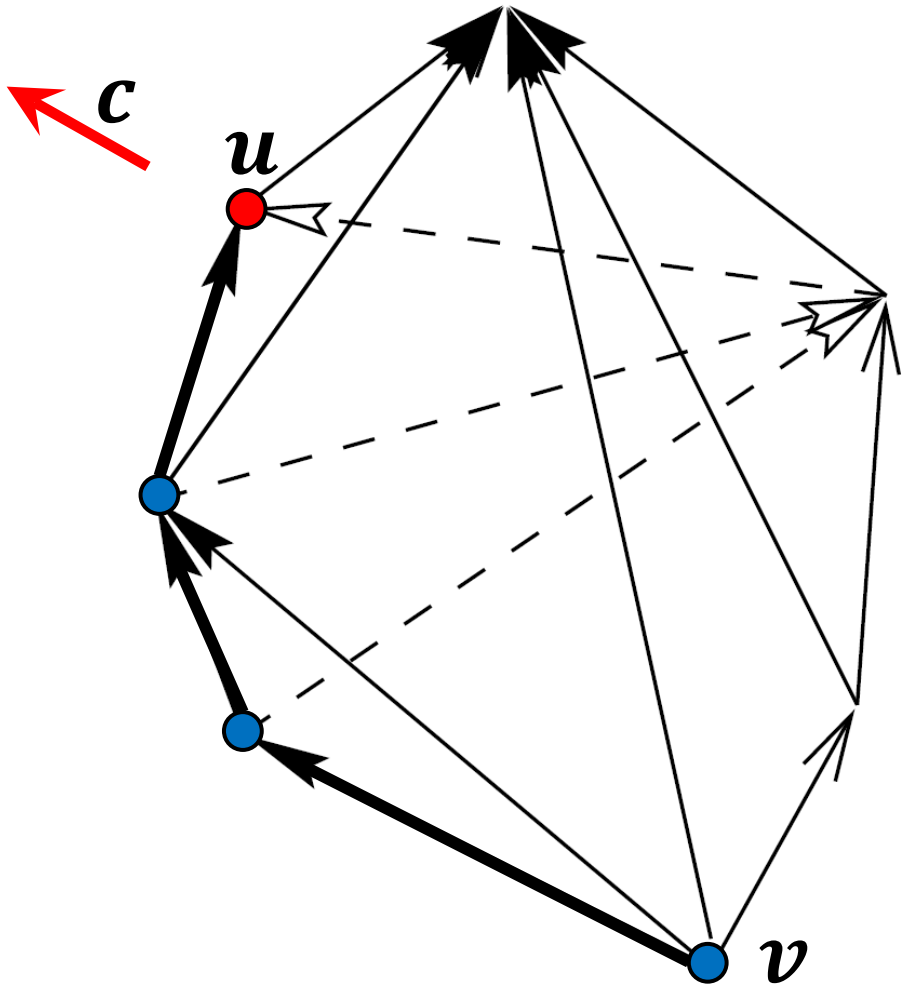
- Стратегии выбора ребра
  - **Максимальная конечная вершина**



Идея метода AlEM (Along Edges Movement)

1. Находясь в вершине, выбираем ребро, ведущее к увеличению целевой функции
2. Выполняем переход к следующей вершине
3. Останавливаемся в вершине, у которой нет ребер, ведущих к увеличению целевой функции

