Международная научная конференция «Параллельные вычислительные технологии 2022»
(Дубна, 29-31 марта 2022 г.)

ViLiPP: Визуализатор многомерных задач линейного программирования

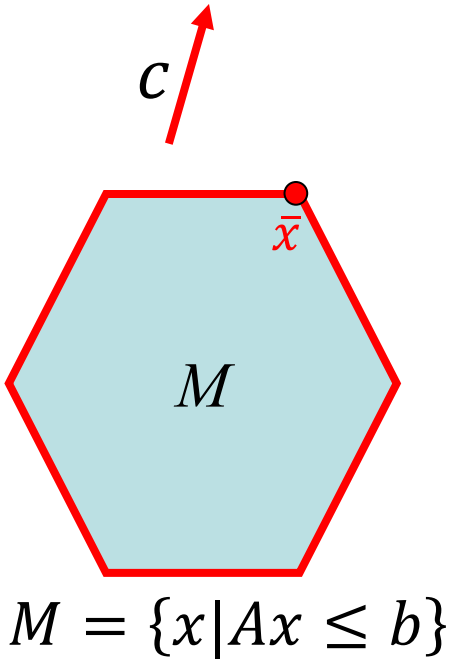
асп. Н.А. Ольховский
д.ф.-м.н. Л.Б. Соколинский

Южно-Уральский государственный университет
(национальный исследовательский университет)

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-07-00092-а и Министерства науки и высшего образования РФ (государственное задание FENU-2020-0022).

Задача линейного программирования

$$\bar{x} = \arg\max\{\langle c, x\rangle | Ax \leq b\}$$



- $x \in \mathbb{R}^n$
- A – матрица $m \times n$
- b – вектор размерности m
- c – вектор размерности n
- $\langle c, x\rangle$ – скалярное произведение

Цель исследования

Разработать новые методы использования искусственных нейронных сетей для решения многомерных задач линейного программирования

Предшествующие работы

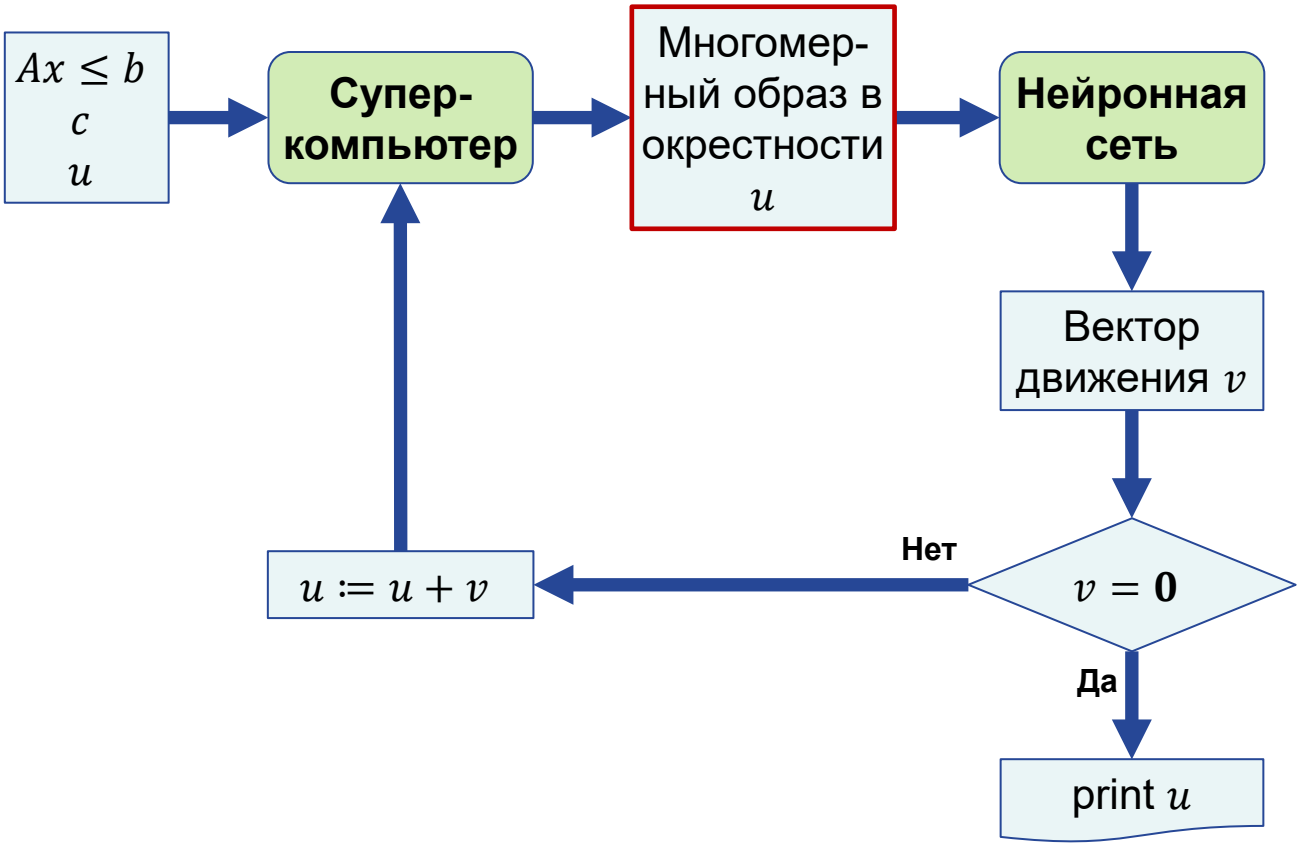
Все публикации основываются на модели Хопфилда-Танка

