

Децентрализованный алгоритм управления конвейерной системой с использованием методов мультиагентного обучения с подкреплением

Мухутдинов Дмитрий, группа М4239

Научный руководитель: Фильченков А. А., к.ф.-м.н., доцент ФИТиП

Консультант: Вяткин В.В., д.т.н., профессор ФИТиП

Факультет Информационных Технологий и Программирования
Мегафакультет Трансляционных Информационных Технологий
Университет ИТМО, Санкт-Петербург

22 мая 2019 г.

Применения:

- Промышленность
- Сортировка грузов
- Распределение багажа
- ...

Будем рассматривать случай с багажом.

- Статические стратегии управления¹
 - Разрабатываются под конкретную топологию системы
- Model predictive control (MPC)²³⁴
 - Решает глобальную оптимизационную задачу в форме LP/QP
 - Вся система контролируется централизованно
 - Не может обрабатывать изменения в системе, не заложенные в модель (поломки)
 - Не найдено алгоритма именно для багажных ленточных конвейеров
- Маршрутизация по аналогии с компьютерными сетями⁵
 - Децентрализованное вычисление, устойчивость к поломкам
 - Оптимизирует только скорость доставки чемоданов

¹De Neufville.

²Cataldo, Scattolini.

³Zeinaly, De Schutter, Hellendoorn.

⁴Luo, Huang, Zhang.

⁵Yan, Vyatkin.

Разработать алгоритм управления конвейерной системой со следующими свойствами:

- Децентрализованность
- Многокритериальная оптимизация (время доставки багажа + энергопотребление)
- Устойчивость к разнородным изменениям в условиях среды
 - Изменения характеристик багажного потока
 - Поломки конвейеров

- Сосредоточимся на обобщенной задаче маршрутизации в ориентированном графе
- Обучение с подкреплением
- Нейросети в качестве обучающихся агентов
- Q-routing⁶
 - Не получил широкого распространения в компьютерных сетях (использует слишком много служебных пакетов)
 - В конвейерных сетях это не является проблемой.

⁶Boyan, Littman.

Постановка задачи в терминах RL

- Рассмотрим *пакет* в сети как обучающегося агента, взаимодействующего с сетью как со средой
- Полное состояние среды неизвестно, состояние текущего роутера — *наблюдение* пакета
- Действие — переход к одному из соседей
- Q-learning:

$$Q(o_t, a_t) \leftarrow r_t + \gamma \cdot \max_{a \in \mathcal{A}_{o_{t+1}}} Q(o_{t+1}, a)$$

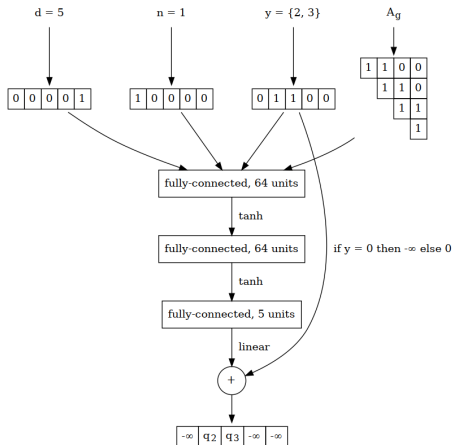
- Принцип аналогичный Q-routing:

$$Q_x(d, y) \leftarrow (t_{finish} - t_{start}) + \max_{z \in \{V | (y, z) \in E\}} Q_y(d, z)$$

Что было сделано: алгоритм DQN-routing⁷

- Объединяет link-state протокол с алгоритмом Q-routing
- Вход нейросети:
 - Номер текущего узла n
 - Номер узла назначения d
 - Узлы-соседи Y
 - Матрица смежности графа
- Выход: $\{Q_x(d, y)\}_{y \in Y}$

Был протестирован в сеттинге компьютерных и конвейерных сетей



⁷“Multi-agent deep learning for simultaneous optimization for time and energy in distributed routing system”.

