State Aggregation for Distributed Value Iteration in Dynamic Programming

Nikolaus Vertovec   Kostas Margellos

Abstract— We propose a distributed algorithm to solve a
dynamic programming problem with multiple agents, where
each agent has only partial knowledge of the state transition
probabilities and costs. We provide consensus proofs for the
presented algorithm and derive error bounds of the obtained
value function with respect to what is considered as the
"true solution" obtained from conventional value iteration.
To minimize communication overhead between agents, state
costs are aggregated and shared between agents only when the
updated costs are expected to influence the solution of other
agents significantly. We demonstrate the efficacy of the proposed
distributed aggregation method to a large-scale urban traffic
routing problem. Individual agents compute the fastest route to
a common access point and share local congestion information
with other agents allowing for fully distributed routing with
minimal communication between agents.

Index Terms— Dynamic programming; Value iteration; Con-
sensus; Distributed algorithms.

I. INTRODUCTION

Value iteration is a well-established method for solving
dynamic programming problems, yet it exhibits scalability
issues for applications with a large state space. To this end,
state aggregation can be used to reduce the number of states
that need to be considered. The aggregation approach has a
long history in scientific computing with applications ranging
from the improvement of Galerkin methods [1], to solving
large-scale optimization models [2], and dynamic program-
ming [3]. In [4] it is shown how the aggregation approach
can be used in conjunction with value iteration, however, for
problems with large decision spaces and potentially cost and
transition probabilities that evolve over time, conventional
aggregation will still be hampered by scalability issues. To
this end, we propose a multi-agent value iteration algorithm
that utilizes aggregation to minimize communication over-
head and allows for solving dynamic programming problems
where each agent has only partial knowledge of the state
transition costs and probability.

Value iteration has been expanded to a multi-agent frame-
work for the class of problems that contain a joint decision
space [5], yet in this approach no restrictions are imposed
as far as the agents' knowledge of the underlying transition
probabilities and costs are concerned. Such restrictions have
been considered for small-scale problems as in [6], however,
require a centralized critic for estimating the expected team
benefit in a non-cooperative setting.

The proposed framework differs from many multi-agent
reinforcement learning approaches that utilize sharing a
weighted average of agents' local estimated value. Rather
than considering a set of global states shared by all the agents
in conjunction with a local reward observed by individual
agents as in [7], [8], we consider a Markov Decision Pro-
cess (MDP) partitioned among agents with the objective of
determining in a distributed manner the global value function
without disclosing information on the transition probabilities
and costs among agents.

This paper performs the following main contributions: (i)
We propose a distributed value iteration algorithm for multi-
agent dynamic programming, preventing transition probabil-
ities and state costs from constituting global information
among agents; (ii) We provide a rigorous mathematical
analysis on the convergence and optimality properties of
the proposed algorithm, merging tools from multi-agent
consensus and dynamic programming principles; and, (iii)
We demonstrate our scheme on a non-trivial traffic routing
problem.

The rest of the paper is organized as follows. Section II
provides some mathematical preliminaries and introduces the
distributed multi-agent value iteration algorithm. Section III
provides the main statements and associated mathematical
analysis supporting the proposed algorithm. In Section IV we
introduce a traffic routing problem and show how our pro-
posed approach is able to solve the problem in a distributed
manner. Finally, Section V provides concluding remarks and
directions for future work.

II. PROBLEM SETUP

A. Multi-agent Markov Decision Processes

We first introduce a standard MDP that will be expanded
to a multi-agent setting in view of the proposed distributed
algorithm presented in the sequel. We consider a finite set
of n states denoted by $\mathcal{X}$ and let $g(i, u, j)$ be the cost-to-go
from state $i \in \mathcal{X}$ to state $j \in \mathcal{X}$, using the input $u \in \mathcal{U}(i)$,
where $\mathcal{U}(i)$ is a finite set of actions/decisions available at
state i. Furthermore, let $p(i, u, j)$ be the transition probability
to transition from state i to state j using the input u, with
$\sum_{j=1}^{n} p(i, u, j) = 1$. The objective is to solve a global
discounted infinite horizon dynamic programming problem.
The associated Bellman equation is given by $J^*(i) =
\min_{u \in U(i)} \sum_{j=1}^{n} p(i, u, j)[g(i, u, j) + \alpha J^*(j)]$, where $\alpha \in
[0, 1)$ is a discount factor, and J^* denotes the optimal
value function that satisfies the Bellman identity. Solving the
Bellman equation using conventional value iteration or policy
iteration requires knowledge of all transition probabilities as
well as associated costs.