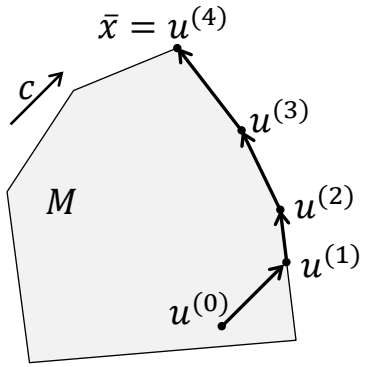
- Используется рекуррентная нейронная сеть на базе минимизации энергетической функции
- Сеть работает до достижения состояния равновесия (вход=выход)
- Архитектура нейронной сети зависит от задачи
- Необходимо придумывать энергетическую функцию и доказывать сходимость
- Нельзя предсказать количество тактов работы сети
- Достижение равновесия не гарантирует правильность ответа
- Проблемы с устойчивостью по Ляпунову

- Tank D.W., Hopfield J.J. Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit // IEEE transactions on circuits and systems. 1986. Vol. CAS-33, no. 5. P. 533-541. https://doi.org/10.1109/TCS.1986.1085953.
- Kennedy M.P., Chua L.O. Unifying the Tank and Hopfield Linear Programming Circuit and the Canonical Nonlinear Programming Circuit of Chua and Lin // IEEE Transactions on Circuits and Systems. 1987. Vol. 34, no. 2. P. 210-214. https://doi.org/10.1109/TCS.1987.1086095.
- Nonlinear Switched-Capacitor "Neural" Networks for Optimization Problems / Rodriguez-Vazquez A., Dominguez-Castro R., Rueda A., Huertas J., and Sanchez-Sinencio E. // IEEE Transactions on Circuits and Systems. 1990. Vol. 37, no. 3. P. 384-398. https://doi.org/10.1109/31.52732.
- Zak S.H., Upatising V. Solving Linear Programming Problems with Neural Networks: A Comparative Study // IEEE Transactions on Neural Networks. 1995. Vol. 6, no. 1. P. 94-104. https://doi.org/10.1109/72.363446.
- Malek A., Yari A. Primal-dual solution for the linear programming problems using neural networks // Applied Mathematics and Computation. 2005. Vol. 167, no. 1. P. 198-211. https://doi.org/10.1016/J.AMC.2004.06.081.
- Liu X., Zhou M. A one-layer recurrent neural network for non-smooth convex optimization subject to linear inequality constraints // Chaos, Solitons and Fractals. 2016. Vol. 87. P. 39-46. https://doi.org/10.1016/j.chaos.2016.03.009.

