

# Глубокие самообучающиеся агенты в мультиагентной системе маршрутизации

Мухутдинов Дмитрий, группа М3438

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

Рецензент: Тарасов В. Б., к.т.н., МГТУ им. Баумана

Кафедра Компьютерных Технологий  
Факультет Информационных Технологий и Программирования  
Университет ИТМО, Санкт-Петербург

6 июня 2017 г.

# Задача маршрутизации

- Сетевой роутинг
- Транспортная логистика
- Управление конвейерными системами
- Автоматическое управление городским трафиком

- Link-state
  - Open Shortest Path First (OSPF)
  - IS-IS
- Distance-vector
  - RIP
  - IGRP
- Прочие
  - AntNet
  - ...

- Примерно все алгоритмы маршрутизации заточены под компьютерные сети
- В других задачах существуют свои, более сложные условия
  - Скорую нужно пропустить сквозь пробку, а обычный автомобиль — нет
  - Чемоданы бизнес-класса хочется доставить первыми
  - ...

Построить алгоритм, способный адаптироваться под гетерогенные условия

- Обучение с подкреплением
- Нейросети в качестве обучающихся агентов
- Q-routing (Boyan & Littman, 1994)

# 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:

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

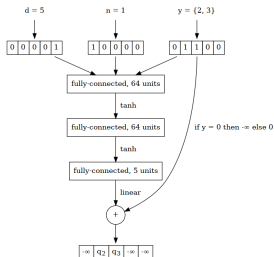
Вход нейросети  $o = (d, n, y_1 \dots y_n, o')$ , где:

- $d$  — узел назначения,  $s$  — текущий узел,  $y_1 \dots y_n$  — номера соседей,  $o'$  — любая дополнительная информация
- Выходы нейросети  $a_1 \dots a_n$  — оценки  $Q(o, a)$  для всех узлов сети ( $-\infty$  для узлов, не являющихся соседями)
- Кодировем номера унитарным кодом, чтобы избежать корреляции результатов
- Используем RMSProp для оптимизации

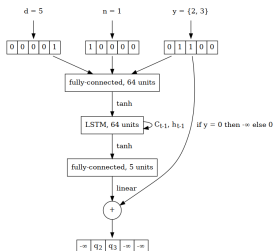


# Варианты архитектуры нейросети

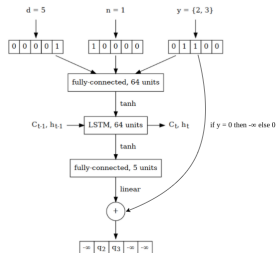
Feed-forward



LSTM с "памятью маршрута"



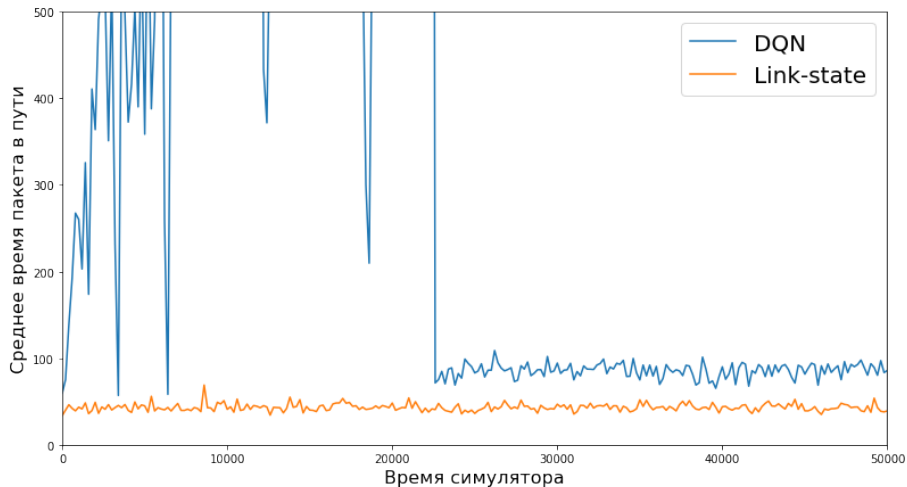
LSTM с "памятью пакета"



# Проблема нестационарности

- Q-обучение, вообще говоря, не сходится к оптимальному решению в случае бесконечного числа состояний
- Другие агенты меняют свое поведение — среда нестационарна, вследствие чего experience replay (Mnih et al., 2015) не работает
- При обучении с нуля на равномерной низкой нагрузке нейросети не могут найти оптимальные пути