- Плюсы

- Адаптация под изменения трафика
- Адаптация после поломок
- Оптимизация времени доставки и энергопотребления с заданным приоритетом

- Минусы

- Размер выходного слоя линейно зависит от размера графа
- Размер входного слоя квадратично (!) зависит от размера графа
- Требуется предварительного обучения с учителем

Можно ли избавиться от минусов, не потеряв плюсов?

Идеи усовершенствования алгоритма

- Предсказание Q-функции отдельно для каждого исходящего ребра
 - Вместо множества текущих соседей подаем на вход одного соседа y .
 - Один нейрон на выходном слое выдает скалярное значение $Q_x(d, y)$
- Использование графовых эмбедингов
 - Вместо кодирования меток узлов унитарным кодом использовать их отображения в векторное пространство фиксированной размерности
 - Отказ от подачи на вход матрицы смежности
 - Вместо этого пересчитывать эмбединги при изменении топологии
 - Эмбединги косвенно передадут информацию о топологии

Модификация: DQN-LE-routing

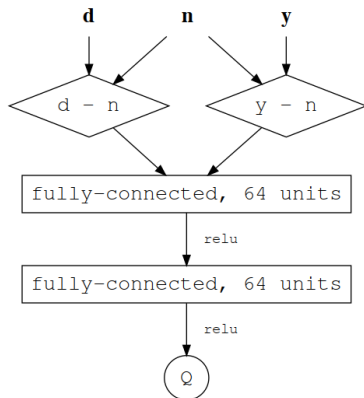
- Получаем эмбединги методом Laplacian Eigenmaps (LE)
- Нормализуем веса ребер по среднему перед расчетом
- Полученные эмбединги домножаем на средний вес

Тогда

$$Q_n(d, y) = f_{\theta}((LE_G(d) - LE_G(n)) \odot (LE_G(y) - LE_G(n)))$$

, где:

- $LE_G(\cdot)$ возвращает эмбединг по номеру узла
- $f_{\theta}(\cdot)$ — feed-forward нейронная сеть



- Запуск экспериментов в симуляторе
- Размерность эмбединга: 8
- Бейзлайны:
 - Табличный Q-routing
 - Дейкстра с протоколом link-state (shortest paths, SP)
 - Оригинальный DQN-routing (DQN)
- Служебные сообщения доставляются бесплатно
- Зачем?
 - Быстрее симулировать, чем конвейеры
 - Показываем универсальность алгоритма

Эксперименты в модели компьютерной сети: предобучение

- У нетабличного Q-обучения нет гарантий сходимости (2)
- Experience replay не работает в нестационарной среде
- Предобучаем на действиях алгоритма shortest paths
- Получаем данные на базовом графе (1) и его модификациях

Рис. 1: Базовая топология сети для тестов

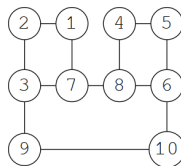
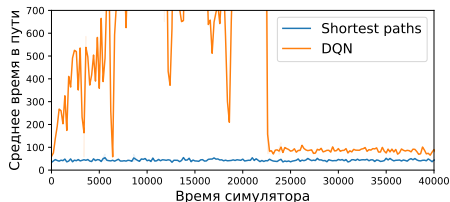
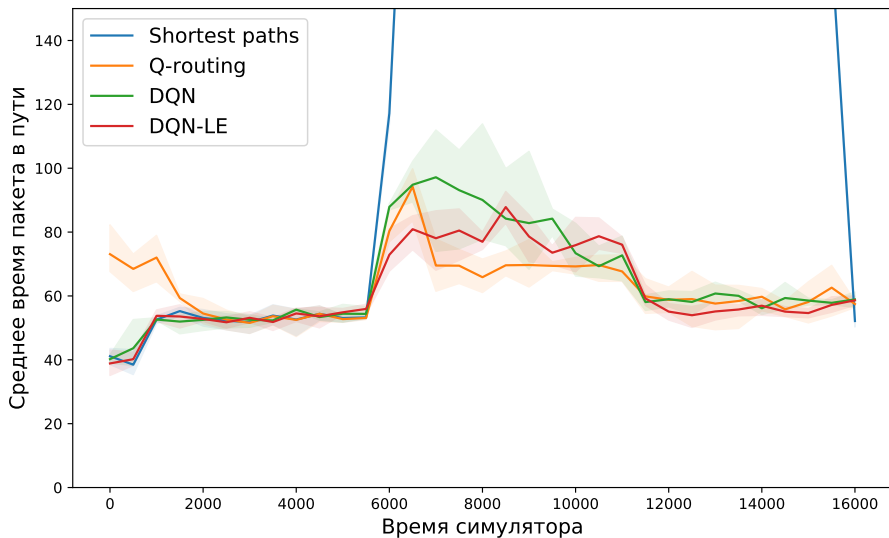


