КЛАСТЕРИЗАЦИЯ КАК МЕТОД ВЫБОРА ЛИДЕРОВ В ПРОТОКОЛАХ КОНСЕНСУСА

Басимова Н.Ф.

Сколковский институт науки и технологий,
ул. Большой бульвар д.30, стр.1, г. Москва, Россия
basimova.nf@phystech.edu,

Чеботарев П.Ю.

Институт проблем управления им. В.А. Трапезникова РАН,
Россия, г. Москва, ул. Профсоюзная д.65
Институт радиотехники и электроники имени В. А. Котельникова РАН,
ул. Моховая д.11, корп. 7, г. Москва, Россия
pavel4e@gmail.com

Аннотация: В схеме «лидер-последователь» лидеры имеют автономную динамику, а последователи подчиняются сетевому протоколу. Эта схема связывает сетевые агентные модели с классическими моделями управления. В работе показано, что в задаче выбора лидеров для протокола консенсуса кластеризация – наиболее точный из известных быстрых алгоритмов.

Ключевые слова: многоагентная система, линейный протокол консенсуса, выбор лидеров, кластеризация

Введение

Многоагентная система (МАС) – это система, состоящая из нескольких взаимодействующих агентов. Такие системы могут использоваться для решения задач, которые сложно или невозможно решить с помощью одного агента или монолитной системы. Важный пример таких задач – совместное движение мобильных роботов. В целом будущее, очевидно, за гетерогенными командами, включающими людей, роботов и программные агенты; миссии подобных команд могут быть чрезвычайно разнообразны, включая, например, устранение аварий и реагирование на чрезвычайные ситуации [1]. Многоагентные системы используются также для моделирования социальных [2] и многих других структур.

Перемещение нескольких мобильных агентов к заданной целевой точке при поддержании взаимосвязи между их состояниями – важная проблема робототехники. В приложениях, связанных с совместным движением беспилотных летательных аппаратов (БПЛА), колесных мобильных роботов и т. п. агентам, как правило, необходимо обеспечивать движение к цели с выстраиванием, сохранением и типовой модификацией геометрической формы своего формирования. Схема «лидер-последователь» является одним из элементов решения такого рода задач управления формациями [3].

За последние годы предложено несколько методов решения задач выбора лидеров в сетевых многоагентных системах, в частности, применительно к протоколам линейного консенсуса. Один из простейших методов – выбор в качестве лидеров агентов, которым соответствуют вершины графа коммуникаций, имеющие высокие степени [4]. Динамически задача выбора лидеров может решаться либо с возрастанием группы лидеров (выбор одного лидера и добавление других до тех пор, пока набор лидеров не станет оптимальным) [5], либо с ее убыванием: оптимальный набор лидеров ищется путем последовательного исключения агентов из полного множества агентов [6]. Для управления БПЛА применялась, в частности, смена лидера в реальном времени с целью оптимизации скорости сближения позиций агентов [7].

Существует несомненная семантическая связь между задачей выбора лидеров и задачей кластеризации узлов графа. Идея использования кластеризации для выбора лидеров состоит в том, что каждый лидер рассматривается как агент, ответственный за свой кластер [8]. Поэтому представляет интерес задача сравнения существующих алгоритмов выбора лидера с подходом, основанным на кластеризации агентов с последующим выбором «основного» узла в каждом кластере.

Среди популярных методов кластеризации узлов можно назвать спектральную кластеризацию [9] и метод k-средних, минимизирующий общее квадратичное отклонение элементов кластеров от их центров [10].

В данной работе рассматривается проблема выбора лидеров в простейшем случае, когда они сохраняют свои исходные состояния. Проблема выбора мобильных лидеров изучалась, например, в [11]. Задача выбора лидеров рассматривается в работе в двух постановках. Первая – выбор в «идеальных» условиях, т.е. при отсутствии внешних ограничений, а второй – моделирование в реальных условиях с использованием модели мобильных роботов e-puck2 [12]. «Реальные условия»

означают наложение физических ограничений, таких, как ограничения на скорость роботов. Целью является сравнение известных алгоритмов выбора лидеров с методом, основанным на кластеризации узлов графа. Критериями при этом служат точность и время сходимости, как в идеальных, так и в реальных условиях.