We consider the setting where a set of q agents collaborate to solve the aforementioned infinite horizon dynamic programming problem in a distributed manner, with only partial knowledge of the state transition probabilities and costs while minimizing communication between agents. As such, we partition the state-space $\mathcal{X}$ into q subsets, $I_\ell \subset \mathcal{X}$, such that for all $m, \ell \in \{1, \ldots, q\}$ with $m \neq \ell$, $I_\ell \cap I_m = \emptyset$. Each agent knows only the state transition probabilities and cost for transitions originating within its state subset I_ℓ, i.e., for each agent ℓ, $g_\ell : I_\ell \times \mathcal{U} \times \mathcal{X} \to \mathbb{R}$, $p_\ell : I_\ell \times \mathcal{U} \times \mathcal{X} \to [0, 1]$. Using conventional value iteration would require each agent to share its knowledge of the transition probabilities and costs, resulting in significant communication overhead. In the subsequent section, we instead introduce an alternative method relying on state aggregation that allows solving the discounted infinite horizon dynamic programming problem under consideration in a distributed manner where some tentative information is exchanged only with neighboring agents.

B. Distributed Value Iteration

We start by aggregating the value function (expected optimal cost-to-go) to construct for each agent $\ell = 1, \ldots, q$, the aggregate value, defined as $r_{\ell,\ell} = \sum_{i \in I_\ell} d_{\ell i} J(i)$, where J constitutes an approximation of the optimal value function of the Bellman equation, and $d_{\ell i}$ is the so-called disaggregation probability (encoding the contribution of each agent's value function to the aggregate one), satisfying for all $\ell = 1, \ldots, q$, $\sum_{i \in I_\ell} d_{\ell i} = 1$ and for all $i \notin I_\ell$, $d_{\ell i} = 0$. The aggregate values for each agent, $r_{\ell,\ell}$, can be combined into a common vector denoted by r_ℓ; tentative values for this vector will be communicated across all agents. Thus we define $r_\ell = \left[r_{\ell,1}, \ldots, r_{\ell,\ell}, \ldots, r_{\ell,q} \right]^T$. In the sequel, we will define an iterative scheme, according to which each agent $\ell = 1, \ldots q$ will update their estimate for the vector r_ℓ; we denote this at iteration k, by r_ℓ^k, where its m-th element is indicated by $r_{\ell,m}^k$, $m = 1, \ldots, q$.

Next, we introduce the aggregation probability satisfying for all $\ell = 1, \ldots, q$,

$$\phi_{j\ell} = \begin{cases} 1, & \text{if } j \in I_\ell, \\ 0, & \text{otherwise.} \end{cases}$$

Such a formulation of the aggregation probability is known as hard aggregation in the literature [9, p. 311].

To solve the discounted infinite horizon dynamic programming problem, we propose a distributed algorithm for which the pseudocode is given in Algorithm 1 and 2. The proposed algorithm involves an agent-to-agent communication protocol. At each algorithm iteration $k \geq 0$ we consider the directed communication graph $(V, E(k))$, where the node set $V = \{1, \ldots, q\}$ includes the agents and the set $E(k)$, the directed edges (m, l), indicating that at iteration k agent ℓ can receive information from agent m. Let $E_B(k) = \bigcup_{i=k}^{(k+B)} E(i)$ for some integer B. We make the fairly standard assumption in the existing literature on the communication structure between agents [8], [10]:

Assumption 2.1: [Connectivity and Communication] There exists a positive integer B such that $(V, E_B(k))$ is fully connected for each iteration k.

Assumption 2.1 implies that for any agent pair (ℓ, m) there is a direct link at least once every B iterations. This prevents agents from having to share information with all other agents at all iterations as well as for the presence of a central authority.

Initially in Algorithm 1, each agent ℓ, $\ell = 1, \ldots, q$, starts with some tentative value of the aggregate vector r_ℓ^0, and local cost function V_ℓ^0. The initialization choice is arbitrary. Utilizing the aggregate vector r_ℓ^k at iteration $k \geq 0$, each agent ℓ locally and in parallel executes Algorithm 2 so as to construct the updated local value function $V_\ell^{k+1} : I_\ell \to \mathbb{R}$ which will, in turn, yield an updated aggregated cost $r_{\ell,\ell}^{k+1} \in \mathbb{R}$ (Algorithm 1, Steps 8).

