Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Факультет информационных технологий и программирования
Кафедра компьютерных технологий

Разработка гибридного алгоритма недоминирующей сортировки

Маркина Маргарита Анатольевна
Группа M3438

Научный руководитель: к.т.н. доцент кафедры КТ
М. В. Буздалов

Решаемая проблема

Предметная область

- Недоминирующая сортировка.
- Многокритериальная задача оптимизации.
- Гибридизация алгоритмов.
- Оценка времени работы алгоритмов.

Решаемая проблема

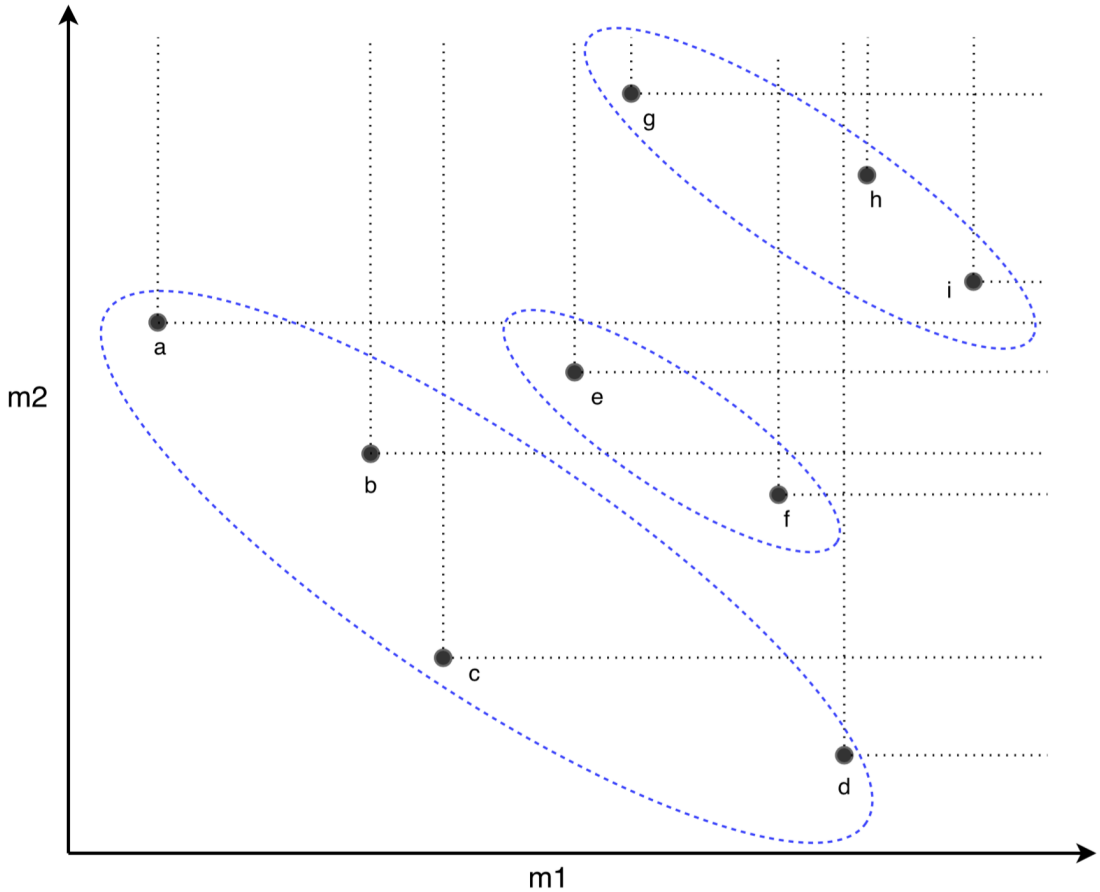
Введение

- Точка $A = (a_1, ..., a_M)$ доминирует точку $B = (b_1, ..., b_M)$, когда $\forall\ 1 \leq i \leq M : a_i \leq b_i$ и $\exists j : a_j < b_j$.
- *Недоминирующая сортировка* множества точек S в M-мерном пространстве — это процедура, назначающая всем точкам из S *ранг*.
- Все точки, которые не доминируются ни одной точкой из S, имеют ранг 0.
- Точка имеет ранг $i + 1$, если максимальный ранг среди доминирующих её точек равен i.

