УДК 621.391

ПРОГНОЗИРОВАНИЕ ВЫЧИСЛИТЕЛЬНОЙ НАГРУЗКИ В КРИТИЧНЫХ ИТ СИСТЕМАХ

А. А. СТАРОВОЙТОВ, д-р техн. наук, проф. В. В. КРАСНОПРОШИН
(Белорусский государственный университет, г. Минск)

Аннотация. *В работе изложены принципы построения (с использованием нейросетевых технологий) и реализации алгоритма функционирования (в условиях неопределенности профиля внешней нагрузки) комбинированной проактивной системы управления вычислительными ресурсами критичной ИТ системы.*

Ключевые слова: *принятие решений; информационная система; проактивное управление; неопределенность внешней нагрузки; нейронные сети; нейромодуль.*

Введение. Поддержка критических ИТ сервисов и систем (банковских, телекоммуникационных, промышленных) в рабочем состоянии (с гарантированным объемом вычислительных ресурсов) является актуальной проблемой современного цифрового общества.

Неопределенность внешней нагрузки, отказы вычислительного оборудования приводят к сбоям в работе и деградации производительности критичных ИТ систем. В результате этого теряется оперативность обработки информации и проведения банковских и других операций. Все это в свою очередь может иметь серьезные последствия (финансовые потери, крупные аварии и т.п.).

Принятие оперативных решений по управлению критичными ИТ сервисами позволяет уменьшить (или не допустить) возникновение негативных последствий. Человеческий фактор, часто является причиной снижения оперативности. Поэтому активно развиваются автоматизированные подходы, направленные на повышение уровня оперативности и проактивности в управлении.

Одним из перспективных подходов к решению проблемы является использование интеллектуальных систем, реализующих схему: сбор данных – построение прогностической модели – проактивное принятие решений – выполнение управляющих действий.

Рядом авторов разработаны системы управления с использованием нейросетевых моделей, способствующие эффективному принятию решений [1–4]. Однако, в данных системах для определенных типов внешней нагрузки обучается только одна нейросетевая модель. Предполагается, что эта модель будет успешно прогнозировать значения необходимых параметров и для других (связанных с неопределённостью) типов нагрузки. Выборки для обучения таких моделей готовятся заранее, они, как правило, содержат длительные по времени данные с большим

количеством элементов. Поэтому для эффективного обучения модели (за допустимое время) требуется большое количество высокопроизводительных ресурсов и времени.

В работе предлагается подход, основанный на предположении, что управляемая ИТ система может находиться в различных (по объему вычислительных ресурсов) состояниях. И для каждого состояния (в течение времени его существования), может быть создана и обучена нейросетевая модель, которая способна предсказывать среднее значение утилизации %CPU по вычислительным модулям. На основании этого значения может быть принято управляющее решение, изменяющее состояние системы.

Модельная система критического ИТ сервиса. Для проведения экспериментов авторами была разработана модельная система, концептуально соответствующая модели критичного информационного сервиса [5]. Система состоит из набора вычислительных модулей, на которых функционируют инстансы прикладного программного обеспечения, между которыми осуществляется балансировка внешних запросов. Для генерация запросов используется система, позволяющая задавать нагрузку и получать статистику по обработанным запросам и временам отклика. Модельная система поддерживает механизм масштабирования (изменения состояния). Основные блоки модельной системы представлены на рисунке 1.

