Улучшение работы планировщика задач в суперкомпьютерных системах с помощью алгоритмов машинного обучения.

Проблема

Статистика работы суперкомпьютерных систем коллективного пользования показывает, что фактическое время выполнения большинства заданий существенно расходится со временем, запрошенным пользователем (в среднем, в 2.3 раза[3]). Это снижает эффективность планирования заданий, поскольку неточная оценка времени выполнения приводит к неоптимальному расписанию запусков заданий. Для повышения качества планирования выполнения заданий предлагается использовать в планировщике предсказания времени выполнения заданий, полученные с помощью алгоритмов машинного обучения[1, 2, 3].

# Анализ существующих работ

Существует две основные стратегии предсказывания времени выполнения программ.

* Первая стратегия заключается в анализе кодов и их скомпилированных файлов [4]. Однако платформа планирования часто не имеет разрешения на получение кода для задания, которое является интеллектуальной собственностью пользователя.
* Вторая стратегия основываются на том, что пользователи суперкомпьютерных систем склонны многократно запускать одно и то же приложение (возможно, с разными параметрами). Этот атрибут вызвал множество работ, в которых исследователи пытались предсказать время выполнения программ на основе исторической информации.

Данное исследование концентрируется на второй стратегии, как наиболее доступной.

В информацию, используемую второй стратегией, чаще всего входит статистика, собираемая системой планировщика: имя пользователя, группа пользователя, имя задачи, запрошенные время, память, количество узлов, реальные память и время выполнения, время постановки в очередь.

Методы предсказания можно разделить на 3 группы:

* Метод прогнозирования на основе шаблонов - этот метод определяет сходство заданий с помощью шаблона, разделенного на атрибуты задания для прогнозирования. Время выполнения оценивается как характерное для данного шаблона [5]
* Метод прогнозирования на основе экземпляров - данный метод не классифицирует данные истории. Вместо этого он ищет k фрагментов задания, похожих на новое прогнозируемое задание, а затем на их основе оценивает время выполнения[6,7]. Часто, подобные алгоритмы основаны на алгоритме k ближайших соседей, представляя экземпляры как точки в n - мерном пространстве.
* Метод прогнозирования на основе сглаживания Этот метод был предложен в [8] для оценки времени выполнения, используя среднее значение двух последних выполнений, представленных одним и тем же пользователем . Этот метод очень прост в реализации и обеспечивает высокую производительность.

# Цель

Как следует из [1], целесообразность применения алгоритмов машинного обучения в планировщиках задач возникает начиная с некоторого порога качества предсказания(около 80%), следовательно, целью текущих исследований является улучшение качества предсказаний.

Целью данной работы будет исследование использования аргументов командной строки запускаемой программы при предсказании времени работы программы.

В результате планируется получить алгоритм обучения и модель на основании машинного обучения, имеющую достаточную для применения в реальном планировщике задач точность.

При успешных результатах работы возможен пересмотр стратегии сбора статистики в суперкомпьютерных системах.

# Постановка задачи

Сформируем соответствующую постановку задачи. Дана статистика запуска задач в суперкомпьютерной системе. Пользовательские запуски задач ограничены по времени 24 часами. Задачи разбиты на 24 класса, в зависимости от длительности. Необходимо создать модель, наиболее точно предсказывающую класс времени выполнения новой поставленной задачи на основании исторической информации. Информация собирается из системы планирования СУППЗ (система управления прохождением параллельных заданий)[9] МСЦ РАН. Для предсказания предлагается использовать разновидность прогнозирования на основе экземпляров.