… … … … … … … … … … … … …

Идея нашего подхода





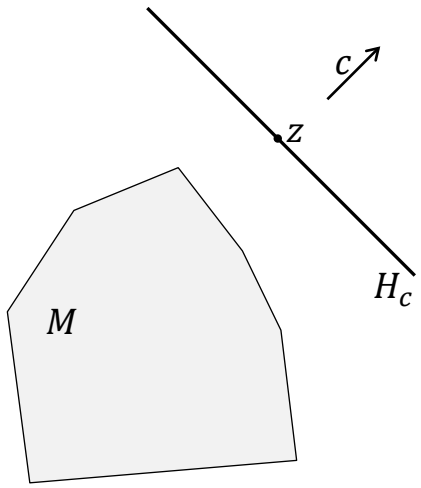
Построение многомерного образа

- Целевая гиперплоскость
- Целевая проекция
- Рецептивное поле

Целевая гиперплоскость

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

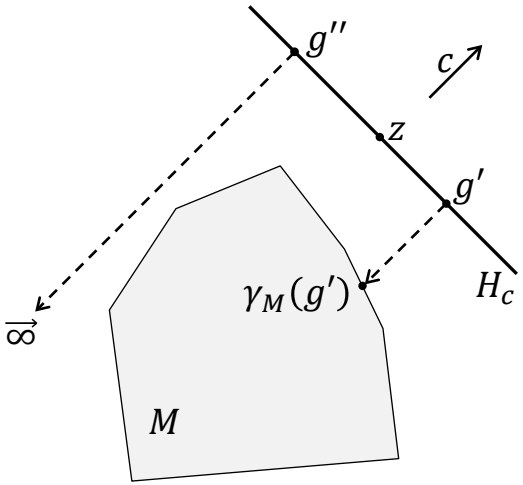
$$\forall x \in M: \langle c, x - z \rangle < 0$$



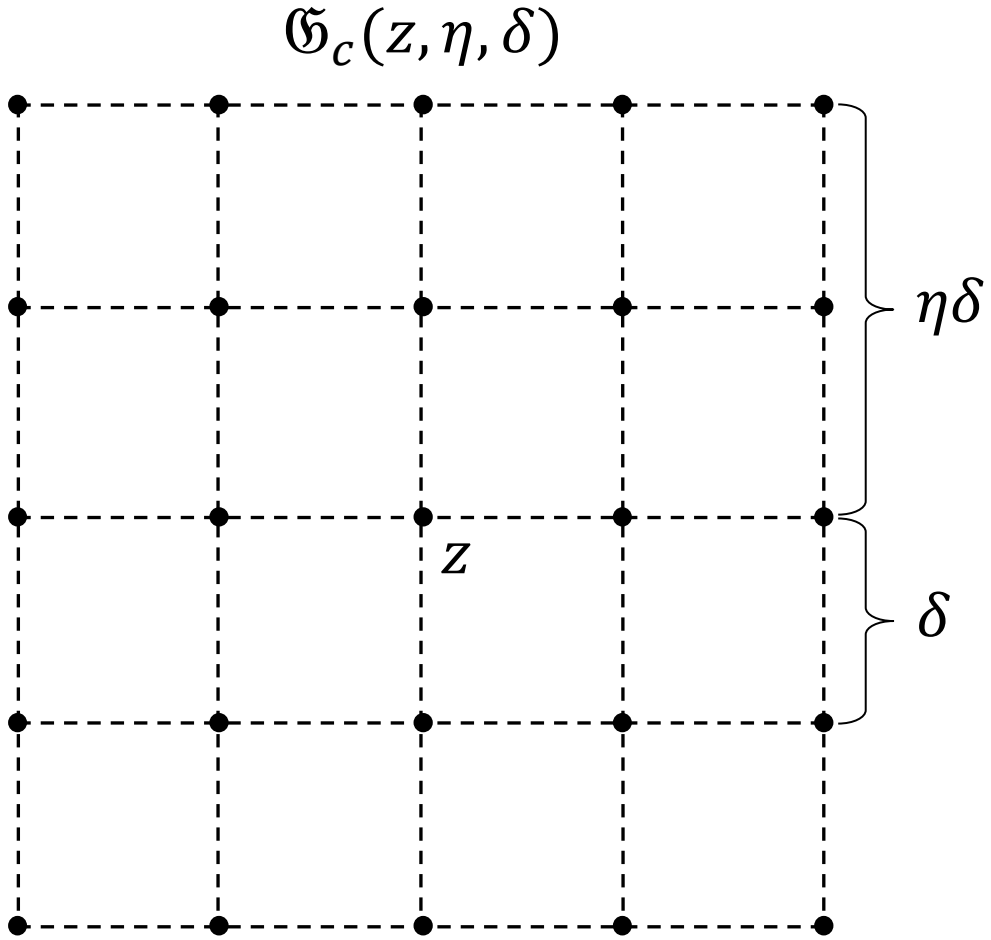
Целевая проекция на многогранник $\gamma_M(x)$