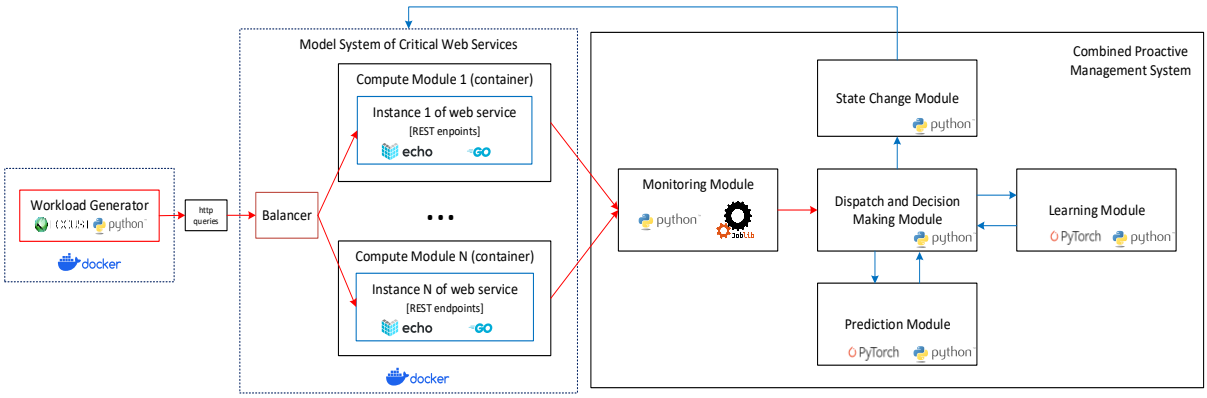

Рисунок 1. – Структурная блок-схема комбинированной системы управления с нейромодулем

Комбинирования система проактивного управления (КСПУ). Архитектурно КСПУ состоит из 5 основных блоков-модулей (см. рисунок 1):

1. Блок мониторинга, который отвечает за сбор метрик производительности вычислительных модулей системы и добавление новых метрик (при изменении состояния системы). Метрики собираются каждые 2 с.

2. Блок изменения состояния, отвечает за отправку управляющих команд (которые изменяют состояние управляемой системы) и контроль корректности изменения состояния.

3. Блок диспетчеризации и принятия решений.

4. Модуль обучения – создает модели нейросетей для набора исторических данных для различных состояний управляемой системы.

5. Модуль прогнозирования – выполняет прогнозы параметров утилизации с определенной заблаговременностью для различных состояний управляемой системы.

Блок диспетчеризации и принятия решений является центральным, в него попадает информация из блока мониторинга. По последним переданным данным производится расчет средних значений по набору вычислительных модулей. Если средние значения по вычислительным модулям превышают пороговые, то принимается решение по изменению состояния управляемой системы и отправляется команда в блок изменения состояния (реактивная компонента).

Параллельно с этим процессом в блоке диспетчеризации и принятия решений происходит накопление исторических данных по вычислительным модулям, которые передаются в модуль обучения. При достижении определенного объема данных, происходит их передача в модуль, в котором выполняется обучение нейросети на этом наборе данных. После получения рабочей модели, блок обучения передает информацию о модели в центральный блок, который берет полученные данные с блока мониторинга и отправляет их и код модели нейросети в блок прогнозирования. Данные механизмы неблокирующего взаимодействия между процессами реализованы с помощью библиотеки multiprocessing [Python]. Более детальное описание технологии и программного алгоритма работы КСПУ представлено в работах [6; 7].

Прогнозирование. Состояние критичной системы связано с объемом вычислительных ресурсов, который требуется для нормального функционирования системы для конкретной нагрузки. Состояние определяется количеством используемых вычислительных модулей. Данные для построения прогностической модели берутся из локальной временной области, соответствующей конкретному состоянию системы. В качестве метрики используется среднее значение %CPU по набору вычислительных модулей.

Время жизни конкретного состояния зависит от характера нагрузки. Переход в другое состояние определяется уровнями измеряемой метрики (уровнями переходов). Задаются максимальный уровень средней утилизации %CPU по вычислительным модулям, при котором система автоматически переходит в новое состояние с большим количеством ресурсов. И минимальный уровень, при котором система переходит в состояние с меньшим количеством ресурсов.

Локальные временные области для различных состояний могут иметь различное количество отсчетов. В это количество входят отсчеты, необходимые для создания прогностической модели. На подстройку параметров модели и обучение так же требуется определенное время.

Из-за неопределенности внешнего воздействия, допустимы резкие скачки нагрузки за характерный временной интервал, который меньше суммарного времени создания модели и подготовки прогноза. В этом случае система изменяет состояние на основе реактивного управления.

Существуют временные ограничения на процесс создания и обучения модели. Желательно, чтобы проактивная компонента, успевала построить адекватную модель и подготовить прогноз, на основе которого будет выполнен переход в новое состояние раньше, чем это произойдет на основе реактивного управления.