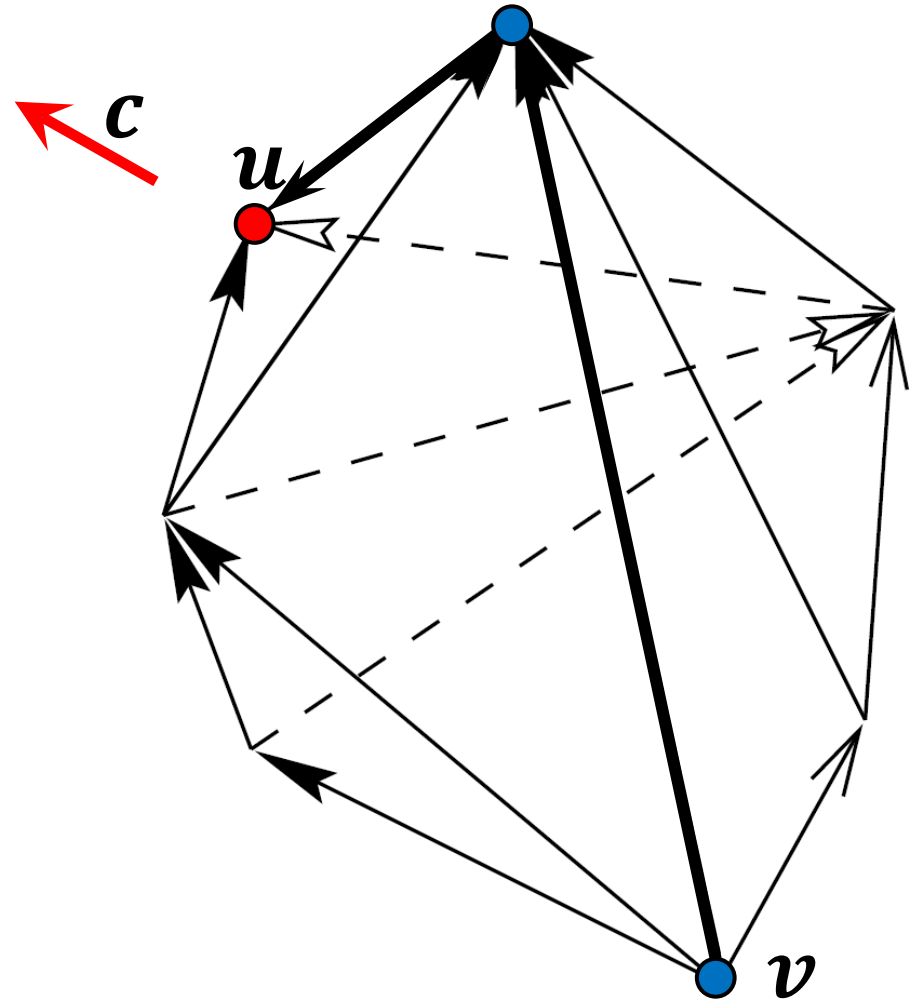
- Стратегии выбора ребра
  - Максимальная конечная вершина
  - **Максимальный градиент**



Идея метода AlEM (Along Edges Movement)

1. Находясь в вершине, выбираем ребро, ведущее к увеличению целевой функции
2. Выполняем переход к следующей вершине
3. Останавливаемся в вершине, у которой нет ребер, ведущих к увеличению целевой функции

- Стратегии выбора ребра
  - Максимальная конечная вершина
  - Максимальный градиент
  - **Случайный / первое попавшееся ребро**



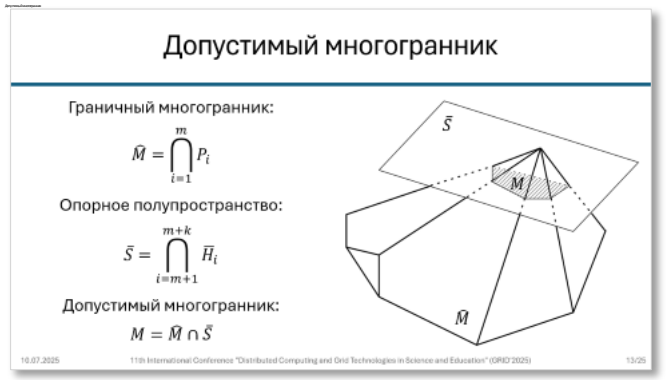
Как получить ребра, исходящие из вершины $\boldsymbol{v}$

Множество номеров граничных гиперплоскостей, проходящих через $\boldsymbol{v}$:

$$I_{\boldsymbol{v}} = \{i \in \{1, \dots, m\} | \langle \boldsymbol{a}_i, \boldsymbol{v} \rangle = b_i\}$$

Подмножество из $n - k - 1$ элементов:

$$\tilde{I}_{\boldsymbol{v}} \subseteq I_{\boldsymbol{v}}, \qquad \left|\tilde{I}_{\boldsymbol{v}}\right| = n - k - 1$$



Прямая, образующая ребро с началом в точке $\boldsymbol{v}$:

$$L_{\boldsymbol{v}} = \bar{S} \cap \bigcap_{i \in \tilde{I}_{\boldsymbol{v}}} \hat{H}_i$$