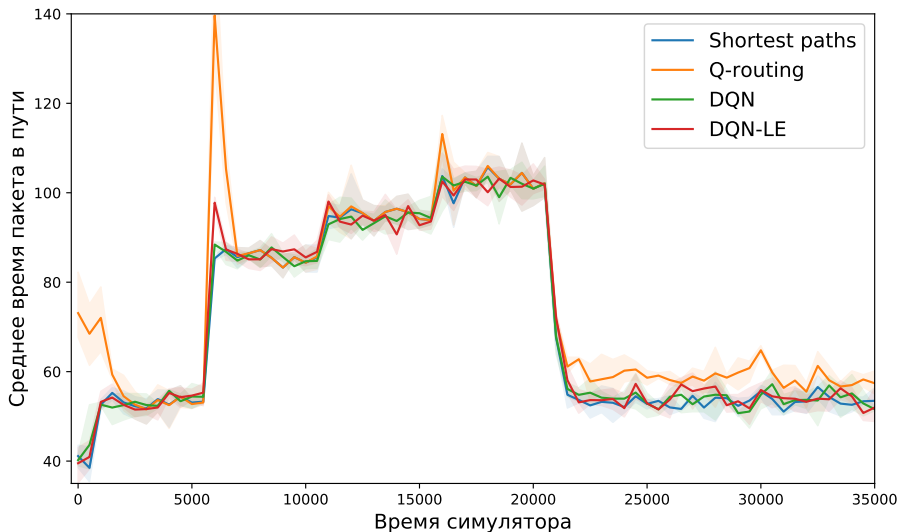
Рис. 2: Работа DQN без предобучения



Эксперименты: резкое изменение нагрузки

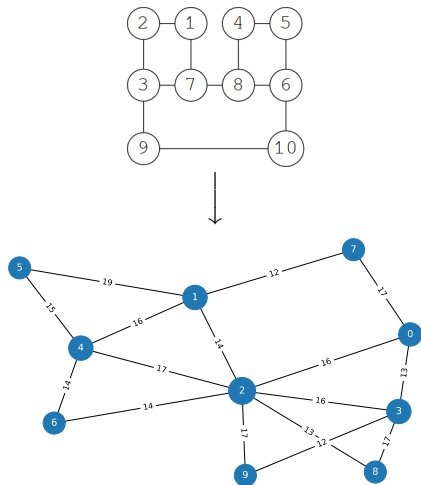


Эксперименты: обрыв и восстановление соединений

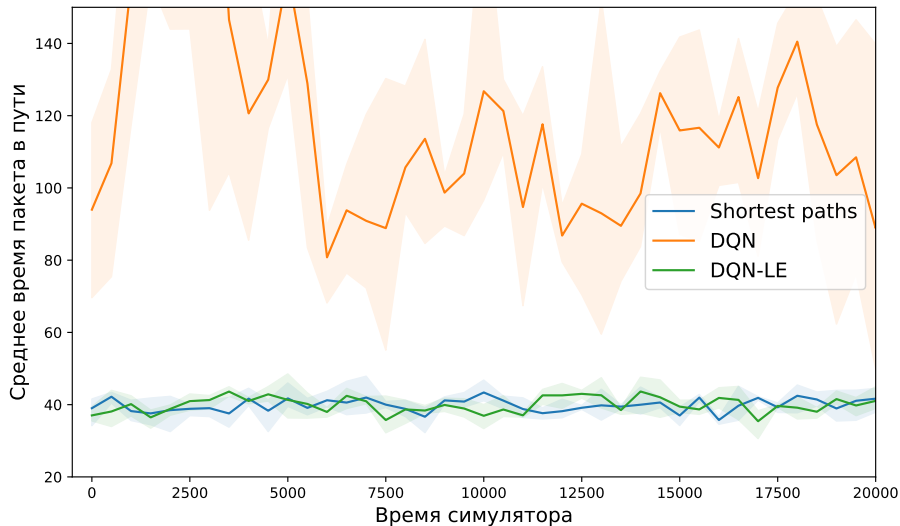


Перенос опыта на новую топологию

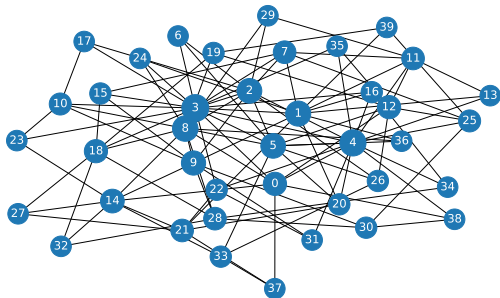
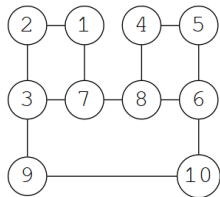
- DQN-LE-routing все еще требует предобучения
- Однако, если опыт переносится на совершенно новые топологии, это не страшно
- Проверим производительность на случайном графе



Перенос опыта на новую топологию: то же кол-во вершин