1 Модель

Рассматривается многоагентная система, структура которой представлена ориентированным графом зависимостей агентов $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Таким образом узел v_i, $i = 1, \dots, V$, орграфа $\mathcal{G}$ отождествляется с i-м агентом. Если агент i зависит от агента j, то в орграфе $\mathcal{G}$ проводится направленная дуга $e = (v_i, v_j) \in \mathcal{E}$. Каждая дуга $e \in \mathcal{E}$ имеет свой вес w; эти веса являются элементами взвешенной матрицы смежности $A = (a_{ij})$ системы [13].

Пусть L – лапласовская матрица [14] орграфа $\mathcal{G}$ с матрицей смежности A: $L = \mathrm{diag}(A1) - A$, где 1 – вектор из единиц. Каждый агент i имеет свое состояние x_i. В простейшей интерпретации x_i – координата (вектор координат) агента i. Протокол линейного консенсуса имеет вид

$$\dot{x} = -Lx, \qquad (1)$$

где $x = (x_1, \dots, x_V)$ – вектор состояния системы.

Лидеры рассматриваются в данном случае как агенты, не меняющие с течением времени своих начальных состояний. Разделим общий вектор состояния системы на подвекторы, задающие состояние лидеров (x_L) и состояние последователей (x_F). В этих обозначениях протокол линейного консенсуса (1) можно переписать следующим образом:

$$\begin{bmatrix} \dot{x_F} \\ \dot{x_L} \end{bmatrix} = \begin{pmatrix} L_{FF} & L_{FL} \\ L_{LF} & L_{LL} \end{pmatrix} \begin{bmatrix} x_F \\ x_L \end{bmatrix}. \qquad (2)$$

Лидеры сохраняют свои состояния, поэтому матрицы, задающие влияния на лидеров, L_{LL} и L_{LF} равны нулю. В силу этого $\dot{x}_L = 0$ и x_L постоянно. Тогда «внутренний» лапласиан системы L_{FF} [15] задает связи между последователями. В данной работе предполагается, что матрица L_{FF} симметрична: любые два последователя влияют друг на друга с равной силой. Асимптотическая скорость сходимости системы может быть оценена посредством наименьшего ненулевого (фидлеровского) собственного значения внутреннего лапласиана L_{FF} [5], поэтому в дальнейшем, говоря о скорости, будем подразумевать именно это собственное значение. Для целей моделирования время сходимости определим следующим образом.

Определение 1. Пусть x – предельное состояние системы. Система достигает x с фиксированной ошибкой e за время сходимости t_e, если t_e – наименьшее число, для которого

$$\|x(t_+) - x^*\|_2 \le e \quad \forall t_+ \ge t_e, \qquad (3)$$

где $\|y\|_2$ – евклидова норма вектора y.

2 Алгоритмы

Далее рассмотрим шесть алгоритмов выбора лидеров и оценим их алгоритмическую сложность, после чего включим в рассмотрение два дополнительных алгоритма, представленных в [16], [17]. Алгоритм работы [16] основан на теоретических расчетах, устанавливающих его оптимальность, однако его алгоритмическая сложность выше, чем у оптимизированной реализации жадного алгоритма из [17], сложность которого, в свою очередь, выше, чем сложность алгоритма k-средних. Перечислим указанные шесть алгоритмов.

1) Алгоритм выбора k лидеров спектральным методом. Его описание в псевдокоде приведено в [5]. Оценка алгоритмической сложности алгоритма – не менее $O(kn^3)$ – определяется необходимостью вычислять собственные значения лапласовской матрицы и k раз выбирать наименьшее собственное значение.

2) Случайный выбор лидера. Случайным образом без повторов генерируются k чисел, равномерно распределенных от 1 до V; соответствующие агенты выбираются в качестве лидеров. Алгоритмическая сложность: $O(n)$.

3) Выбор k лидеров с максимальной степенью захода в орграфе зависимостей. Алгоритмическая сложность составляет $O(n^2)$, т.к. вычисляется степень каждого узла.

4) Выбор лидеров со средними степенями. В орграфе зависимостей выбираются k узлов со средними степенями захода. Алгоритмическая сложность также $O(n^2)$.