Перебор всех вариантов можно сделать с помощью алгоритма TWIDDLE*

*Phillip J. Chase. 1970. Algorithm 382: combinations of M out of N objects [G6]. Commun. ACM 13, 6 (June 1970), 368. https://doi.org/10.1145/362384.362502

Необходимость HPC

- Если через точку $\boldsymbol{v}$ проходит k гиперплоскостей, то количество исходящих ребер равно

$$C_k^{n-1} = \frac{k!}{(n-1)!\,(k-n+1)!}$$

- Эксперименты на вычислительном кластере показали, что алгоритмы выбора оптимального ребра обладают практически неограниченной масштабируемостью

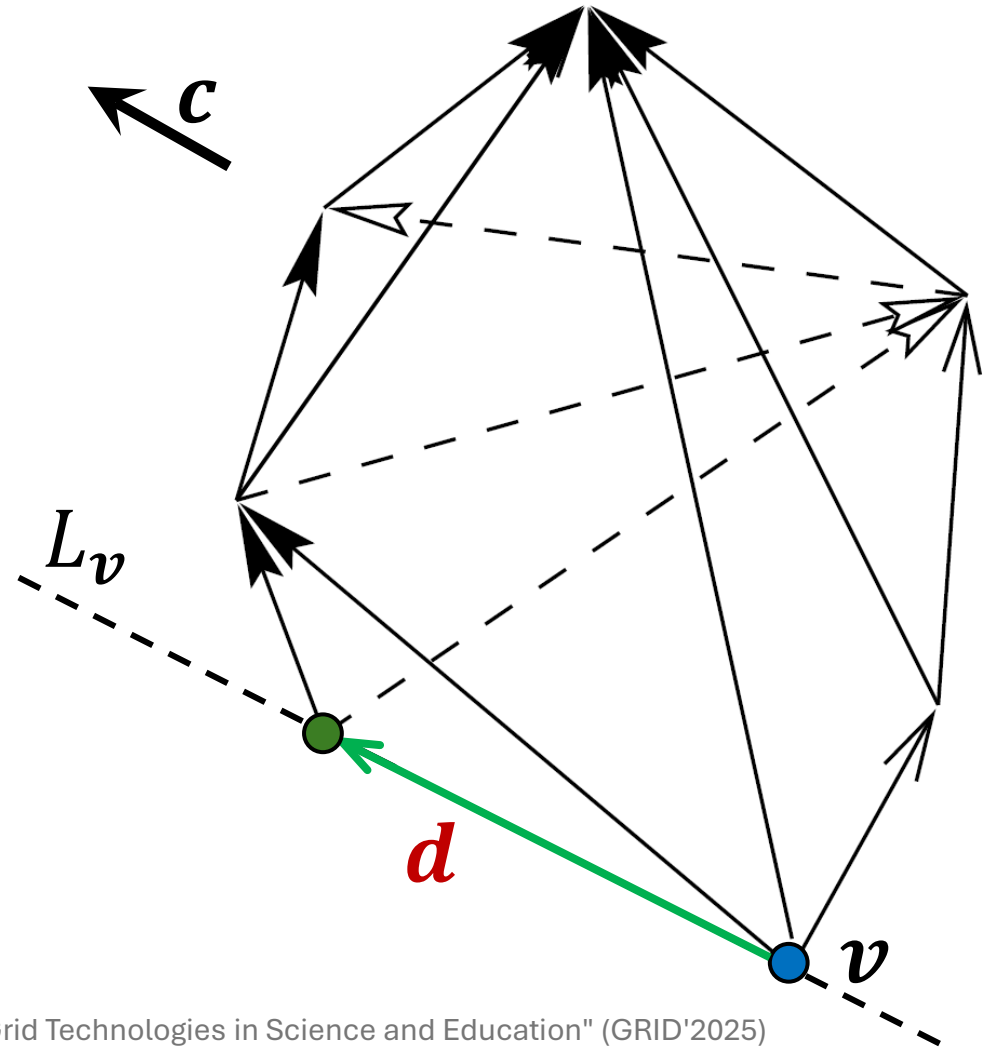
- Чем больше вариантов, тем выше масштабируемость

Для определения конечной точки ребра необходим направляющий вектор $\boldsymbol{d}$