$$\gamma_M(g) = \begin{cases} g - \sigma_M(g)c, & \text{если } \exists \sigma \in \mathbb{R}_{\geq 0} : g - \sigma c \ \in M \\ \overrightarrow{\infty}, & \text{если } \neg\exists \sigma \in \mathbb{R}_{\geq 0} : g - \sigma c \ \in M \end{cases}$$

$$\sigma_M(g) = \min\{\sigma \in \mathbb{R}_{\geq 0} \mid g - \sigma c \ \in M\}$$



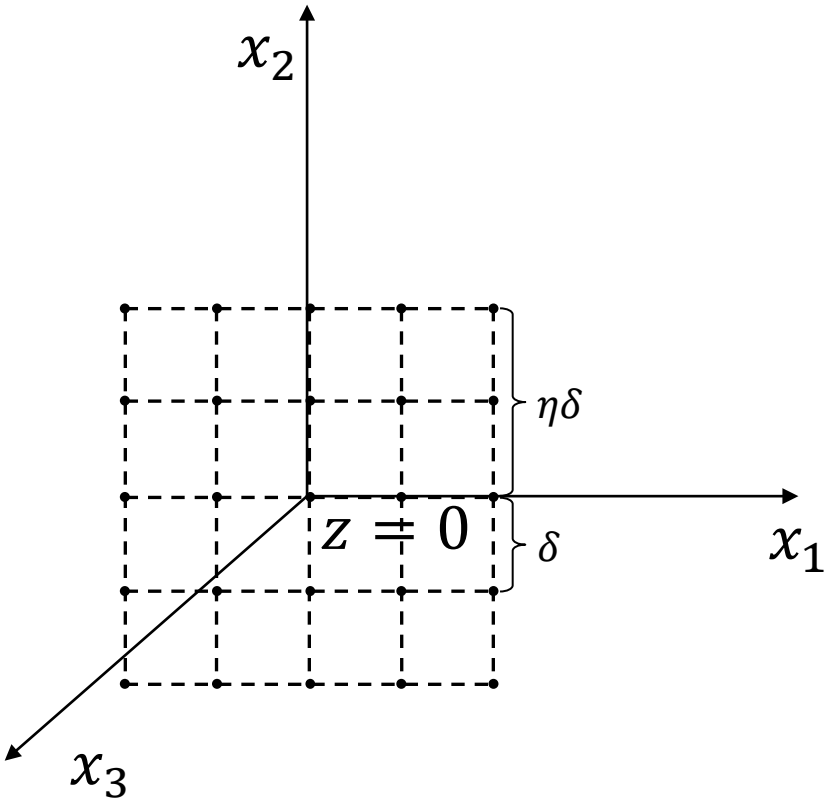
Рецептивное поле



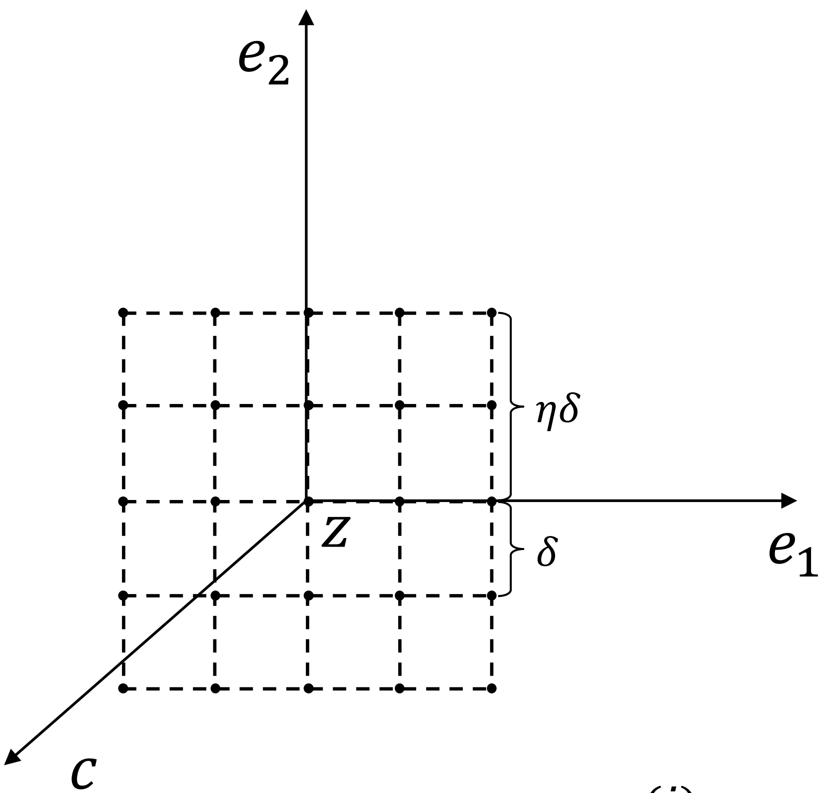
η – ранг

δ – плотность

Построение рецептивного поля



Размещение рецептивного поля в H_c



$$c^{(0)} = c = (c_1, c_2, c_3, c_4, \ldots, c_{n-1}, c_n);$$

$$c^{(1)} = \begin{cases} \left(-\frac{1}{c_1}\sum_{i=2}^{n} c_i^2, c_2, c_3, c_4, \ldots, c_{n-1}, c_n\right), \text{ если } c_1 \neq 0; \\ (1, 0, \ldots, 0), \text{ если } c_1 = 0; \end{cases}$$