To minimize communication overhead, an updated aggregated cost $r_{\ell,\ell}^{k+1}$ is only shared with other agents in the set $N_\ell(k)$ (the neighbors of ℓ at iteration k, when the aggregated cost has changed significantly, i.e., $\|r_{\ell,\ell}^{k+1} - r_{\ell,\ell}^{\text{prev}}\| > C_{\text{threshold}}$, where $r_{\ell,\ell}^{\text{prev}}$ is the aggregated cost previously broadcast to other agents and $C_{\text{threshold}}$ is a communication threshold (Algorithm 1, Steps 11 - 13), or when agent ℓ and a neighboring agent m have not communicated in the last B iterations (see Step 11; this ensures that there exists a bounded intercommunication time as required by Assumption 2.1). For further discussion on the communication threshold to limit the transmission of insignificant data, we refer to [11].

Until convergence is reached, each agent will repeat the execution of Algorithm 2 and the subsequent communication step. Note that at iteration k, the vector r_ℓ^k may contain aggregate values from other agents received at prior time steps (Algorithm 1, Step 17). It will be shown in the proof of Theorem 3.1 that convergence will nevertheless be reached.

We now turn to the update of V_ℓ^k and $r_{\ell,\ell}^k$, performed when calling Algorithm 2. Each agent uses the local value function, V_ℓ^k, to represent the cost-to-go function at each state $i \in I_\ell$, and the aggregated cost, r_ℓ^k, to approximate the cost for states $j \in \mathcal{X} \backslash I_\ell$. The local value function is iteratively updated for each state $i \in I_\ell$ (Algorithm 2, Steps 2 - 6). This update follows from standard value iteration; choosing the control action at state i that minimizes the expected cost-to-go (Algorithm 2, Step 4) and using a Gauss-Seidel update on the local value function to improve convergence and minimize memory usage (Algorithm 2, Step 5). We perform an update only to the local value function since we restrict ourselves to states for which sufficient knowledge of the transition probabilities and costs is available. After the local value function has been updated, the updated local aggregated cost, $r_{\ell,\ell}^{k+1}$, is computed (Algorithm 2, Step 7).

Notice that Algorithms 1 and 2 prevent disclosing the transition probabilities to all agents; in contrast, each agent ℓ, has access only to the probabilities and costs associated with transitions originating from their partition I_ℓ, $\ell = 1, \ldots, q$.

Algorithm 1 Distributed value iteration
1: **Initialization**
2: Set $r_\ell^0 \leftarrow [0, \ldots, 0]$ for all $\ell = 1, \ldots, q$
3: Set $r_{\ell,\ell}^{\text{prev}} \leftarrow 0$ for all $\ell = 1, \ldots, q$
4: Set $V_\ell^0(i) \leftarrow 0$ for all $i \in I_\ell$ and $\ell = 1, \ldots, q$
5: $k = 0$
6: **repeat**
7:     **for** each Agent $\ell \in 1, \ldots, q$ **do**
8:         $V_\ell^{k+1}, r_{\ell,\ell}^{k+1} \leftarrow$ Agent-Update(V_ℓ^k, r_ℓ^k)
9:         $N_\ell(k) = \{m \in \{1, \ldots, q\} : (\ell, m) \in E(k)\}$
10:         **if** $\|r_{\ell,\ell}^{k+1} - r_{\ell,\ell}^{\text{prev}}\| > C_{\text{threshold}}$,
11:             **or if** $(\ell, m) \notin \bigcup_{i=k-B+1}^{k} E(i)$ **then**
12:             Send $r_{\ell,\ell}^{k+1}$ to all agents $m \in N_\ell(k)$
13:         **end if**
14:         **if** receiving information from an agent m **then**
15:             $r_{\ell,m}^{k+1} \leftarrow r_{m,m}^{k+1}$
16:         **else**
17:             $r_{\ell,m}^{k+1} \leftarrow r_{\ell,m}^k$
18:         **end if**
19:     **end for**
20: **until** $\|r_\ell^{k+1} - r_\ell^k\| \leq$ tolerence for all $\ell = 1, \ldots, q$

Algorithm 2 Agent-Update
1: **function** AGENT-UPDATE(V_ℓ^k, r_ℓ^k)
2:     **for** $i \in I_\ell$ **do**
3:         $V_\ell^{k+1}(i) \leftarrow \min_{u \in U(i)} \sum_{j=1}^n p_\ell(i, u, j)[g_\ell(i, u, j)$
4:                 $+ \alpha(\phi_{j\ell}V_\ell^k(j) + \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}r_{\ell,m}^k)]$
5:         $V_\ell^k(i) \leftarrow V_\ell^{k+1}(i)$
6:     **end for**
7:     $r_{\ell,\ell}^{k+1} \leftarrow \sum_{j \in I_\ell} d_{\ell j} V_\ell^{k+1}(j)$
8:     **return** $V_\ell^{k+1}, r_{\ell,\ell}^{k+1}$
9: **end function**

III. ALGORITHM ANALYSIS

The following theorem is the main result of this section.