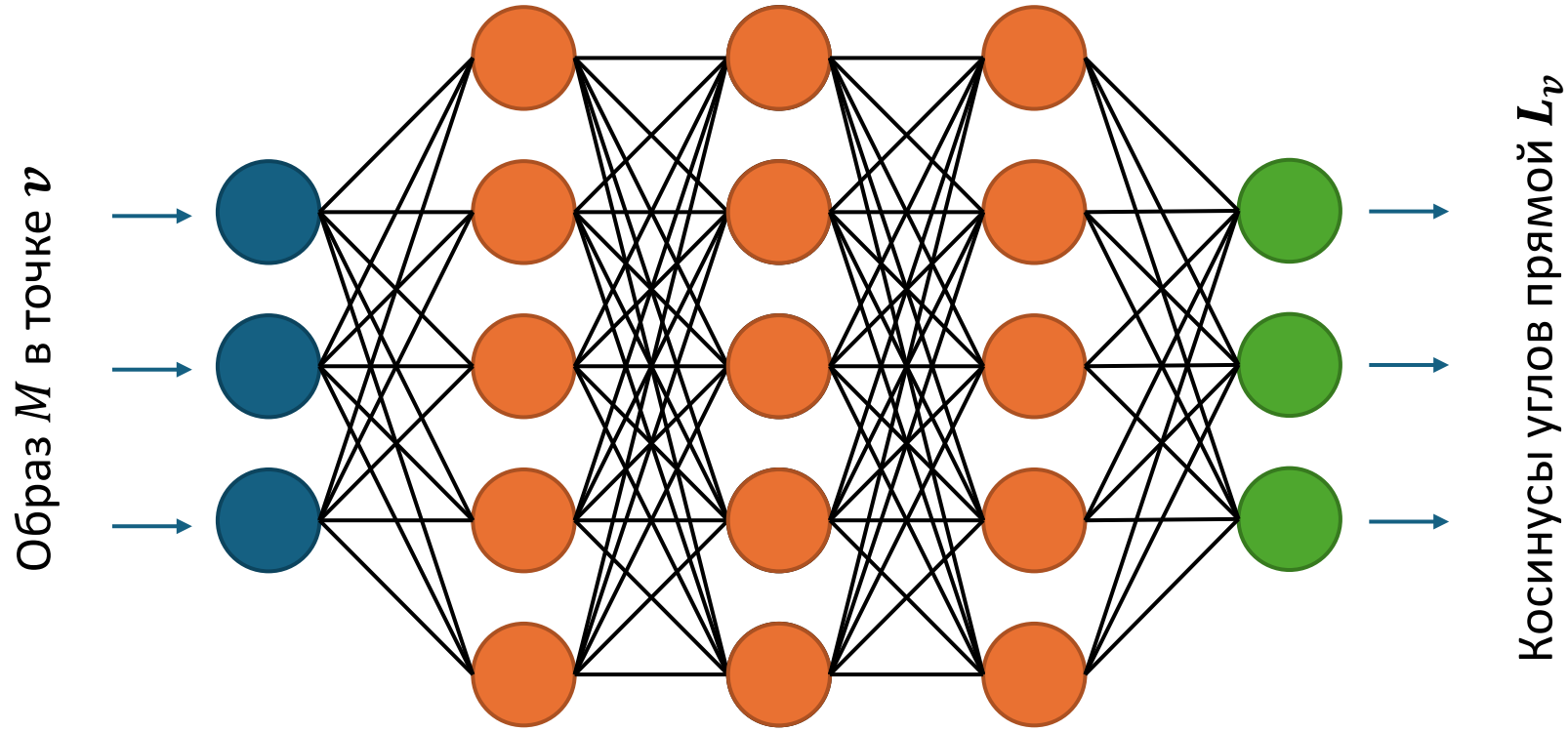
$$L_{\boldsymbol{v}} = \bar{S} \cap \bigcap_{i \in \tilde{I}_{\boldsymbol{v}}} \widehat{H}_i = \left(\bigcap_{i=m+1}^{m+k} \overline{H}_i \right) \cap \left(\bigcap_{i \in \tilde{I}_{\boldsymbol{v}}} \widehat{H}_i \right)$$

$$L_{\boldsymbol{v}} = \{ \boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{x} = \boldsymbol{v} + \lambda \boldsymbol{d}, \lambda \in \mathbb{R} \}$$