$$c^{(2)} = \begin{cases} \left(0, -\frac{1}{c_2}\sum_{i=3}^{n} c_i^2, c_3, c_4, \ldots, c_{n-1}, c_n\right), \text{ если } c_2 \neq 0; \\ (0, 1, 0, \ldots, 0), \text{ если } c_2 = 0; \end{cases}$$

$$c^{(3)} = \begin{cases} \left(0, 0, -\frac{1}{c_3}\sum_{i=4}^{n} c_i^2, c_4, \ldots, c_{n-1}, c_n\right), \text{ если } c_3 \neq 0; \\ (0, 0, 1, 0, \ldots, 0), \text{ если } c_3 = 0; \end{cases}$$

....

$$c^{(n-2)} = \begin{cases} \left(0, \ldots, 0, -\frac{1}{c_{n-2}}\sum_{i=n-1}^{n} c_i^2, c_{n-1}, c_n\right), \text{ если } c_{n-2} \neq 0; \\ (0, \ldots, 0, 1, 0, 0), \text{ если } c_{n-2} = 0; \end{cases}$$

$$c^{(n-1)} = \begin{cases} \left(0, \ldots, 0, -\frac{c_n^2}{c_{n-1}}, c_n\right), \text{ если } c_{n-1} \neq 0; \\ (0, \ldots, 0, 0, 1, 0), \text{ если } c_{n-1} = 0. \end{cases}$$