Перенос опыта на новую топологию: большее кол-во вершин



Перенос опыта на новую топологию: большее кол-во вершин

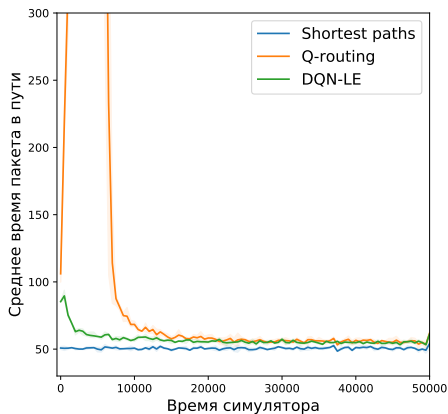


Рис. 3: Низкая нагрузка

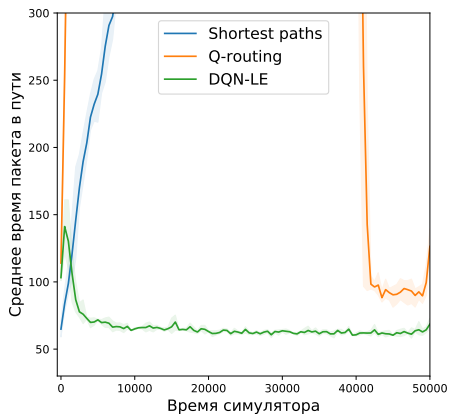


Рис. 4: Высокая нагрузка

Эксперименты в модели системы багажных конвейеров

- Конвейерная сеть моделируется как ориентированный граф
- Каждая секция конвейера – отдельная вершина
- В оптимизируемую функцию включено *энергопотребление*
- На вход нейросети дополнительно подается информация о состоянии конвейеров
- Важность экономии энергии регулируется параметром α
- Размерность эмбединга: 10