Решаемая проблема

Введение

На рисунке 3 фронта:
$\{a, b, c, d\}$ имеет ранг 0, $\{e, f\}$ - ранг 1, $\{g, h, i\}$ - ранг 2.



Решаемая проблема

Актуальность

- Многокритериальные эволюционные алгоритмы.
- Задача минимазациии.

Решаемая проблема

Цель исследования

- Выбрать наиболее подходящие алгоритмы.
- Выявить преимущества каждого алгоритма.
- Научиться по входным данным выбирать стратегию.
- Сделать гибридный алгоритм.

Решение

Fast + BOS

- Fast Version of the Generalized Algorithm.
- Best Order Sort.

Решение

Fast

- Fast Version of the Generalized Algorithm.
  - Разделяй и властвуй по N и M.
  - На каждом этапе делим на 3 множества по k_i критерию текущее множество точек.
  - Если все k_i в одном из подмножеств равны между собой, переходим к k_{i-1}
  - Запускаемся рекрсивно.

Решение

BOS

- Best Order Sort.
  - M отсортированных списков, i список отсортирован по i критерию.
  - Далее определяем ранг начиная с наиболее подходящих элементов.
  - Во время определения ранга используем уже обработанные точки.

Решение

Асимптотика

- Fast $O(N \log^{M-1} N)$.
- BOS $O(MN \log N + MN^2)$.
  - лучшем случае – за $\Theta(MN \log N)$
  - худшем случае – за $\Theta(MN^2)$

Решение

Гибридизация

- По входным данным подбирать стратегию сортировки
- В момент рекурсивного запуска мы можем переключиться на BOS.
- Оценка входных данных должна быть очень быстрая, так как она принимается неоднократно.

Практические результаты

Эксперименты

Рассмотриваем влияние входные данных на время работы алгоритмов

- Случайные точки в гиперкубе.
- Точки одного ранга.

Практические результаты

Случайные точки в гиперкубе

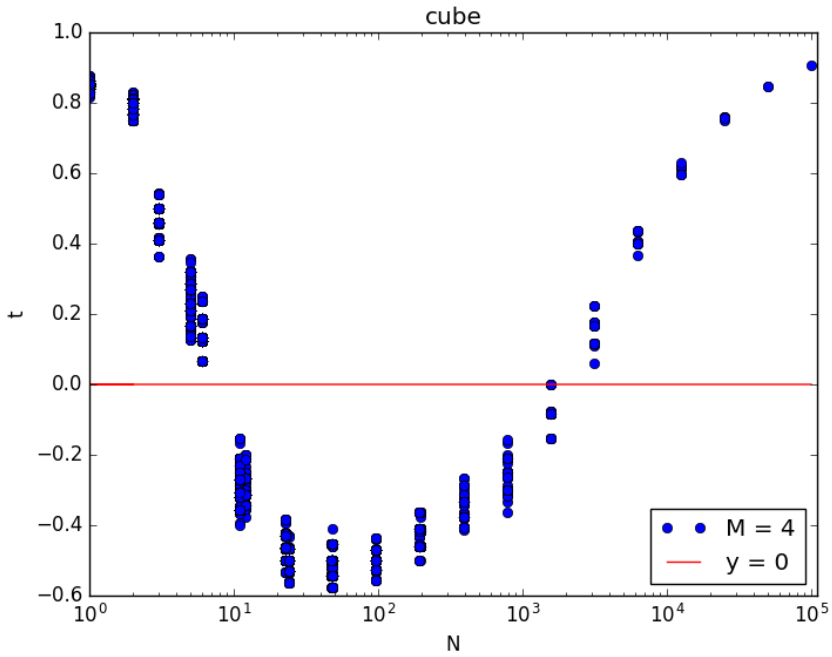
Алгоритм запускался для N = 100 000 при
$M \in \{4, 6, 8, 10, 12, 14, 16, 18, 20\}$