Theorem 3.1: Consider Assumption 2.1. Algorithm 1 converges asymptotically to a common value for r among agents, i.e., there exists $\bar{r}$ such that for all $\ell = 1, \ldots, q$, $\lim_{k \to \infty} \|r_\ell^k - \bar{r}\| = 0$. Moreover, for all $\ell = 1, \ldots, q$, for all $i \in \mathcal{X}$, $V_\ell^k(i)$ converges to some $\overline{V}_\ell^*(i)$.

Proof: We show convergence when agents transmit $r_{\ell,\ell}^k$, $\ell = 1, \ldots, q$, irrespective of whether this deviates more than $C_{\text{threshold}}$ from their previously transmitted cost. This is without loss of generality as in this case the resulting sequence of transmitted costs would form a subsequence of $\{r_{\ell,\ell}^k\}_{k=0}^\infty$, hence it will be convergent as we will show that $\{r_{\ell,\ell}^k\}_{k=0}^\infty$ converges.

Fix any $k \geq B$, and any $\ell \in \{1, \ldots, q\}$. Let $\epsilon_{\ell,m}^k = r_{\ell,m}^{k+1} - r_{\ell,m}^k$, $\epsilon_{\ell,\infty}^k = \max_{m=1,\ldots,q} |r_{\ell,m}^{k+1} - r_{\ell,m}^k|$, and $\epsilon_\infty^k = \max_{\ell=1,\ldots,q} \sum_{n=k-B}^k \epsilon_{\ell,\infty}^n$, i.e., ϵ_∞^k is the maximum among agents of the cumulative incremental update over the most recent $B + 1$ iterations; recall that due to Assumption 2.1 this is the window within which agent ℓ communicates with

other agents at least once.

Consider the update to the value $V_\ell^k(i)$, for an arbitrary iteration k and agent $\ell = 1, \ldots, q$, which we define as

$$\delta_\ell^k(i) = V_\ell^{k+1}(i) - V_\ell^k(i) = \min_{u \in U(i)} \sum_{j=1}^n p_\ell(i, u, j)\Big(g_\ell(i, u, j)$$

$$+ \alpha[\phi_{j\ell}V_\ell^k(j) + \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}r_{\ell,m}^k]\Big) - V_\ell^k(i)$$

$$= \min_{u \in U(i)} \Big(\sum_{j=1}^n p_\ell(i, u, j)\Big(g_\ell(i, u, j) + \alpha[\phi_{j\ell}V_\ell^{k-1}(j)$$

$$+ \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}r_{\ell,m}^{k-1}]\Big) + \alpha \sum_{j=1}^n p_\ell(i, u, j)\Big(\phi_{j\ell}\delta_\ell^{k-1}(j)$$

$$+ \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}\epsilon_{\ell,m}^{k-1}\Big)\Big) - V_\ell^k(i)$$

$$\leq V_\ell^k(i) + \max_{u \in U(i)} \alpha \sum_{j=1}^n p_\ell(i, u, j)\Big(\phi_{j\ell}\delta_\ell^{k-1}(j)$$

$$+ \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}\epsilon_{\ell,m}^{k-1}\Big) - V_\ell^k(i)$$

$$= \max_{u \in U(i)} \alpha \sum_{j=1}^n p_\ell(i, u, j)\Big(\phi_{j\ell}\delta_\ell^{k-1}(j) + \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}\epsilon_{\ell,m}^{k-1}\Big),$$

where the first equality follows from the definition of V_ℓ^k, and the second one from a rearrangement after substituting $V_\ell^k(j) = V_\ell^{k-1}(j) + \delta_\ell^{k-1}(j)$ and $r_{\ell,m}^k = r_{\ell,m}^{k-1} + \epsilon_{\ell,m}^{k-1}$. The inequality follows from the definition of V_ℓ^k and by considering the maximum over $u \in U(i)$.