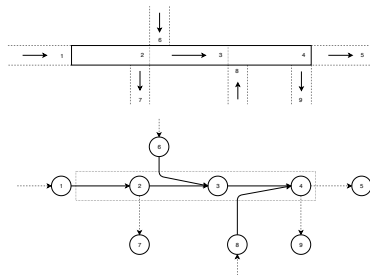
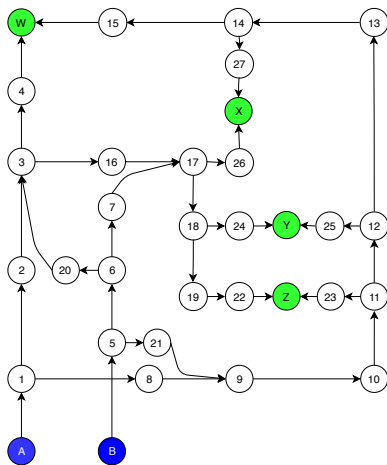
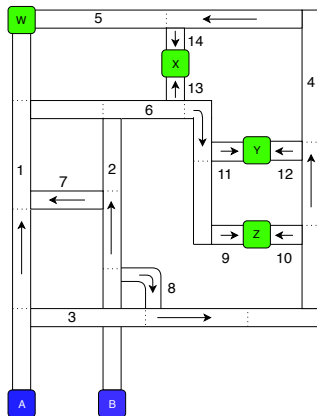
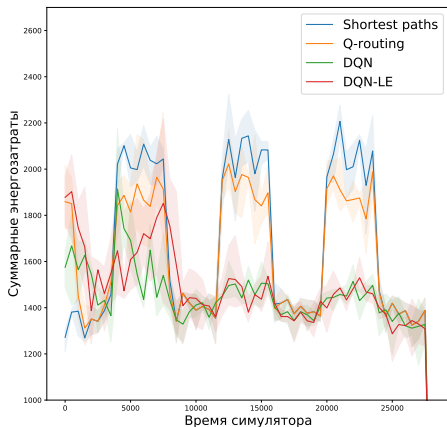
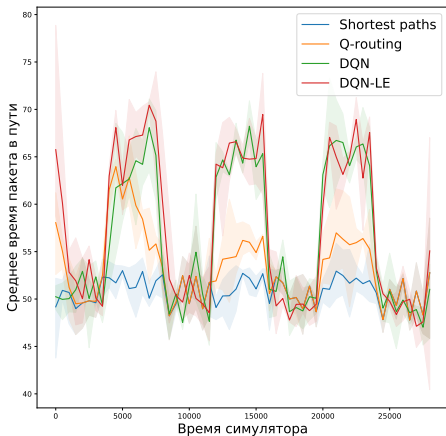


Рис. 5: Участок конвейерной сети и соответствующий участок графа

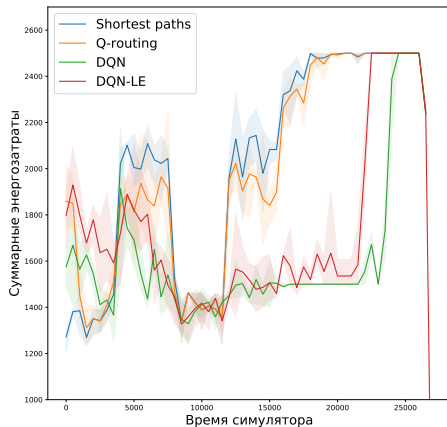
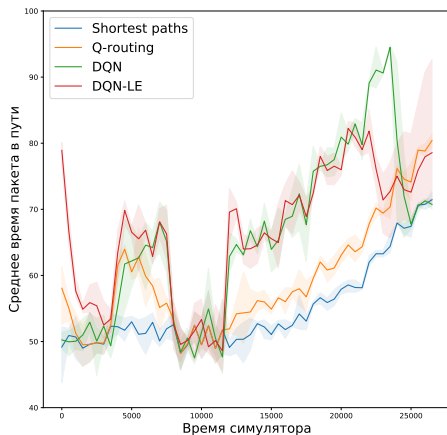
Модель конвейерной системы для проведения экспериментов