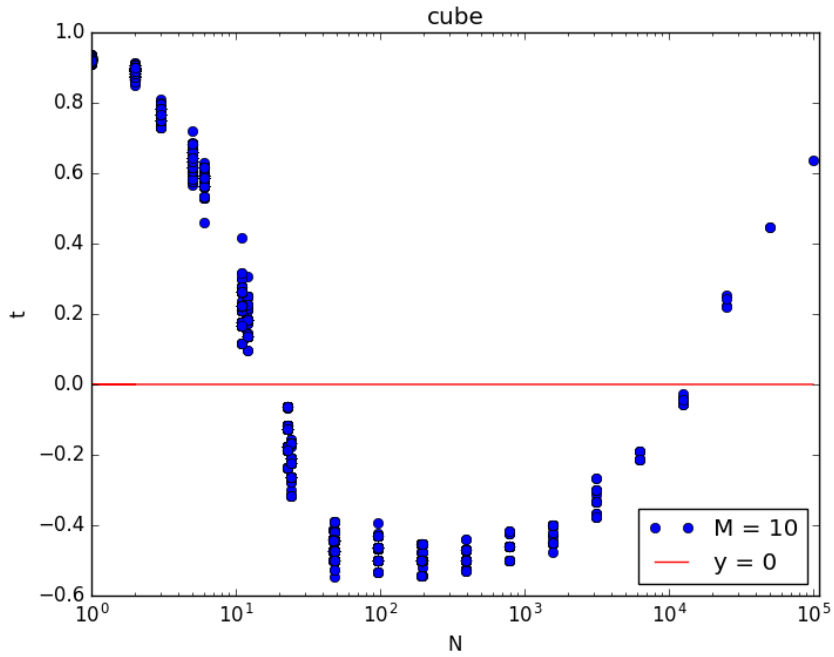
- T_{Fast} - время за которое алгоритм Fast отсортировал экспериментальное множество точек S.
- T_{BOS} - время алгоритма BOS.
- $T_{\text{max}} = max(T_{\text{BOS}}, T_{\text{Fast}})$
- Оценивать будем с помощью графика, где по абсциссе будет мощность множества S для которого проводился эксперимент. По ординате будет $\frac{T_{\text{BOS}} - T_{\text{Fast}}}{T_{\text{max}}}$.