5) Алгоритм k-средних. Граф зависимостей разбивается на k кластеров с помощью алгоритма k-средних. Затем ближайший к центру кластера узел выбирается в качестве лидера в этом кластере. Если этот узел уже выбран в качестве лидера для другого кластера, то выбирается второй ближайший узел и так далее. Используется реализация метода k-средних из библиотеки Python sklearn с параметрами по умолчанию; его средняя алгоритмическая сложность $O(kn)$, наивысшая сложность $O(n^{k+1})$ [18]. Однако наивысшая сложность не достигается благодаря фиксированию максимального количества итераций.

6) «Большой» случайный выбор. Сначала с использованием равномерного распределения выбираются 10000 чисел от 1 до C_n^k, каждое из которых служит индексом в списке, в котором все возможные выборки k лидеров расположены в лексикографическом порядке; при этом индекс однозначно определяет выбор. Получаем 10000 пробных решений. Алгоритм перебирает их и находит набор(ы) с максимальным фидлеровским собственным значением «внутреннего» лапласиана L_{FF}. Если количество различных выборов дополнительно не ограниченно, то алгоритм имеет комбинаторную сложность. Поэтому была взята верхняя граница 10000 для сокращения времени выполнения. Очевидно, что алгоритм не гарантирует наилучшего результата, но, как можно убедиться в экспериментах, его результаты лучше, чем у представленных выше алгоритмов.

Что касается оставшихся двух алгоритмов, жадный алгоритм из [17] с оптимизированной реализацией имеет алгоритмическую сложность $O(n^3)$, а сложность без оптимизации составляет $O(n^4)$. Точный алгоритм из [16] для задачи выбора k лидеров имеет полиномиальную алгоритмическую сложность $O(kn^3)$ в случае использования графов, имеющих структуру путей или колец. Таким образом, алгоритм k-средних имеет наименьшую среднюю асимптотическую сложность среди рассмотренных неслучайных алгоритмов.

3 Имитационное моделирование

3.1 Общее описание и параметры экспериментов

Рассматривается многоагентная система из 100 агентов, равномерно распределенных на площади 10×10 м^2. Предполагается, что два агента-последователя симметрично связаны тогда и только тогда, когда начальное расстояние между ними не более 3 м. Веса этих связей выбираются равномерно из интервала (0, 50). Сформированная таким образом взвешенная матрица смежности в дальнейшем не меняется.

Во всех экспериментах измеряется время сходимости (см. определение 1). Поскольку агенты имеют координаты *x* и *y*, рассчитываются отклонения по обеим осям от предельного состояния, и когда оба эти отклонения не превышают фиксированную ошибку, фиксируется, что система достигла предела. Алгоритмы сравниваются по полученному времени сходимости. Кроме того, в качестве альтернативного метода, скорости сходимости алгоритмов оценивались нахождением наименьших ненулевых собственных значений соответствующих «внутренних» лапласовских матриц L_{FF}; результаты представлены в разделе 4.

Каждый эксперимент повторялся от 30 до 100 раз в зависимости от количества лидеров *k* (от 1 до 90); полученное время сходимости усреднялось для каждого *k*.

Для приближения экспериментальной многоагентной системы к реальности были наложены также физические ограничения, в данном случае – ограничения на максимальную скорость. Моделирование проводилось в Webots – симуляторе роботов с открытым исходным кодом, который поддерживает различные модели, в том числе модель роботов e-puck2. Эта модель использовалась с актуальным для e-puck2 ограничением, что скорость не превосходит 15,4 см/с.

Эксперименты с физическими ограничениями реализовывались в Webots на языке C, без ограничений – на Python. Более подробная информация о деталях проведенных экспериментов представлена в подразделе 3.2.

Для всех экспериментов допустимая ошибка сходимости (см. определение 1) принималась равной $5 \cdot 10^{-8}$ см. Эта погрешность выбрана экспериментально, так как она позволяет получить достаточно точные результаты за разумное время.

Количество лидеров изменялось от 1 до 90, но на графиках показаны значения только от 1 до 9, поскольку результаты для большего количества лидеров аналогичны. С ростом количества лидеров время сходимости постепенно уменьшается, а скорость сходимости постепенно увеличивается для всех зависимостей.

Временной шаг моделирования для всех экспериментов с физическими ограничениями и без них был равен 1 мс, что является минимальным доступным шагом в Webots.