$$\boldsymbol{d} = ?$$



Нейросетевой метод вычисления направляющего вектора $\boldsymbol{d}$



Рецептивное поле Q

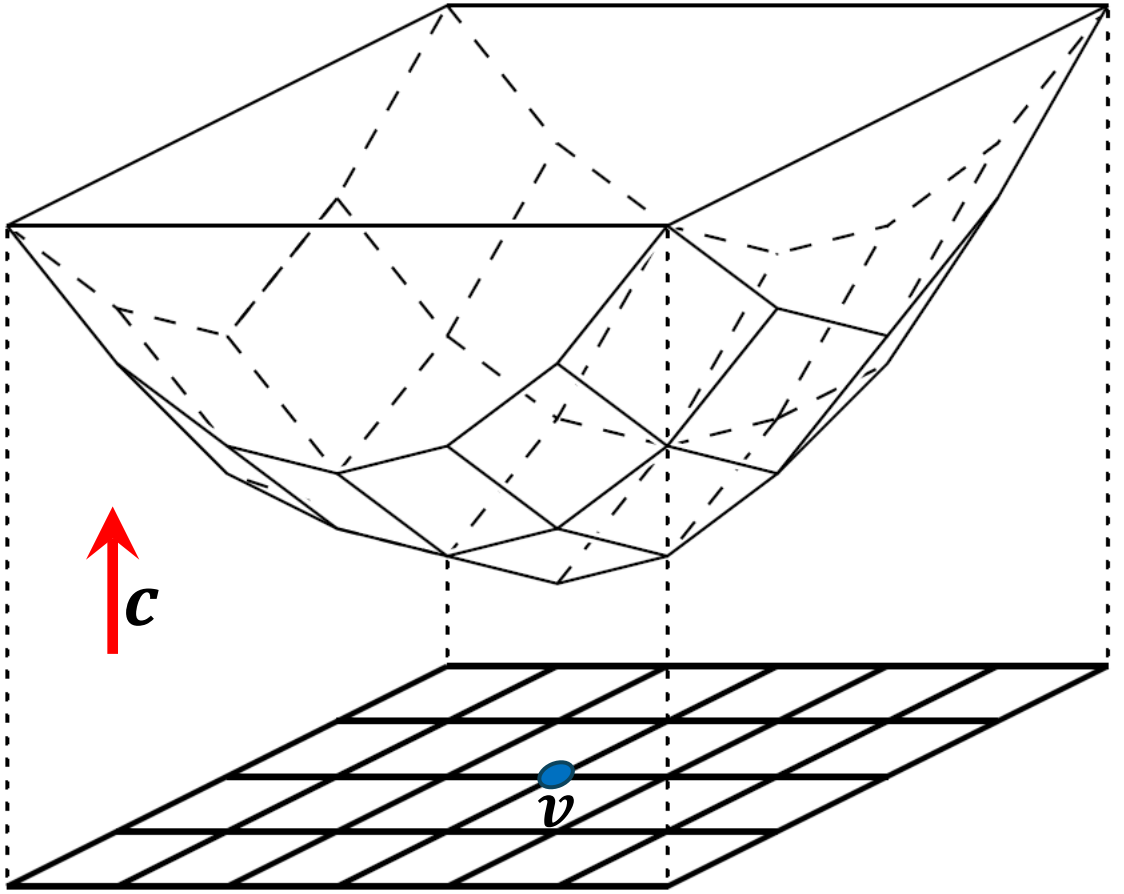
- В целевой гиперплоскости H_c формируем рецептивное Q поле в виде гиперкубической решетки точек

- Каждой точке $\boldsymbol{x}$ рецептивного поля Q сопоставляем смещение относительно M