Практические результаты

Случайные точки в гиперкубе

$$t = \frac{T_{BOS} - T_{Fast}}{T_{max}}.$$

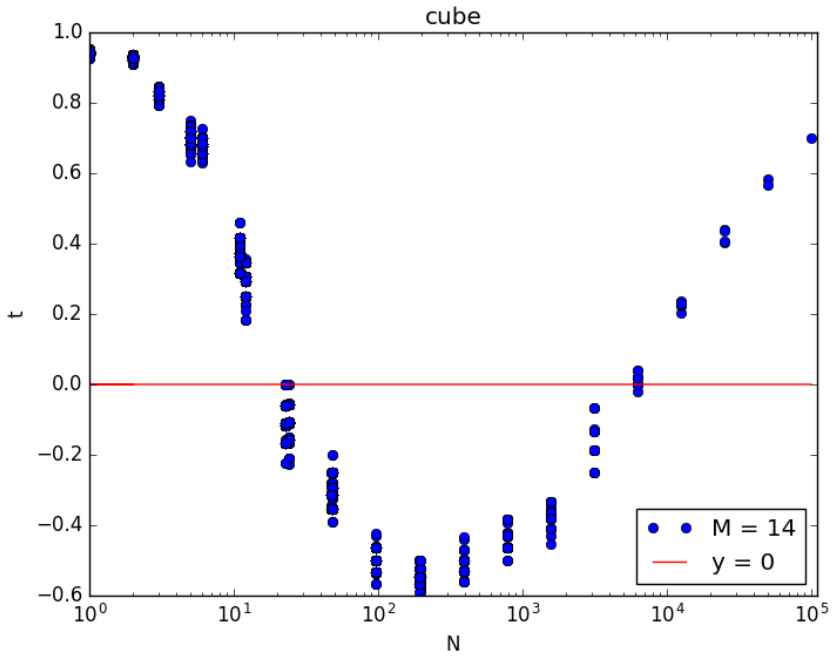


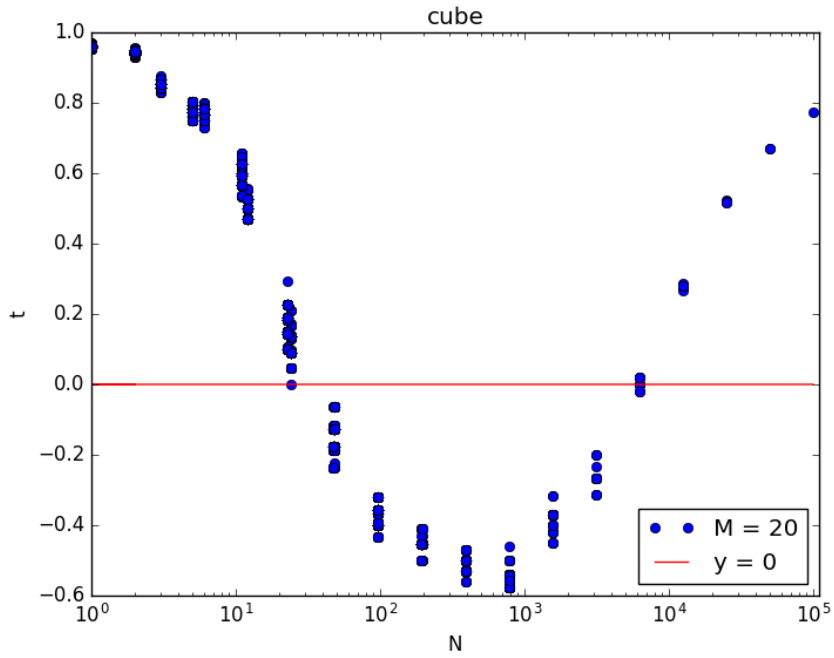


Практические результаты

Случайные точки в гиперкубе

$$t = \frac{T_{BOS} - T_{Fast}}{T_{max}}.$$



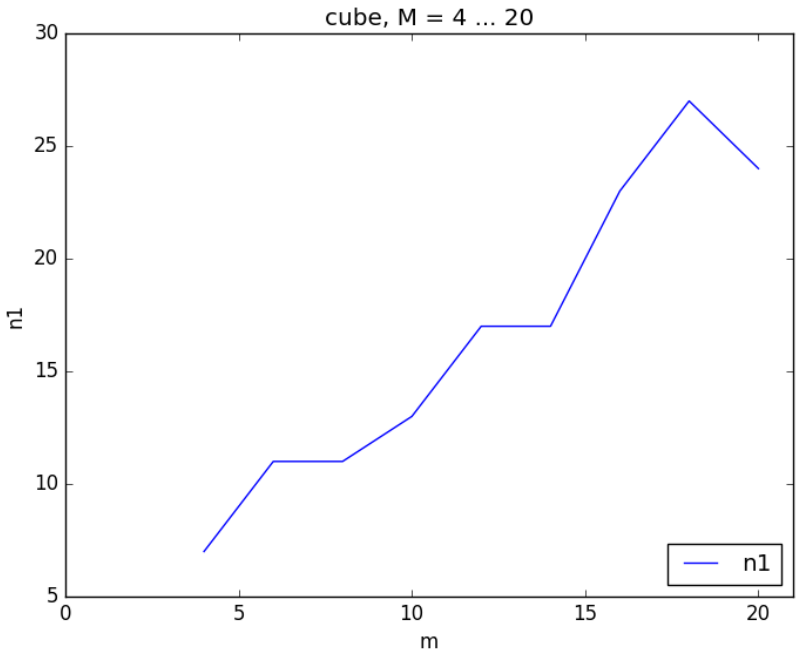


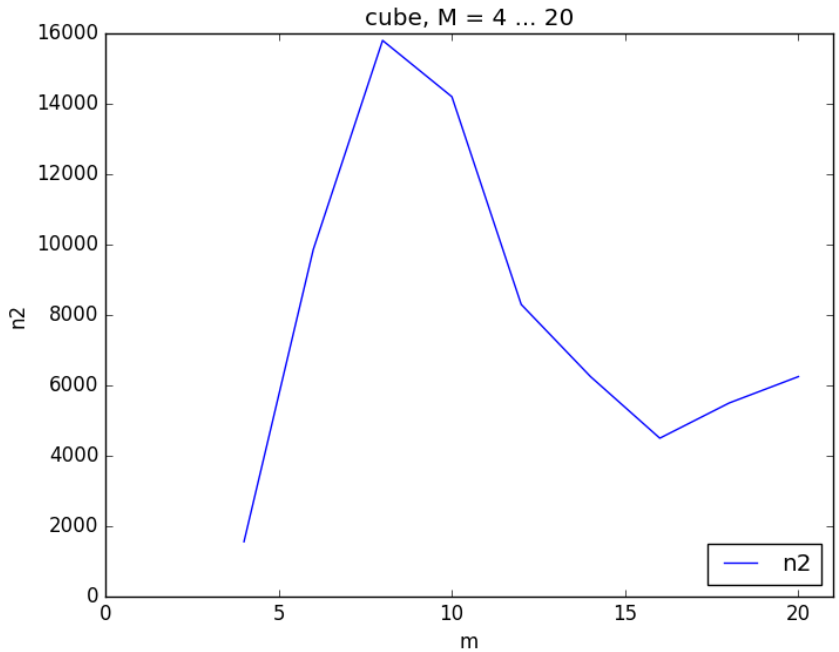
Практические результаты

Случайные точки в гиперкубе

Левая и правая границы:

- m - размерность входных данных,
- n1 и n2 - размеры входных данных при которых меняется более эффективный алгоритм.

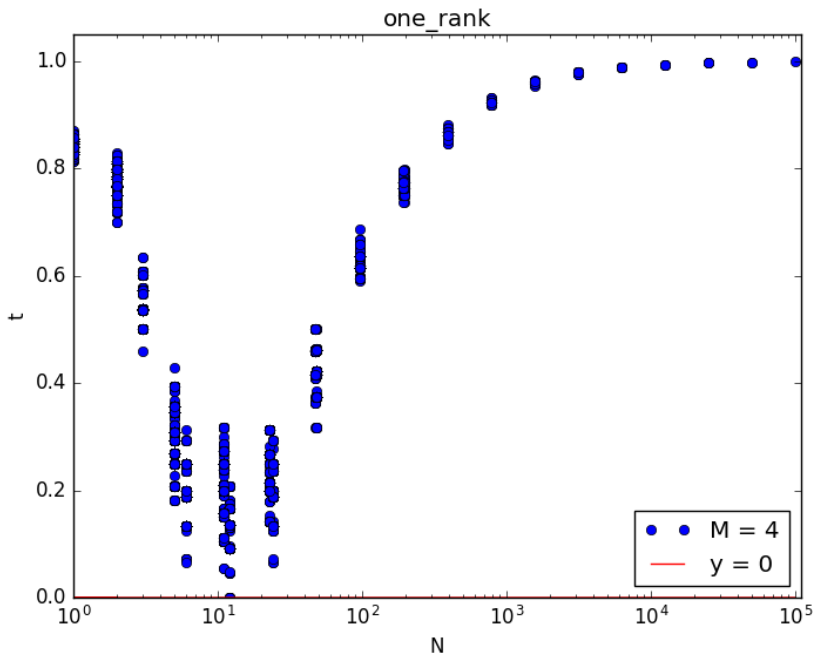


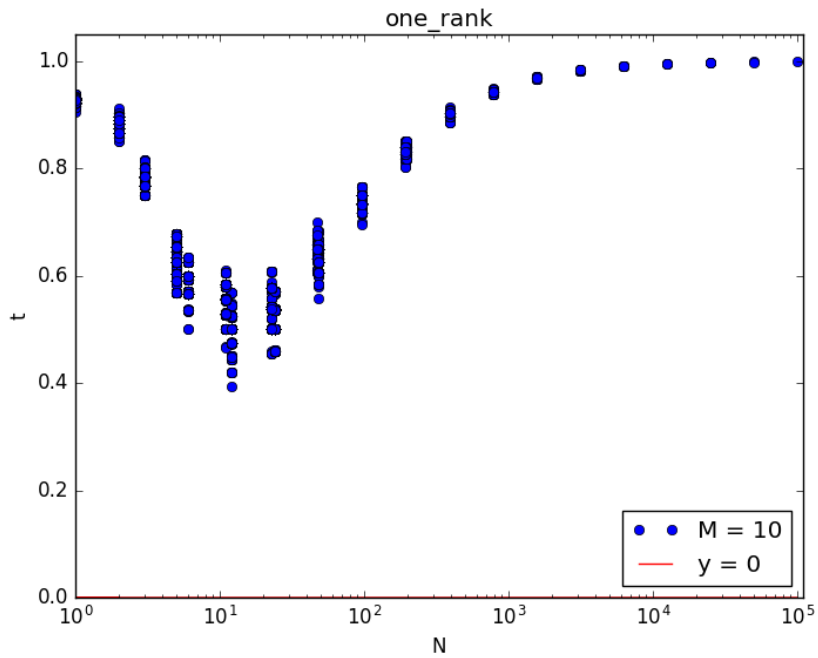


Практические результаты

Точки одного ранга

$$t = \frac{T_{BOS} - T_{Fast}}{T_{max}}.$$

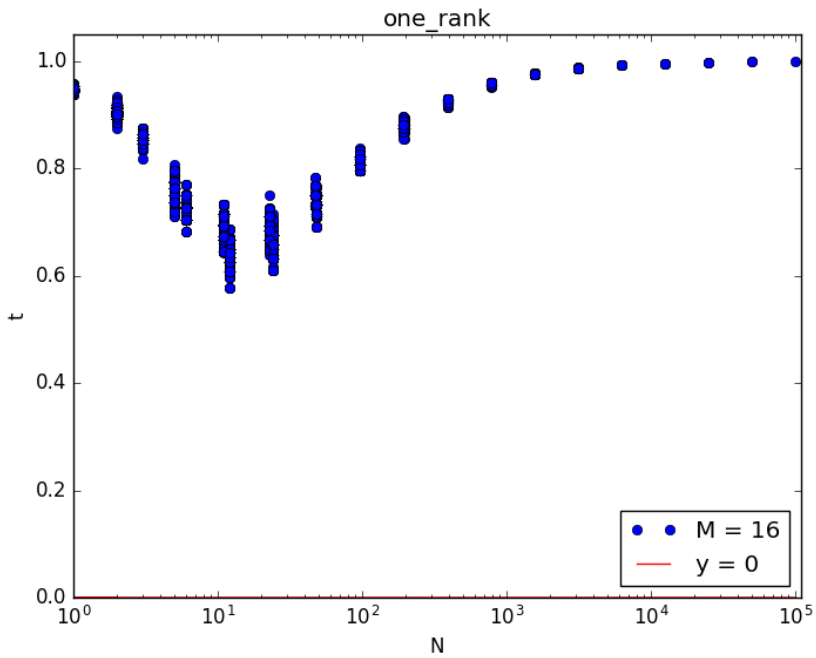


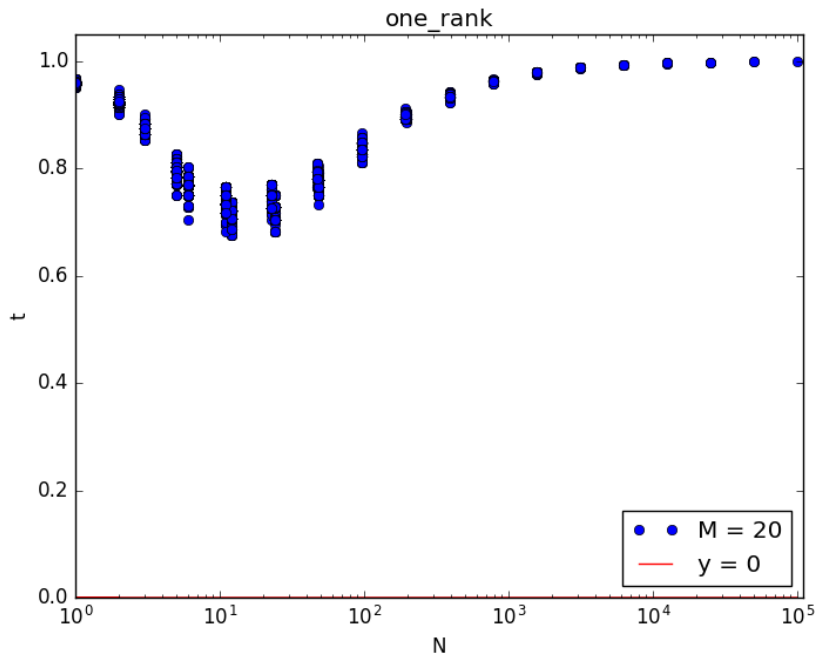


Практические результаты

Точки одного ранга

$$t = \frac{T_{BOS} - T_{Fast}}{T_{max}}.$$





Практические результаты

Выводы

- Стратегия выбора алгоритма, основывающаяся на размере входных данных и их размерности, будет достаточно эффективной на случайных входных данных
- Для более эффективной работы в крайних случаях требуется предсказывать число слоев входных данных.

Дальнейшие действия

- Выявить зависимость правой границы для случайных входных данных
- Научиться детектировать крайние случаи и разработать стратегию выбора наиболее подходящего алгоритма для них

Спасибо за внимание!

Дополнительные материалы