3.2 Реализация экспериментов

Моделирование на Python и Webots состояло из 5 основных шагов:

1) Генерация координат;

2) Формирование матрицы смежности на основе этих координат и выбранного правила связи;

3) Построение лапласовской матрицы орграфа зависимостей агентов;

4) Преобразование лапласовской матрицы в соответствии с алгоритмом определения лидеров (раздел 2);

5) Реализация протокола консенсуса.

Первые четыре шага, одинаковые для обоих типов экспериментов, были реализованы на Python.

После генерации координат и построения лапласовской матрицы моделирование повторялось в цикле, содержащем до $\mathrm{N} = 2 \cdot 10^6$ шагов.

Конечное предельное состояние x^* вычислялось как $\mathrm{e}^{\mathrm{LNt_s}}\mathrm{x}(0)$, где $\mathrm{t_s}$ – время шага при имитационном моделировании. Если система не достигала этого предельного состояния за N или меньшее число шагов, то генерировались новые координаты и соответствующая лапласовская матрица, и следующий тур моделирования проводился для них.

3.3 Эксперименты без физических ограничений

На каждом шаге моделирования на Python координаты агентов обновлялись по формуле

$$\mathrm{x_{k+1}} = \mathrm{x_k} - \mathrm{Lx_k t_s};$$

временной шаг моделирования $_{\mathrm{t}}$ в этих экспериментах выбирался равным 1 мс. После данного обновления расстояние между текущим состоянием и предельным состоянием вычислялось в соответствии с определением 1 и сравнивалось покоординатно с фиксированной ошибкой e. Если эти отличия были меньше выбранной ошибки, то моделирование останавливалось, в противном случае осуществлялся переход к следующему шагу.

3.4 Эксперименты с физическими ограничениями

Моделирование на Webots в основном было аналогично моделированию на Python с той разницей, что перед каждым шагом продолжительности t_s агенты обновляли свои скорости по формуле $\dot{\boldsymbol{x}} = \min(-L\boldsymbol{x};\ 15{,}4\ \text{см/с})$; где 15,4 см/с – максимальная допустимая скорость. По своему алгоритму Webots обновляет скорость с учетом текущей скорости, ускорения и крутящего момента двигателя. После шага моделирования продолжительности t_s специально созданный программный агент («инспектор») получает координаты мобильных агентов, рассчитывает отличия до предельного состояния, сравнивает их с фиксированной ошибкой и решает, требуется ли продолжение моделирования. Если требуется, то инспектор пересчитывает скорости мобильных агентов, отправляет им новые значения, и шаг моделирования повторяется снова.

В выбранной схеме моделирования не предусматривались задержки на обмен информацией, но эти задержки могут внести определенные коррективы в случае реализации алгоритма на реальных роботах.

4 Результаты

В этом разделе представлены полученные результаты относительно времени сходимости алгоритмов в зависимости от количества лидеров – с ограничениями на скорость и без таковых.

Прежде всего, сравним алгоритмы с первого по пятый, поскольку их алгоритмическая сложность намного ниже сложности алгоритма «большого» случайного выбора, которая определяется выбранным числом 10000. Как показывает рис. 1, алгоритм k-средних дает наилучший результат, в то время как алгоритмы случайного выбора лидера, выбора k лидеров и выбора лидеров со средней степенью дают примерно одинаковые результаты. Наконец, выбор лидеров с максимальной степенью дает самый худший результат в экспериментах с ограничением скорости. В то же время этот

алгоритм показывает довольно высокие результаты при небольшом количестве лидеров в экспериментах без физических ограничений. Объяснение этой его особенности имеет смысл включить в программу дальнейших исследований.

Следует обратить внимание, что в экспериментах с физическими ограничениями разница между алгоритмами видна четко, и не всегда ранжирование методов совпадает с ранжированием по результатам экспериментов без физических ограничений, в то время как асимптотическая скорость сходимости, которая может быть отождествлена с наименьшим ненулевым собственным значением «внутреннего» лапласиана, одинакова в обоих экспериментах. Тем самым асимптотическую скорость сходимости не следует рассматривать как единственный фактор при выборе алгоритма.