# Проблема нестационарности

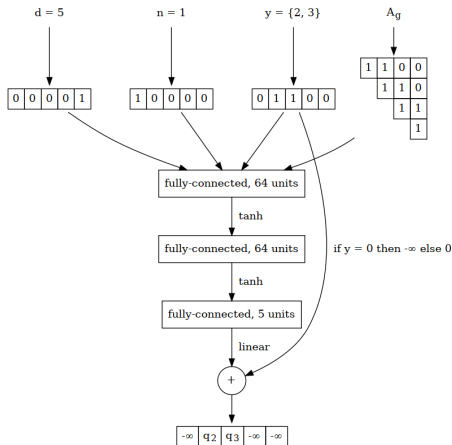


- Предобучение сети (bootstrapping)
  - Собираем данные работы алгоритма кратчайших путей
  - Обучаем одну нейросеть на данных от всех роутеров
  - Предобученная сеть повторяет работу алгоритма кратчайших путей
- Отказ от experience replay во время работы
  - Условия работы в сети меняются
  - Старый опыт перестает быть актуальным
  - Если обращать на него внимание, адаптивность к изменяющимся условиям страдает

# Расширение наблюдаемого состояния

Добавим на вход дополнительную информацию

- Топологию сети полезно учитывать
- Используем link-state протокол для обновления информации о топологии
- Подаем матрицу смежности графа на вход



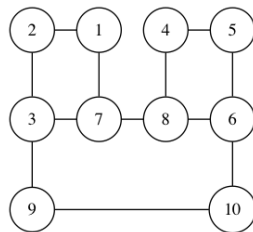
- Алгоритм может застревать в локальном максимуме
- Сложности к возвращению к оптимальной стратегии после изменений условий среды
- Решение: используем softmax-стратегию, чтобы иметь шанс выбрать не максимально “выгодное” действие:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \text{ for } j = 1 \dots K$$

$$P(a \mid o) = \sigma(Q(o, a))$$

# Эксперименты в модели компьютерной сети

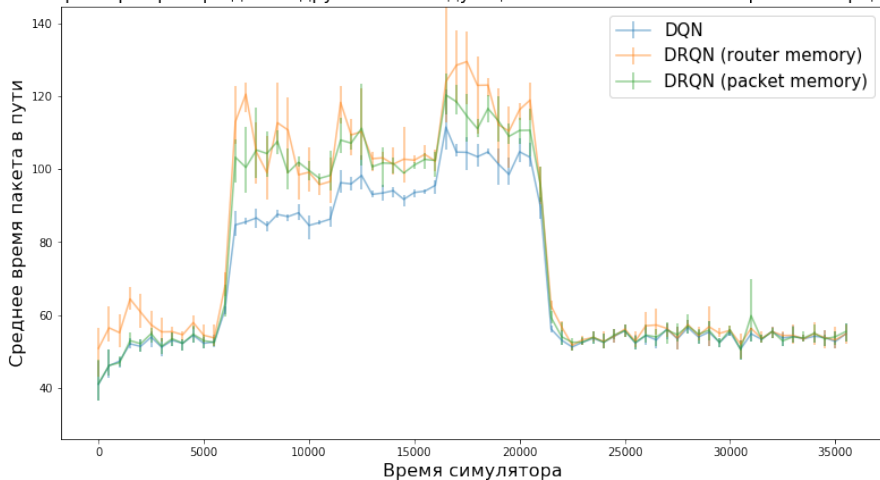
- Запуск экспериментов в симуляторе
- Сравнение архитектур между собой и сравнение с алгоритмами link-state и Q-routing
- Служебные сообщения доставляются бесплатно



Топология сети для тестов

# Изменение топологии сети: сравнение архитектур нейросетей

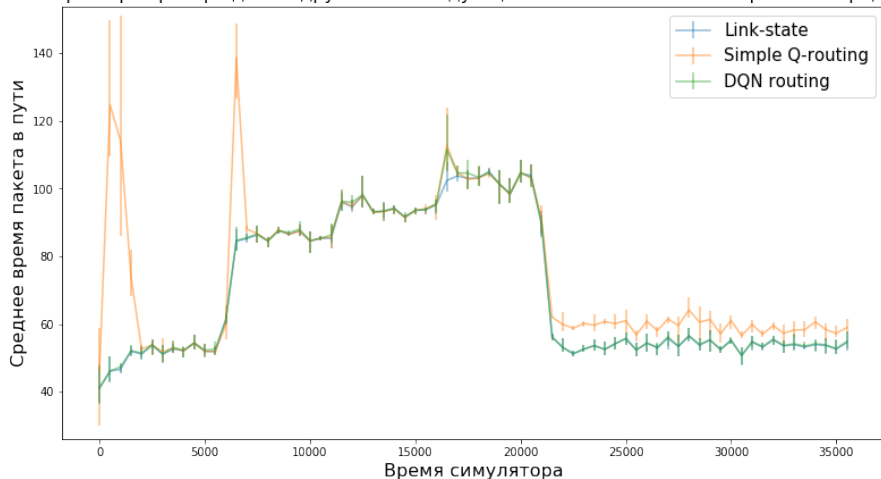
Обрыв трех ребер одно за другим с последующим восстановлением в обратном порядке