$$e^{(i)} = \frac{c^{(i)}}{\left\|c^{(i)}\right\|} \ (i = 1, \ldots, n-1)$$

Координаты точки по её номеру

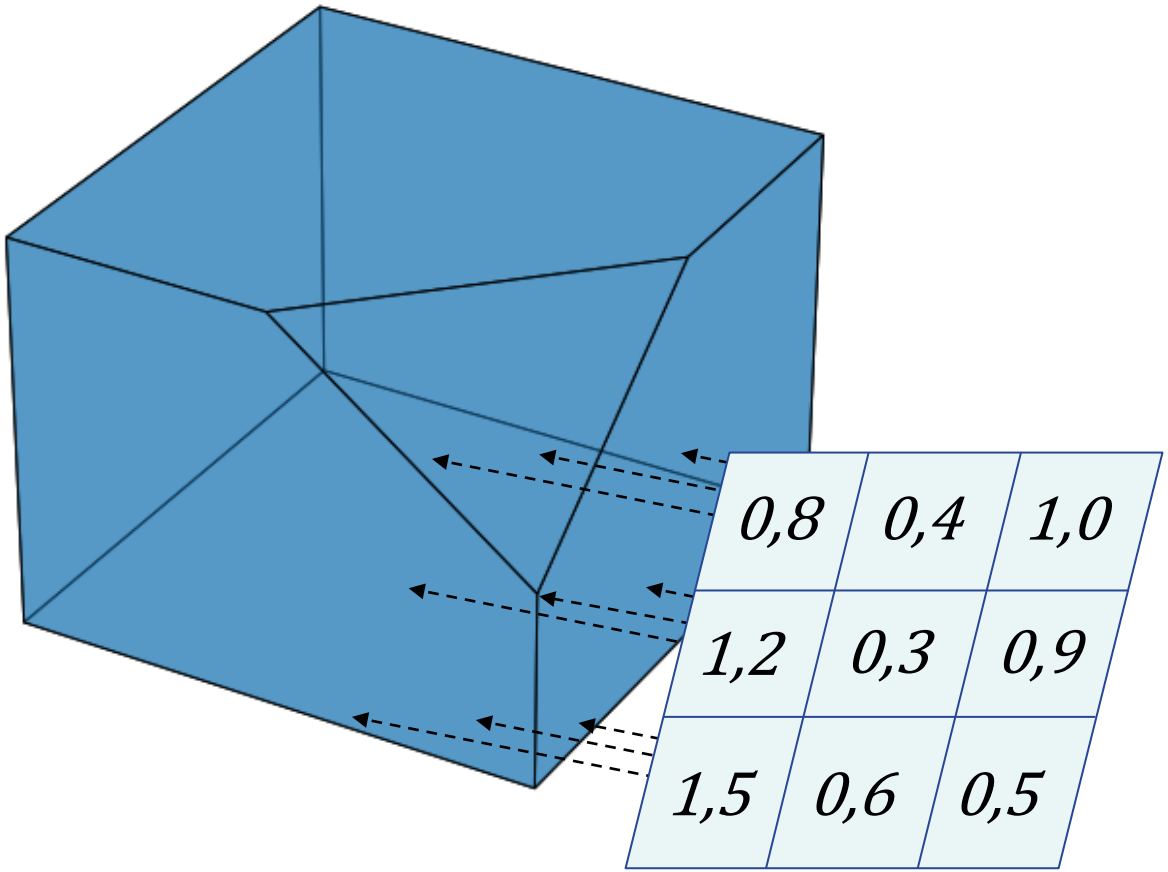
Алгоритм 2 Функция G вычисляет точку рецептивного поля по ее номеру k

Require: $z \in H_c$, $\eta \in \mathbb{N}$, $\delta \in \mathbb{R}_{>0}$

function $\mathrm{G}(k, n, z, \eta, \delta)$
  for $j = (n-1) \dots 1$ **do**
    $l_j := \lfloor k/(2\eta+1)^{j-1} \rfloor$
    $k := k \mod (2\eta+1)^{j-1}$
  end for
  $g := z$
  for $j = 1 \dots (n-1)$ **do**
    $g := g + (l_j\delta - \eta\delta)e^{(j)}$
  end for
  $\mathrm{G} := g$
end function

Временная сложность может быть оценена как $O(n^2)$

Образ рецептивного поля



Образ рецептивного поля

Алгоритм 4 Построение образа $\mathfrak{I}(z, \eta, \delta)$

Require: $z \in H_c$, $\eta \in \mathbb{N}$, $\delta \in \mathbb{R}_{>0}$

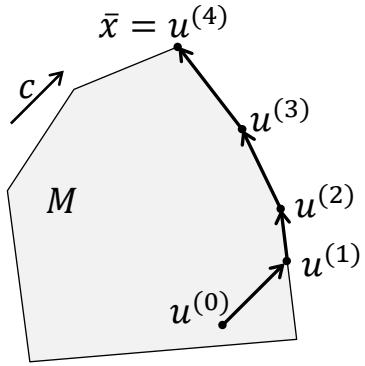
function $\mathfrak{I}(z, \eta, \delta)$
    $\mathfrak{I} := [\,]$
    for $k = 0 \ldots ((2\eta + 1)^{n-1} - 1)$ **do**
        $g_k := \mathrm{G}(k, n, z, \eta, \delta)$
        $\mathfrak{I} := \mathfrak{I} +\!\!+ [\rho_c(\gamma_M(g_k))]$
    end for
end function

Сопряжение с нейронной сетью

Алгоритм 5 Линейное программирование с использованием DNN

Require: $u^{(1)} \in H_i \cap M,\ \langle \tilde{a}_i, c \rangle > 0,\ z \in H_c;\ \eta \in \mathbb{N},\ \delta \in \mathbb{R}_{>0}$