а *б*

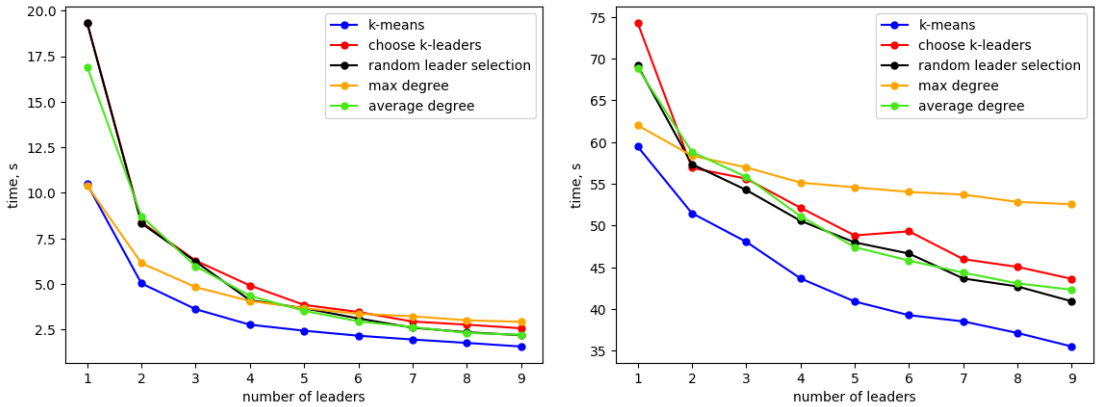

Рис. 1. Сравнение алгоритма k-средних с четырьмя другими алгоритмами: (а) эксперименты без физических ограничений; (б) эксперименты с ограничением скорости

Далее можно заметить, что время сходимости в экспериментах с физическими ограничениями (рис. 1*б*) не всегда монотонно уменьшается с увеличением числа лидеров; имеет смысл исследовать, сохраняются ли эти особенности при снижении разброса среднего в результате увеличения объема эксперимента.

а *б* *в*

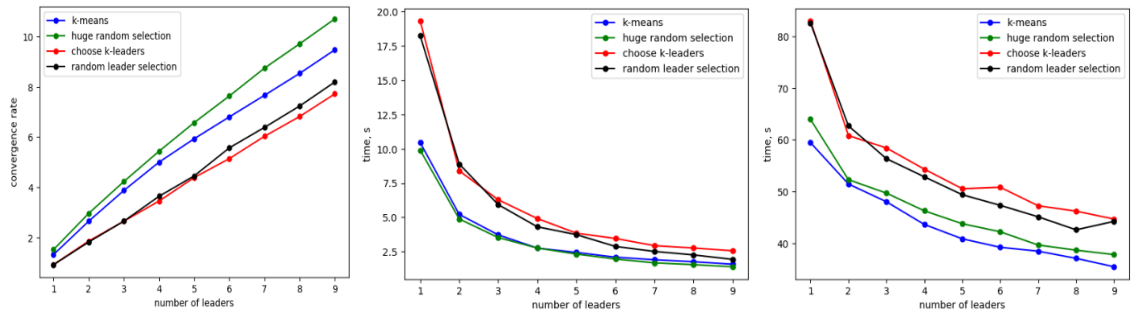

Рис. 2. Сравнение «большого» случайного выбора с тремя другими алгоритмами: (а) скорость сходимости в экспериментах; (б) время в эксперименте без физических ограничений; (в) время в эксперименте с ограничением скорости

Сравним теперь алгоритмы *k*-средних, спектрального выбора *k* лидеров и случайного выбора лидеров с «большим» случайным выбором. При этом этапе сравниваются алгоритмы, которые дали наилучшие либо похожие результаты в предыдущих экспериментах; остальные алгоритмы не включены в сравнение, чтобы улучшить читаемость графиков. Как видно на рис. 2*а*, «большой» случайный выбор обеспечивает наилучшую скорость сходимости, а результат *k*-средних заметно, но не сильно от него отличается. Рисунки 2*б* и 2*в* демонстрируют, что эти два алгоритма дают наилучшие результаты также и по времени сходимости, причем алгоритм *k*-средних показывает даже немного меньшее время сходимости в экспериментах с физическими ограничениями. Очевидно, это связано с критерием остановки, который определяется выбранной допустимой ошибкой *e* (определение 1).

Наконец, исследуем разброс полученных данных, что дает оценку стабильности алгоритма для реального использования. На рис. 3 по вертикальной оси времени отложена разница между максимальным и минимальным временем сходимости для каждого из четырех алгоритмов выбора лидеров. Видно, что разброс времени для алгоритмов *k*-средних и «большого» случайного выбора значительно меньше, чем для алгоритмов спектрального выбора *k* лидеров и случайного выбора лидера; для остальных алгоритмов диапазон близок к разбросу двух последних.