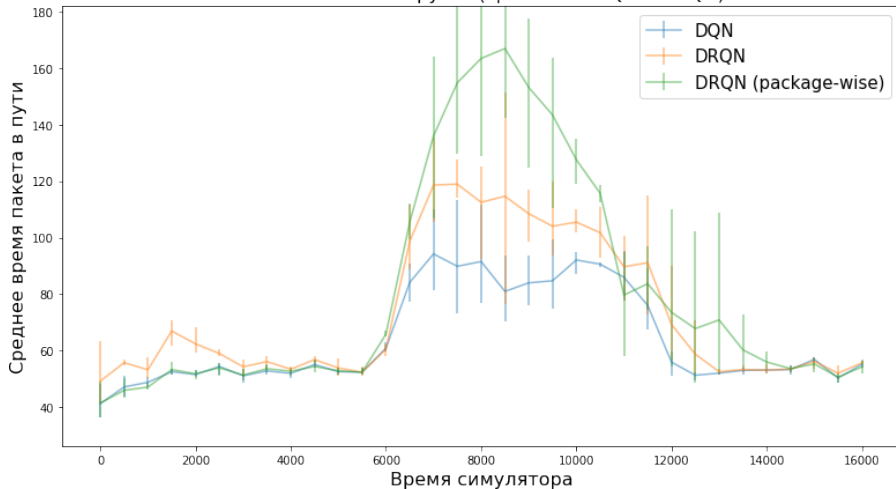
# Изменение топологии сети: сравнение с link-state и Q-routing

Обрыв трех ребер одно за другим с последующим восстановлением в обратном порядке

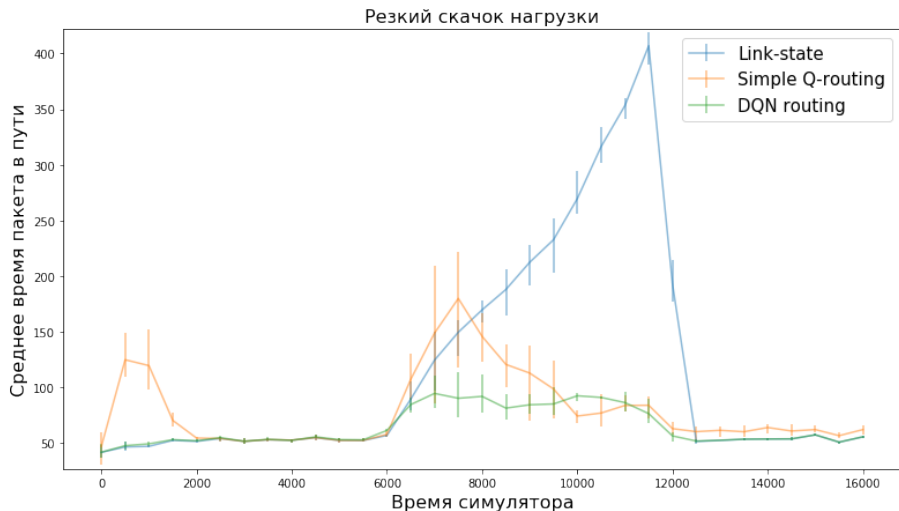


# Изменение нагрузки: сравнение с архитектур нейросетей

Резкий скачок нагрузки (сравнение DQN и DRQN)