В статье [7] авторами для построения прогноза в локальной временной области была сформулирована следующая задача:

Пусть задана критичная система, которая может находиться в конечном множестве состояний $S = \{S_1, S_2, ..., S_n\}$, которые определяются воздействием на систему и уровнями переходов. В каждом состоянии *i* система генерирует дискретный сигнал в виде короткого временного ряда $X_t^{S_i}$ (или набора рядов). Необходимо по части этого сигнала $\tilde{X}_t^{S_i}$ построить предиктор в виде нейронной сети, который прогнозирует переход системы в новое состояние S_{i+1}.

Для решения данной задачи был разработан метод динамической локальной апроксимации нейросетевыми моделями (DLANN). Суть данного метода заключается в том, что в процессе работы системы для каждого её состояния строится простая нейросетевая модель (с одним скрытым и одним выходным слоем), которая учится на части данных локальной временной области. Данный метод послужил основной, для разработки КСПУ.

Для создания и обучения моделей используется библиотека Pytorch [Python]. Процесс обучения должен занимать мало времени, поэтому используется небольшое количество эпох обучения и простая нейронная сеть с одним скрытым слоем (nn.Linear). Функция активации – ReLU, для регуляризации используется отключение части нейронов (dropout). Функция потерь – nn.MSELoss(). Метод оптимизации - torch.optim.Adam, скорость обучения 0.002.

Время обучения нейронной сети с указанной архитектурой для набора данных составляет ~ 2 с. Скорость выполнения предсказания с учетом заблоговременности в 40 отсчетов $\ll 1$ с.

Результаты экспериментов. В качестве примера рассмотрим результаты работы управляющей системы при постепенно возрастающей нагрузке, достигающей 600 пользователей, выполняющих различные запросы примерно за 0,5 часа. В процессе эксперимента система меняла свое состояние 6 раз, при этом проактивные изменения происходили 4 раза (переходы S0→S1, S2→S3, S4→S5, S5→S6), реактивные изменения происходили 2 раза (переходы S1→S2, S3→S4).