Отношение информационной связи агентов также называют отношением соседства. Это отражает тот факт, что данное отношение естественно строить, руководствуясь близостью агентов в пространстве – так на практике часто и делают. С другой стороны, следствие этого – высокая корреляция расстояний агентов в пространстве (в данном случае – на плоскости) и по графу зависимостей. Имеет смысл провести дополнительное исследование скорости сходимости к консенсусу при нескольких алгоритмах выбора лидеров также в другой постановке, когда отмеченной корреляции нет: сначала строится граф информационных связей мобильных агентов, а затем, независимо от него, генерируются их начальные состояния.

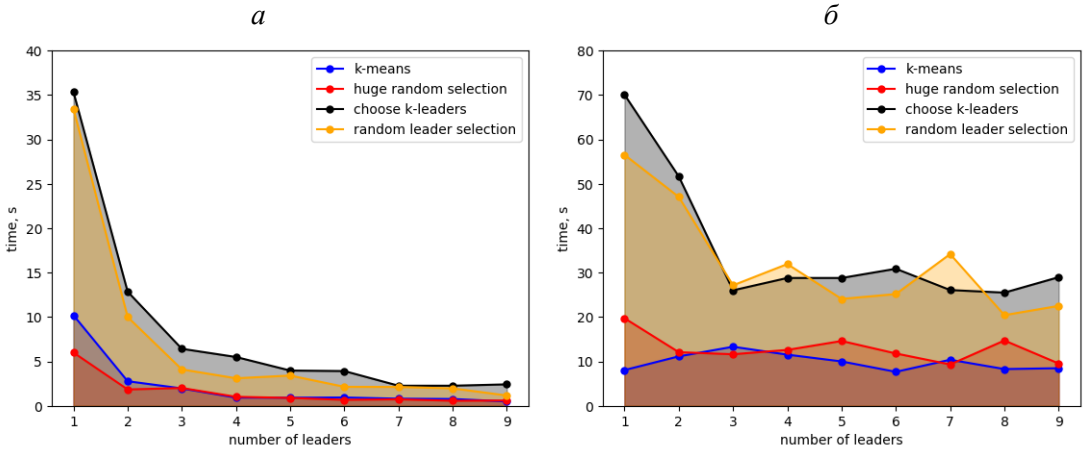

Рис. 3. Разброс времени сходимости: (а) эксперимент без физических ограничений; (б) эксперимент с физическими ограничениями

Заключение

В работе проведено сравнение нескольких алгоритмов решения задачи выбора лидера в схеме лидер-последователь для протокола линейного дифференциального консенсуса – как для систем без физических ограничений, так и для систем с ограничениями по скорости. Сравнивались следующие алгоритмы:

- алгоритм спектрального выбора *k* лидеров [5];
- случайный выбор лидеров;
- выбор агентов с максимальными входящими степенями в графе зависимостей агентов;
- выбор агентов со средними степенями в графе зависимостей агентов;
- алгоритм кластеризации «*k*-средних» с последующим выбором узлов, ближайших к центру каждого кластера.

Все эти алгоритмы сравнивались с алгоритмом «большого» случайного выбора. Алгоритм кластеризации показал наилучшие результаты во всех экспериментах, если исключить из рассмотрения алгоритм «большого» случайного выбора, алгоритмическая сложность которого определяется верхней границей количества различных выборок (в данной серии экспериментов: 10000); без такого ограничения этот алгоритм имеет комбинаторную сложность.

Перспективным продолжением данной работы будет тестирование других алгоритмов кластеризации, а также альтернативной версии алгоритма *k*-средних, где в качестве лидера в каждом кластере выбирается узел с максимальной степенью. В предварительных экспериментах установлено, что метод агломеративной кластеризации, имеющий более высокую среднюю алгоритмическую сложность, чем алгоритм *k*-средних [19], дает результаты, близкие к результатам последнего.

Главный вывод работы состоит в том, что алгоритм кластеризации «*k*-средних» строит наиболее эффективные в классе достаточно быстрых алгоритмов решения задачи о выборе лидеров применительно к протоколу линейного консенсуса. Предварительные результаты тестирования

алгоритма спектральной кластеризации [20] оказались хуже, чем у алгоритма выбора в качестве лидеров агентов, соответствующих узлам с максимальными степенями.