$$\beta(\boldsymbol{x}) = \min_i \beta_i(\boldsymbol{x})$$

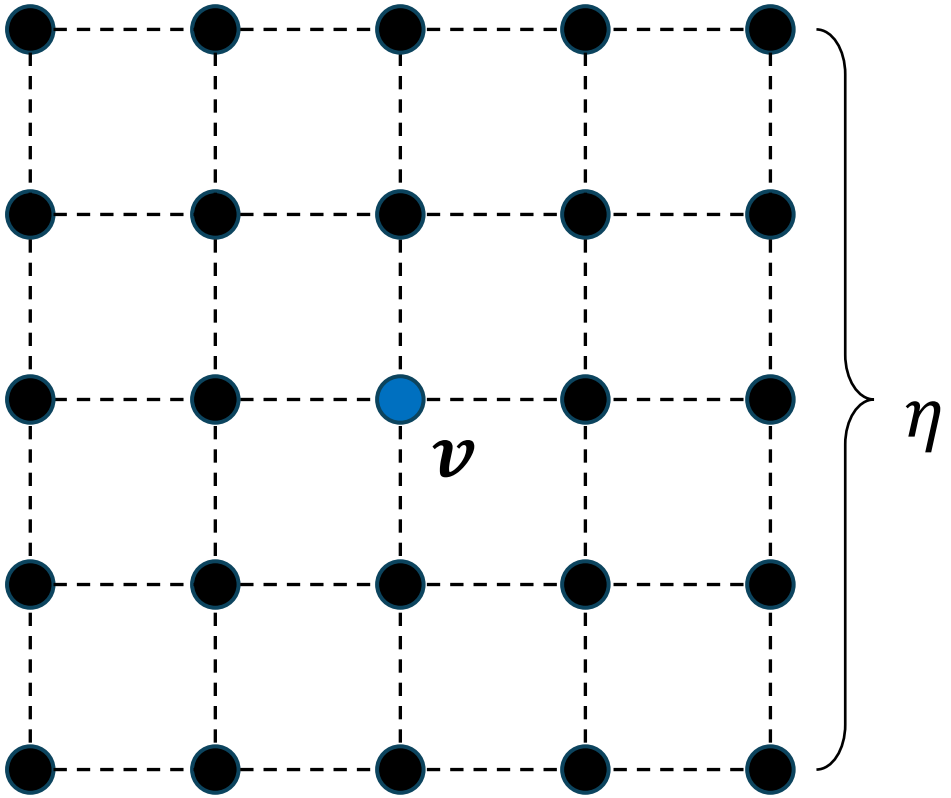
$$\beta_i(\boldsymbol{x}) = -\frac{\langle \boldsymbol{a_i}, \boldsymbol{x} \rangle - b_i}{\langle \boldsymbol{a_i}, \boldsymbol{c} \rangle} \|\boldsymbol{c}\|$$

- Получаем образ G в виде матрицы размерности $n-1$

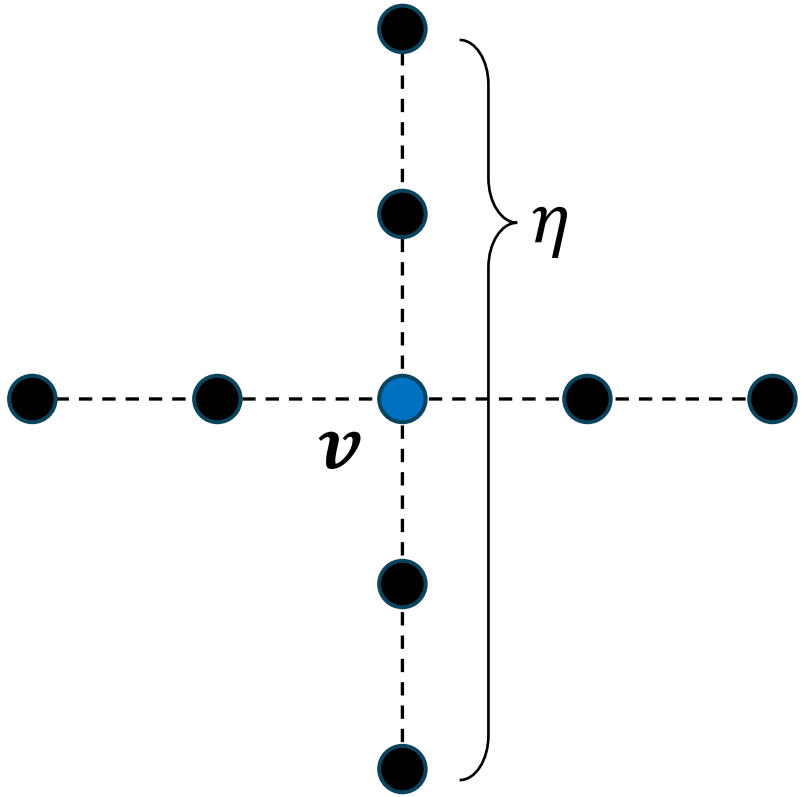


$$H_c = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{c}, \boldsymbol{x} - \boldsymbol{v} \rangle = 0\}$$