Статистка запуска задач собирает следующую важную для предсказания информацию:

* Время постановки в очередь
* id пользователя, его группы пользователей, его организации
* количество запрошенных процессов
* количество запрошенного времени
* фактическая длительность с момента запуска задачи на выполнение

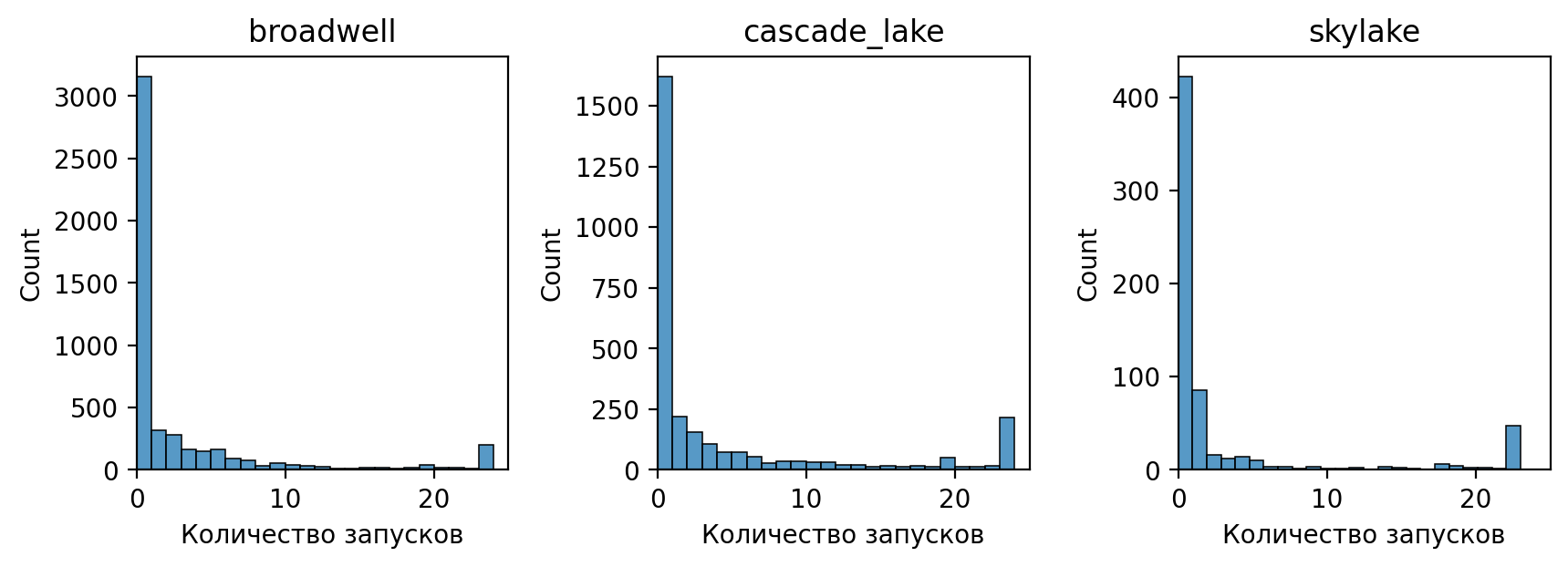
Для проведения исследования были поставлены следующие задачи:

1. Изменение алгоритма сбора статистики для добавления в статистику аргументов командной строки запускаемой задачи
2. Разработка метода кодирования аргументов командной строки для их использования при построении модели прогнозирования времени работы программы.
3. Выбор лучшей модели для предсказания фактической длительности задачи на основании: времени постановки в очередь, id пользователя, группы, организации, названия задачи, количества запрошенных процессов и времени, аргументов командной строки запускаемого в задаче файла. Анализ работ показывает, что чаще всего для предсказания используют вариации классификаторов на основе регрессии(линейная, гребенчатая, с помощью деревьев принятия решений, эластичная сеть, лассо)[10][11] или метрические классификаторы (KNN)
4. Реализация методов прогнозирования.
5. Проведение тестирования реализованных методов прогнозирования.

# Подготовка данных

Для обучения были взяты данные 3 кластеров: "broadwell", "cascade\_lake", "skylake".

Распределение классов:



Данные были очищены от выбросов – задач, которые были принудительно завершены в связи с превышением запрошенного лимита времени. Затем данные были нормированны.

# Рассматриваемые модели

Было выбрано 3 модели:

1)Классификатор на основе ребенчатой регрессии

2) Классификатор на основе дерева решений

3) Классификатор на основе К ближайших соседей

Гребенчатая регрессия была выбрана из-за высоких показателей корреляции между id пользователя, id группы и временем постановки задач в очередь.