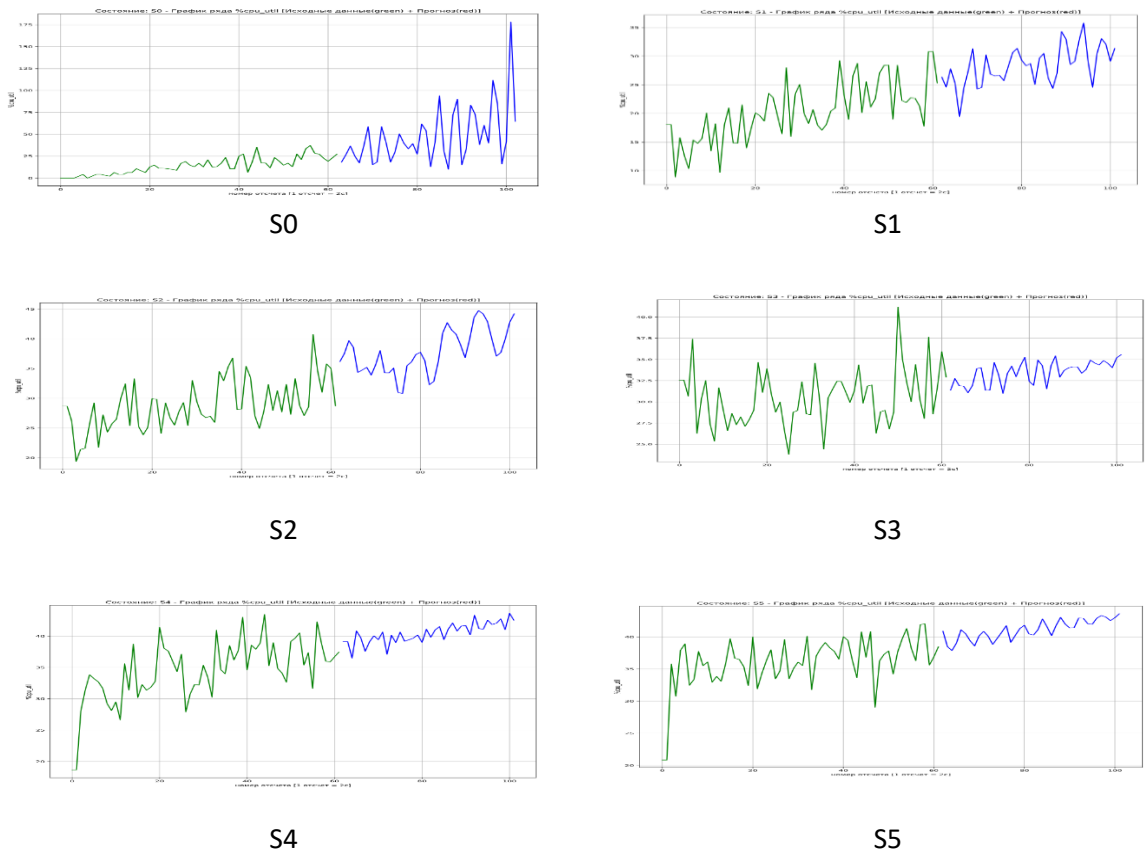

Рисунок 2. – График %CPU (зеленый – исходные данные, синий – прогноз) для состояний S1, S2, S3, S4, S5

Характерным примером является ситуация, полученная для состояния S1 (рисунок 3). Мы видим реактивное изменение состояния, резкий пик около 112 отсчета (модель нейросети проиграла). Было выполнено обучение модели на данных, соответствующих первым 64 отсчетам и был выполнен последний прогноз, но до этого уже была отправлена команда на изменение состояния системы. Эта ситуация показывает, что сложность процесса увеличивается по мере увеличения количества пользователей (числа запросов к системе), модель не учитывает изменение сложности, поскольку сложность (архитектура) самой модели не меняется.

Увеличение нагрузки (сложности сигнала) приводит к тому, что модель выполняет обобщение сигнала и теряет возможности в воспроспроизведении более сложной нерегулярной структуры. Модель ведет себя подобно фильтру, который обобщающает сигнал в тренд. Этот эффект хорошо заметен, если рассмотреть прогнозы моделей для состояний, когда приближаемся к максимальному количеству пользователей для системы.

Дальнейший анализ показал, что для повышения качества и точности прогнозирования временного ряда, требуется создавать нейронные сети с более сложной архитектурой. Расплачиваться за усложнение архитектуры нейронной сети приходится за счет увеличения времени обучения, снижая оперативность принятия

решений. Усложнение модели (увеличение количества весов) само по себе требует более ювелирного подбора гиперпараметров и может приводить к худшим результатам и переобучению.