Переход к крестообразному рецептивному полю

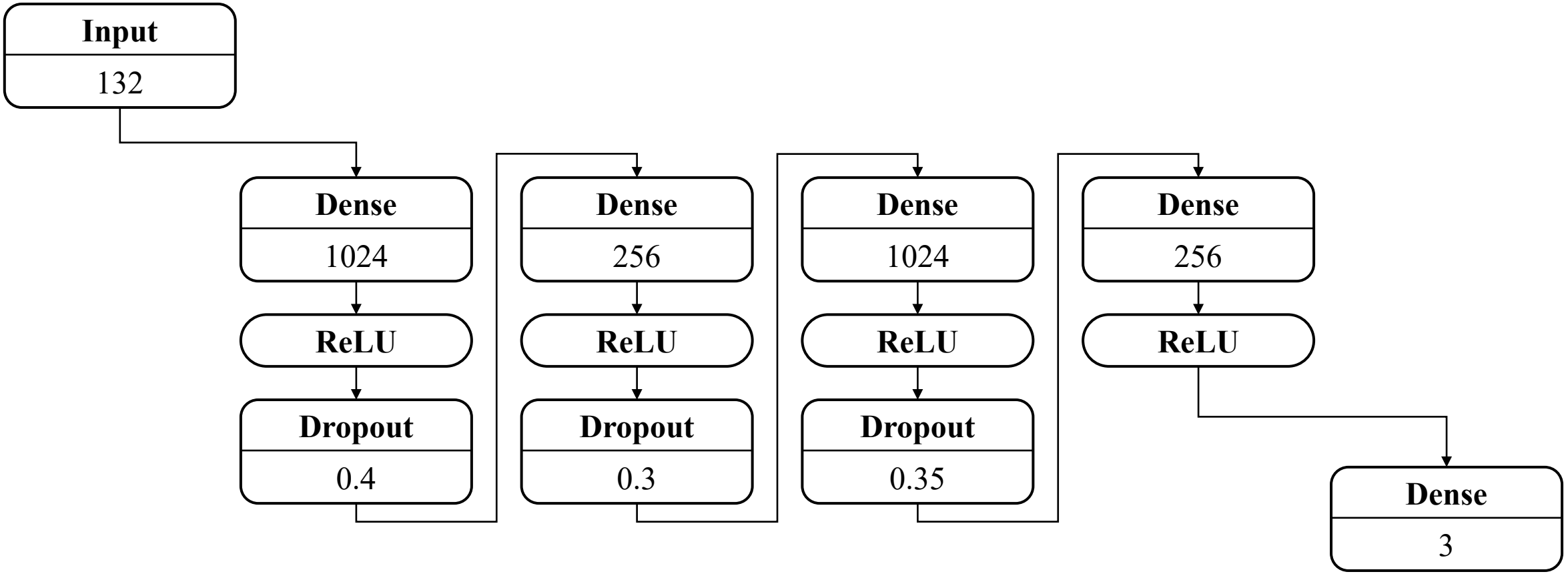


$$|Q_{cube}| = \eta^{n-1}$$



$$|Q_{cross}| = (\eta - 1)n + 1$$

Нейронная сеть для $\mathbb{R}^5$



Метрики и результаты

- Функция потерь (Bidirectional Mean Absolute Percentage Error):

$$BiMAPE = 100\frac{1}{K}\sum_{i=1}^{K}\min(\|\widehat{y_i} - y_i\|, \|\widehat{y_i} + y_i\|)$$

- Косинусное подобие (Bidirectional Mean Percentage Cosine Similarity):

$$BiMPCS = 100\frac{1}{K}\sum_{i=1}^{K}\mathrm{abs}\left(\frac{\langle\widehat{y_i}, y_i\rangle}{\|\widehat{y_i}\| \cdot \|y_i\|}\right)$$

y_i – точное значение
$\widehat{y_i}$ – выход нейронной сети
K – количество прецедентов

Dataset elements count	171000
Train/validate ratio	80:20
Optimizer	AdamW
Learning rate	0.001
Batch size	256

Точность $(100 - BiMAPE)$	**95%**
Косинусное подобие $(BiMPCS)$	**99.8%**

Спасибо за внимание!