1: $k := 1$
2: **repeat**
3: $\quad \mathcal{I} := \mathfrak{I}(u^{(k)}, \eta, \delta)$
4: $\quad u^{(k+1)} := \mathrm{DNN}(\mathcal{I})$
5: $\quad k := k + 1$
6: **until** $u^{(k)} \neq u^{(k-1)}$
7: $\bar{x} := u^{(k)}$
8: **stop**



Нужен суперкомпьютер

$$K_{\mathfrak{G}} = (2\eta + 1)^{n-1}$$

Параллельный алгоритм

Алгоритм 7 Параллельный алгоритм построение образа $\mathfrak{I}$ задачи ЛП

Мастер	**Рабочий** $(l=0,\ldots,L-1)$
1: **input** n	1: **input** $n, m, A, b, c, z, \eta, \delta$
2: $\mathfrak{I} := [\,]$	2: $L := \text{NumberOfWorkers}$
3: $k := 0$	3: $\mathcal{L}_{map(l)} := [lm/L, \ldots, ((l+1)m/L) - 1]$
4: **repeat**	4: **repeat**
5: **SendToWorkers** k	5: **RecvFromMaster** k
6:	6: $\mathcal{L}_{reduce(l)} := \text{Map}\left(\text{F}_k, \mathcal{L}_{map(l)}\right)$
7:	7: $\rho_l := \text{Reduce}\left(\Downarrow, \mathcal{L}_{reduce(l)}\right)$
8: **RecvFromWorkers** $[\rho_0, \ldots, \rho_{L-1}]$	8: **SendToMaster** ρ_l
9: $\rho := \text{Reduce}\left(\Downarrow, [\rho_0, \ldots, \rho_{L-1}]\right)$	9:
10: $\mathfrak{I} := \mathfrak{I} +\!\!+ [\rho]$	10:
11: $k := k + 1$	11:
12: $exit := \left(k \geqslant (2\eta+1)^{n-1}\right)$	12:
13: **SendToWorkers** $exit$	13: **RecvFromMaster** $exit$
14: **until** $exit$	14: **until** $exit$
15: **output** $\mathfrak{I}$	15:
16: **stop**	16: **stop**

Sokolinsky L.B. BSF: A parallel computation model for scalability estimation of iterative numerical algorithms on cluster computing systems // Journal of Parallel and Distributed Computing. 2021. Vol. 149. P. 193-206. DOI:10.1016/j.jpdc.2020.12.009

Реализация

https://github.com/nikolay-olkhovskiy/LP-visualization-MPI

- C++
- MPI & OpenMP
- Параллельный программный каркас BSF
  - Sokolinsky L.B. BSF-skeleton: A template for parallelization of iterative numerical algorithms on cluster computing systems // MethodsX. 2021. Vol. 8. Article 101437. DOI: 10.1016/j.mex.2021.101437
  - https://github.com/leonid-sokolinsky/BSF-skeleton

Суперкомпьютер «Торнадо ЮУрГУ»

Количество узлов:	**384**
Тип процессоров:	**2 x Intel Xeon X5680 (12 ядер по 3.33 ГГц; 2 потока на ядро)**
Оперативная память узла:	**24 Гб**
Тип сопроцессора:	**Intel Xeon Phi SE10X: (61 ядро по 1.1 ГГц; 4 потока на ядро)**
Память сопроцессора:	**8 Гб**
Тип системной сети:	**InfiniBand QDR**
Тип управляющей сети:	**Gigabit Ethernet**
Операционная система:	**Linux CentOS 6.2**