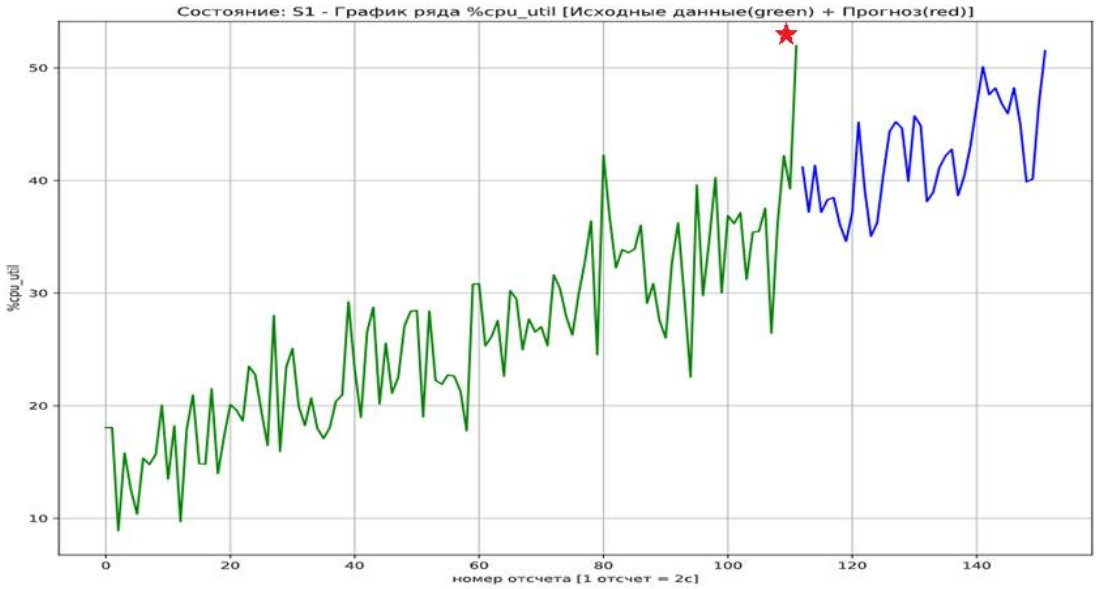

Рисунок 3. – График %CPU (зеленый – исходные данные, синий – прогноз) для состояний S1

Заключение. Результаты экспериментов показали, что данный подход можно использовать в практических задачах. А модель, обученная на части данных, может делать прогнозы для всей локальной области. Появляется возможность создания библиотеки нейронных моделей, способных выполнять прогнозы для различных состояний системы, выполнять дальнейшее закономерное усложнение предиктора за счет использования ансамблей моделей. Для точного прогнозирования сложно устроенного сигнала необходимо усложнение архитектуры предиктора. Более сложная архитектура, которая потенциально способна к более точным предсказаниям требует увеличения времени обучения, что снижает оперативность принятия решений. Эта ситуация наглядно показывает, что подготовка более точного прогноза для сложного сигнала требует не только улучшения алгоритма, но и для обеспечения оперативности принятия решений более производительных вычислительных ресурсов.

ЛИТЕРАТУРА

1. M. Straesser, J. Grohmann, J. von Kistowski, S. Eismann, A. Bauer, S. Kounev. Why Is It Not Solved Yet?: Challenges for Production-Ready Autoscaling // In ICPE '22: ACM/SPEC International Conference on Performance Engineering, Bejing, China, April 9 – 13, 2022, p. 105–115, ACM, 2022.
2. N. Khan, D. A. Elizondo, L. Deka, M. A. M.–Cabello. Fuzzy Logic applied to Sys-tem Monitors // IEEE Access, Vol. 9, p. 56523–56538, 2021.

3. B. M. Nguyen, G. Nguyen, Giang. A Proactive Cloud Scaling Model Based on Fuzzy Time Series and SLA Awareness // Procedia Computer Science (International Conference on Computational Science ICCS 2017), 108, p. 365–374, 2017.
4. V. Persico, D. Grimaldi, A. Pescape, A. Salvi, S. Santini. A Fuzzy Approach Based on Heterogeneous Metrics for Scaling Out Public Clouds // IEEE Transactions on Parallel and Distributed Systems, Vol. 28, No. 8, p. 2117–2130, 2017.
5. Старовойтов, А.А. Моделирование систем для проактивного управления вычислительными комплексами // XXV Всероссийская студенческая научно-практическая конференция Нижневартовского государственного университета (г. Нижневартовск, 4–5 апреля 2023 г.) / Под общей ред. Д.А. Погонышева. Ч. 3. Информационные технологии. – Нижневартовск: НВГУ, 2023. – 148 с.
6. Старовойтов, А.А. Алгоритм проактивного управления вычислительными ресурсами // 80-я научная конференция студентов и аспирантов Белорусского государственного университета [Электронный ресурс]: материалы конф., Минск, 10–20 марта 2023 г. В 3 ч. Ч. 1 / Белорус. гос. ун-т; редкол.: А. В. Блохин (гл. ред.) [и др.]. – Минск: БГУ, 2023. – 398 с.
7. Starovoytov, A. A. Technology for making real-time decisions based on neural network forecasting / A. A. Starovoytov, V. V. Krasnoproshin // Pattern Recognition and Information Processing (PRIP'2023) = Распознавание образов и обработка информации (2023) : Proceedings of the 16th International Conference, October 17–19, 2023, Minsk, Belarus / United Institute of Informatics Problems of the National Academy of Sciences of Belarus. – Minsk, 2023. – P. 58–63.