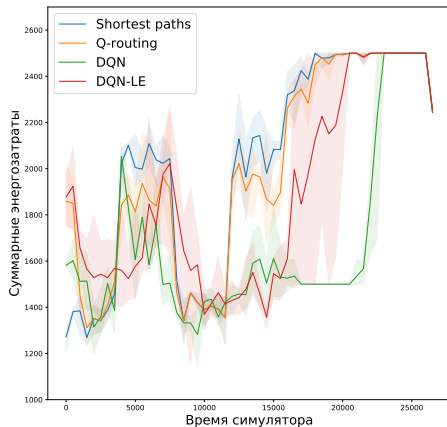
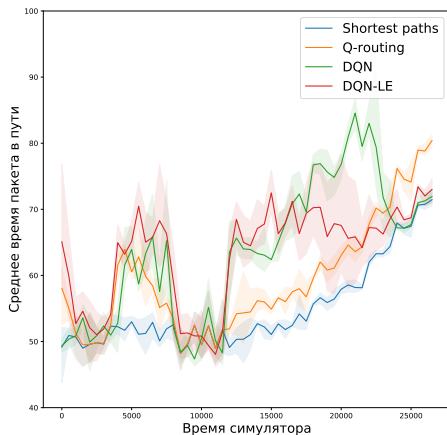
Неравномерный поток до выходных вершин



Плавное повышение нагрузки: $\alpha = 1$, $\alpha_{LE} = 0.4$



Плавное повышение нагрузки: $\alpha = 0.8$, $\alpha_{LE} = 0.2$



- Разработана модификация алгоритма DQN-routing — DQN-LE-routing
- Модифицированный алгоритм свободен от ключевых недостатков DQN-routing:
 - Зависимость размера нейронной сети от размера графа
 - Необходимость переобучения с учителем на новых графах
- Не уступает изначальному алгоритму по качеству работы в модели компьютерной или конвейерной сети

Направления дальнейших исследований

- Поиск/изобретение более качественных бейзлайнов (например, на основе MPC)
- Исследование идеи глобальной графовой нейронной сети⁸⁹¹⁰
- Проведение большего числа экспериментов на различных топологиях
 - В т. ч. случайная генерация правдоподобных конвейерных сетей
- Выход за пределы маршрутизации
 - Обучение контроллера скорости конвейера

⁸“The graph neural network model”.

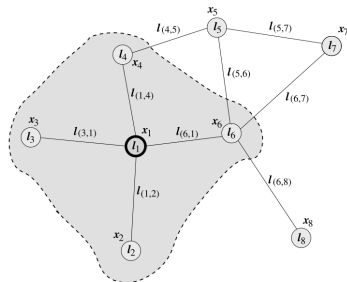
⁹“Gated graph sequence neural networks”.

¹⁰Geyer, Carle.

Спасибо за внимание!

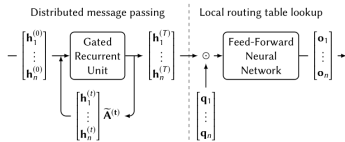
Глобальная графовая нейронная сеть: идея

- Рассматриваем весь граф с состояниями ребер и узлов как вход для графовой нейронной сети (GG-NN)
- Считается распределенно на физических узлах системы
- Промежуточные состояния и их градиенты передаются между узлами по сети
 - Применяется для компьютерных сетей с обучением с учителем¹¹
 - Добавим обучение с подкреплением во время работы



$$x_1 = f_w(l_1, l_{(1,2)}, l_{(3,1)}, l_{(1,4)}, l_{(6,1)}, x_2, x_3, x_4, x_6, l_2, l_3, l_4, l_6)$$

$l_{co[1]} \quad x_{ne[1]} \quad l_{ne[n]}$



¹¹Geyer, Carle.