Вычислительные тесты

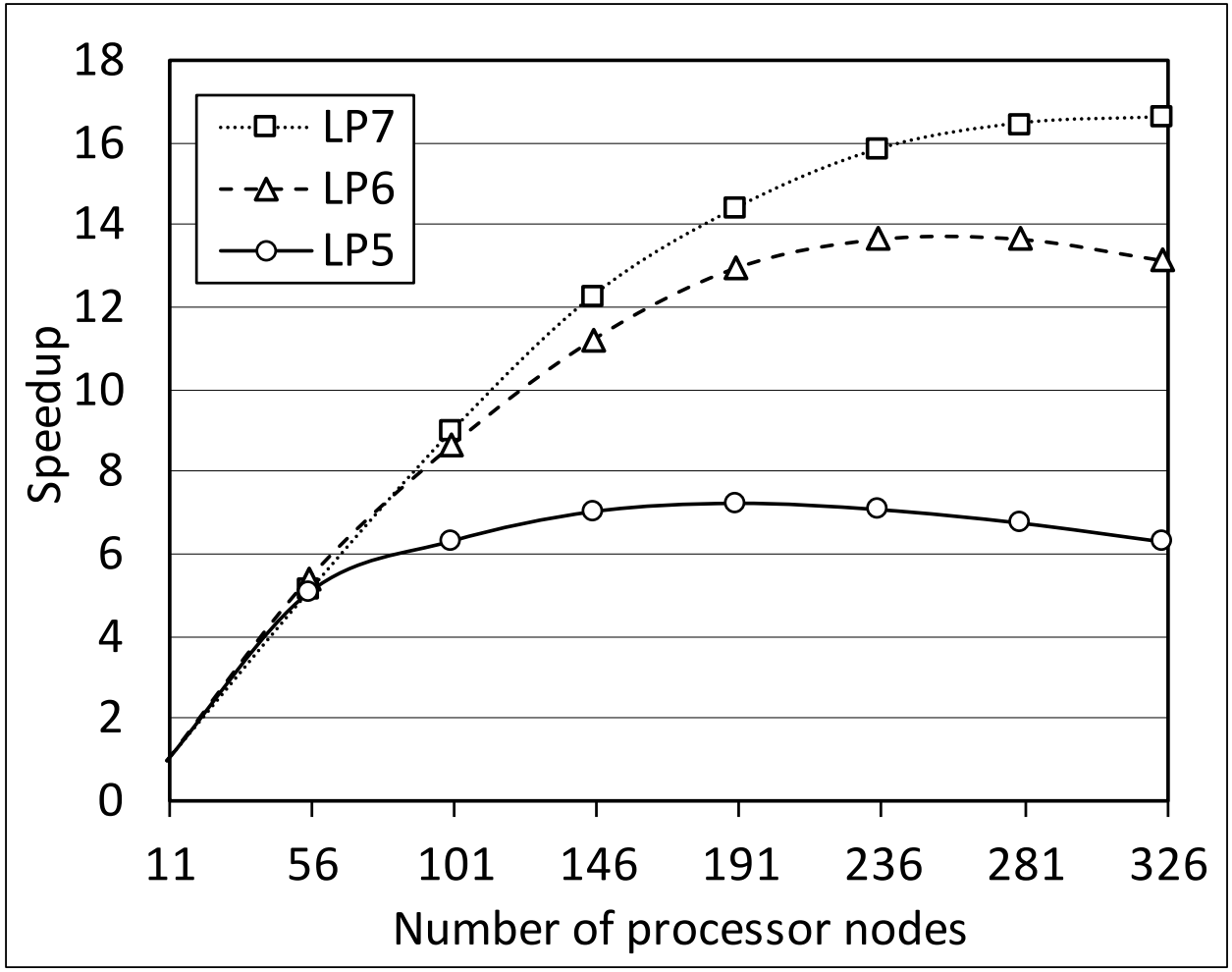
Таблица 2. Параметры тестовых задач ЛП

Идентификатор задачи	Число переменных	Число ограничений	Процент ненулевых значений в A	Мощность рецептивного поля
$LP7$	7	4016	100%	15 625
$LP6$	6	4014	100%	3 125
$LP5$	5	4012	100%	625

Sokolinsky L.B., Sokolinskaya I.M. FRaGenLP: A Generator of Random Linear Programming Problems for Cluster Computing Systems // Parallel Computational Technologies. PCT 2021. Communications in Computer and Information Science. 2021, vol. 1437. 164-177. DOI:10.1007/978-3-030-81691-9_12

https://github.com/leonid-sokolinsky/BSF-LPP-Generator

Масштабируемость



Применимость на практике

- Наблюдается экспоненциальный рост времени решения задачи:

Размерность	Время (на 11 узлах)
$n = 5$	10 секунд
$n = 7$	5 минут
$n = 9$	1.5 часа

- При современном уровне развития вычислительной техники, предложенный метод может быть эффективен для задач ЛП с $n \leq 100$ и $m \leq 100\,000$.

Спасибо за внимание!