В данной работе не рассматривались системы с мобильными лидерами, с динамическими связями, системы с ускорением, с поддержанием формы. Задачи выбора лидеров для них во многом схожи с задачами выбора лидеров при поиске консенсуса, рассмотренными в статье, но, разумеется, они более сложны как теоретически, так и экспериментально. Развитием данной работы может быть тестирование алгоритмов кластеризации при решении этих задач.

Литература

1 . *Schurr N., Marecki J., Tambe M., Scerri P. Kasinadhuni N., Lewis J. P.* The future of disaster response: humans working with multiagent teams using DEFACTO. AAAI Spring Symposium: AI Technologies for Homeland Security. 2005. – P.9-16.

2 . *Sun R., Naveh I.* Simulating organizational decision-making using a cognitively realistic agent model // Journal of Artificial Societies and Social Simulation. Vol. 7. 2004, № 3.

3 . *Mesbahi M., Egerstedt M.* Graph Theoretic Methods in Multiagent Networks. – Princeton: Princeton University Press, 2010.

4 . *Borsche T., Attia S.A.* On leader selection in multi-agent control systems. 2010 Chinese Control and Decision Conference. – IEEE. 2010. – P.102-107.

5 . *Clark A., Alomair B., Bushnell L., Poovendran R.* Leader selection in multi-agent systems for smooth convergence via fast mixing. Proceedings of the 51st IEEE Conference on Decision and Control. – IEEE. 2012. – P.818-824.

6 . *Sato K.* Optimal leader selection and demotion in leader-follower multi-agent systems. arXiv preprint arXiv:1802.06479. 2018.

7 . *Franchi A., Blthoff H.H., Giordano P. R.* Distributed online leader selection in the bilateral teleoperation of multiple UAVs. 2011 50th IEEE Conference on Decision and Control and European Control Conference. – IEEE. 2011. – P.3559-3565.

8 . *Shah D., Zaman T.* Community detection in networks: The leader-follower algorithm. Proc. of Workshop on Networks Across Disciplines: Theory and Applications. 2010. – P.1-13.

9 . *Meila M., Shi J.* Learning segmentation by random walks. Advances in Neural Information Processing Systems. – MIT Press. 2001. – P.873-879.

10 . *Gorban A.N., Zinovyev A.Y.* Principal graphs and manifolds. Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques. 2009. – P.28-60.

11 . *Lin F., Fardad M., Jovanovic M.R.* Algorithms for leader selection in stochastically forced consensus networks // IEEE Trans. Automat. Control. Vol. 59. 2014, № 7. – P.1789-1802.

12 . *GCTronik, e-puck2. Online* . 2018. Available: https://www.gctronic.com/doc/index.php/e-puck2

13 . *Horn R.A., Johnson C.R.* Matrix Analysis. – Cambridge: Cambridge University Press, 2012.

14 . *Chebotarev P., Agaev R.* Forest matrices around the Laplacian matrix // Linear Algebra and Its Applications. Vol. 356. 2002. – P.253-274.

15 . *Barooah P., Hespanha J.P.* Graph effective resistance and distributed control: Spectral properties and applications. 45th IEEE Conference on Decision and Control. 2006. – P.3479-3485.

16 . *Patterson S., McGlohon N., Dyagilev K.* Optimal k-leader selection for coherence and convergence rate in one-dimensional networks // IEEE Trans. Control Netw. Syst. Vol. 4. 2016, № 3. – P.523-532.

17 . *Wang Y., Yang W., Wang X.* Optimal leader selection for fast consensus via consensus centrality. Proceedings of the 33rd Chinese Control Conference, IEEE. 2014. – P.1482-1487.

18 . *Pedregosa F., Varoquaux G., et al.* Scikit-learn: machine learning in Python // Journal of Machine Learning Research. Vol. 12. 2011. – P.2825-2830.

19 . *Rokach L., Maimon O.* Clustering methods. Data Mining and Knowledge Discovery Handbook. Boston, MA: Springer. 2005. – P.321-352.

20 . *Ng A.Y., Jordan M.I., Weiss Y.* On spectral clustering: Analysis and an algorithm // Advances in Neural Information Processing Systems. Vol. 14. 2002. – P.849-856.