Деревья решений были самым популярным решением в схожих задачах([2][12])

KNN – классификатор был выбран как подходящая реализация для метода прогнозирования на основе экземпляров

# Кодирование аргументов командной строки

Для улучшения алгоритма KNN была создана пользовательская метрика:

1. Все командные строки, содержащиеся в истории кластера, разбиваются на отдельные команды: если за ключом, начинающемся с символа ‘-‘ следует слово без ‘-‘, то оно включается в состав предыдущего
2. Из команд собирается уникальный словарь
3. Теперь каждая командная строка кодируется вектором из 0 и 1 – наличия либо отсутствия команды в строке
4. Тогда если запись состоит из полей:

Nproc – количество запрошенных процессоров

Ntime – запрошенное время

Userid – id пользователя

Gid – id группы

Orgid – id огранизации

Cmdline – векторное представление командной строки

Timein – время поступления задачи в очередь

То расстояние между двумя записями вычисляется как



Где ,

# Промежуточные результаты



Предварительные результаты выполнения моделей, оптимальные параметры ищутся с помощью grid и random search.

## References:

1) Шумилин С.С. , Воробьев М.Ю. “Использование симулятора планировщика заданий для оценки эффективности предсказания времени работы задания” http://swsys.ru/index.php?page=article&id=4883&lang=

2) Баранов А.В., Николаев Д.С. Применение машинного обучения для прогнозирования времени выполнения суперкомпьютерных заданий http://swsys.ru/index.php?page=article&id=4700

3) Savin, G.I., Lyakhovets, D.S. & Baranov, A.V. Influence of Job Runtime Prediction on Scheduling Quality. *Lobachevskii J Math* **42**, 2562–2570 (2021). https://doi.org/10.1134/S1995080221110196

4)D. Tetzlaff and S. Glesner, "Intelligent prediction of execution times," 2013 Second International Conference on Informatics & Applications (ICIA), 2013, pp. 234-239, doi: 10.1109/ICoIA.2013.6650262.

5) Gibbons R (1997) A historical application profiler for use by parallel schedulers. In: Workshop on job scheduling strategies for parallel processing. Springer, pp 58–77

6)Tanash, Mohammed & Dunn, Brandon & Andresen, Daniel & Hsu, William & Yang, Huichen & Okanlawon, Adedolapo. (2019). Improving HPC System Performance by Predicting Job Resources via Supervised Machine Learning. PEARC '19: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning). 2019. 1-8. 10.1145/3332186.3333041.

7) Senger LJ, Santana MJ, Santana RC (2004) An instance-based learning approach for predicting execution times of parallel applications. In: Proceedings of international information and telecommunication technologies symposium, pp 9–15

8) Tsafrir D, Etsion Y, Feitelson D (2007) Backfilling using system-generated predictions rather than user runtime estimates. IEEE Trans Parallel Distrib Syst 18(6):789–803. doi:[10.1109/TPDS.2007.70606](https://doi.org/10.1109/TPDS.2007.70606)

9) <http://suppz.jscc.ru/>

10) Chupakhin, A.A., Kolosov, A., Smeliansky, R.L., Antonenko, V.A., & Ishelev, G. (2020). New approach to MPI program execution time prediction. ArXiv, abs/2007.15338.

11) Tanash, Mohammed & Dunn, Brandon & Andresen, Daniel & Hsu, William & Yang, Huichen & Okanlawon, Adedolapo. (2019). Improving HPC System Performance by Predicting Job Resources via Supervised Machine Learning. PEARC '19: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning). 2019. 1-8. 10.1145/3332186.3333041.

12) Guo, J., Nomura, A., Barton, R., Zhang, H., Matsuoka, S. (2018). Machine Learning Predictions for Underestimation of Job Runtime on HPC System. In: Yokota, R., Wu, W. (eds) Supercomputing Frontiers. SCFA 2018. Lecture Notes in Computer Science(), vol 10776. Springer, Cham. https://doi.org/10.1007/978-3-319-69953-0\_11