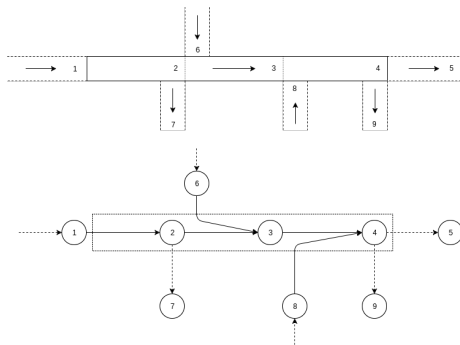


# Изменение нагрузки: сравнение с link-state и Q-routing



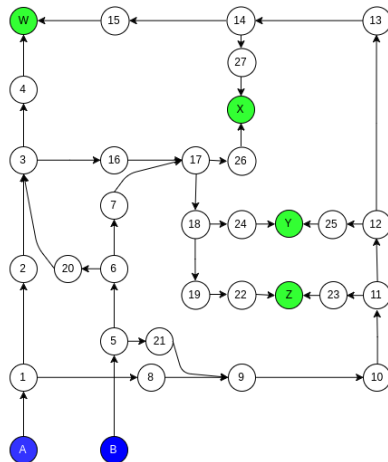
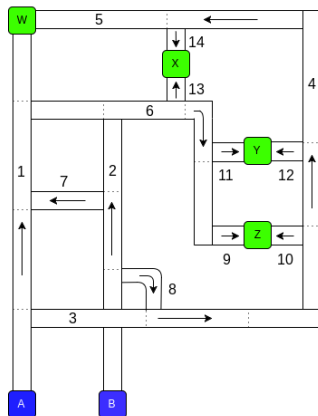
# Эксперименты в модели системы багажных конвейеров

- Конвейерная сеть моделируется как ориентированный граф
- Каждая секция конвейера – отдельная вершина
- Отдельно выделены входные и выходные вершины
- В оптимизируемую функцию включено *энергопотребление*
- Важность экономии энергии регулируется параметром  $\alpha$



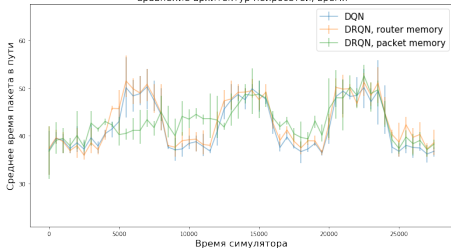
Участок конвейерной сети и соответствующий участок графа

# Модель конвейерной системы для проведения экспериментов

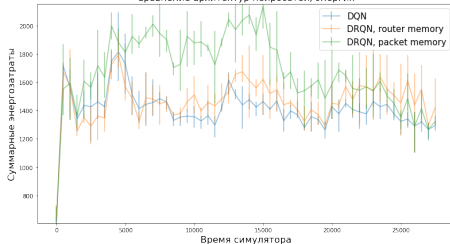


# Неравномерный поток до выходных вершин: сравнение архитектур нейросетей

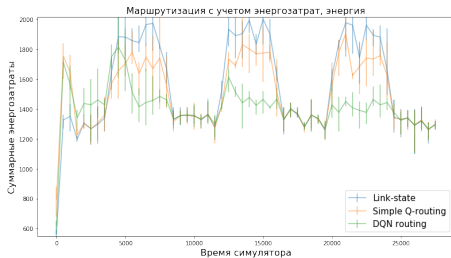
Сравнение архитектур нейросетей, время



Сравнение архитектур нейросетей, энергия

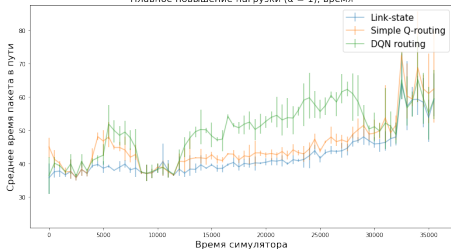


# Неравномерный поток до выходных вершин: сравнение с link-state и Q-routing

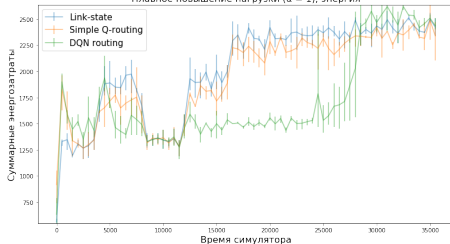


# Плавное повышение нагрузки: $\alpha = 1$

Плавное повышение нагрузки ( $\alpha = 1$ ), время



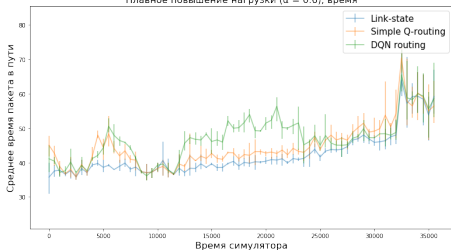
Плавное повышение нагрузки ( $\alpha = 1$ ), энергия



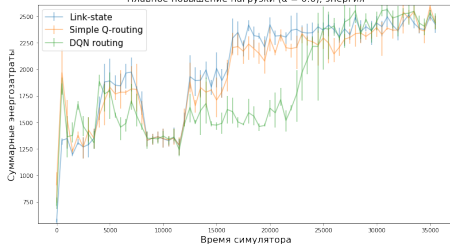


# Плавное повышение нагрузки: $\alpha = 0.6$

Плавное повышение нагрузки ( $\alpha = 0.6$ ), время



Плавное повышение нагрузки ( $\alpha = 0.6$ ), энергия



Спасибо за внимание!