For each $\ell = 1, \ldots, q$, we now define the maximum update of $|\delta_\ell^k|$, over all states $i \in I_\ell$, as $\overline{\delta}_\ell^k = \max_{i \in I_\ell} |\delta_\ell^k(i)|$. It then follows that for all $i \in I_\ell$,

$$\delta_\ell^k(i) \leq \max_{u \in U(i)} \alpha \sum_{j=1}^n p_\ell(i, u, j)\Big(\phi_{j\ell}\overline{\delta}_\ell^{k-1} + \sum_{\substack{m=1 \\ m \neq \ell}}^q \phi_{jm}\epsilon_{\ell,\infty}^{k-1}\Big)$$

$$\leq \max_{u \in U(i)} \alpha \sum_{j=1}^n p_\ell(i, u, j)\Big(\sum_{m=1}^q \phi_{jm} \max\{\overline{\delta}_\ell^{k-1}, \epsilon_{\ell,\infty}^{k-1}\}\Big)$$

$$= \alpha \max\{\overline{\delta}_\ell^{k-1}, \epsilon_{\ell,\infty}^{k-1}\}, \tag{1}$$

where the first inequality follows from the definition of $\overline{\delta}_\ell^k$, and the equality follows from the fact that $\sum_{j=1}^n p(i, u, j) = 1$, and $\sum_{m=1}^q \phi_{jm} = 1$. Define $\overline{\delta}_\infty^k := \max_{\ell=1,\ldots,q} \sum_{n=k-B}^k \overline{\delta}_\ell^n$, resulting in

$$\overline{\delta}_\infty^k \leq \alpha \max_{\ell=1,\ldots,q} \sum_{n=k-B}^k \max\{\overline{\delta}_\ell^{k-1}, \epsilon_{\ell,\infty}^{k-1}\}$$

$$= \alpha \max\{\overline{\delta}_\infty^{k-1}, \epsilon_\infty^{k-1}\}, \tag{2}$$

where the inequality is since (1) holds for all $i \in I_\ell$, and the exchange of the summation and the maximization operator

is since all quantities are non-negative. At the same time,

$$\sum_{n=k-B}^{k} \max_{m=1,\ldots,q} |r_{\ell,m}^{n+1} - r_{\ell,m}^n|$$

$$\leq \sum_{n=k-B}^{k} \max_{m=1,\ldots,q} |r_{m,m}^{n+1} - r_{m,m}^n|$$

$$= \sum_{n=k-B}^{k} \max_{m=1,\ldots,q} |\sum_{i \in I_m} d_{mi}(V_m^n(i) + \delta_m^n(i)) - r_{m,m}^n|$$

$$= \sum_{n=k-B}^{k} \max_{m=1,\ldots,q} |\sum_{i \in I_m} d_{mi} \delta_m^n(i)|$$

$$= \sum_{n=k-B}^{k} \max_{m=1,\ldots,q} \overline{\delta}_m^n = \max_{m=1,\ldots,q} \sum_{n=k-B}^{k} \overline{\delta}_m^n = \overline{\delta}_\infty^k, \quad (3)$$

where the inequality is since $\sum_{n=k-B}^{k} |r_{\ell,m}^{n+1} - r_{\ell,m}^n|$ is bounded by the value this would become if agents ℓ and m communicated at every iteration $n = k - B + 1, \ldots, k$ (see Steps 14-18, Algorithm 1). The first equality follows from the fact that $r_{m,m}^{n+1} = \sum_{i \in I_m} d_{mi} V_m^{n+1}(i)$ (see Step 7, Algorithm 2), and since $V_m^{n+1}(i) = V_m^n(i) + \delta_m^n(i)$, while the last one is due to the definition of $\overline{\delta}_\infty^k$.

By (3) we can upper-bound $\epsilon_\infty^k = \max_{\ell=1,\ldots,q} \sum_{n=k-B}^{k} \max_{m=1,\ldots,q} |r_{\ell,m}^{n+1} - r_{\ell,m}^n|$ as

$$\epsilon_\infty^k \leq \max_{\ell=1,\ldots,q} \overline{\delta}_\infty^k \leq \alpha \max\{\overline{\delta}_\infty^{k-1}, \epsilon_\infty^{k-1}\}, \quad (4)$$

where the last inequality follows from (2), and the fact that the bound in (2) is independent of ℓ.

By (2) and (4) we then have that $\max\{\overline{\delta}_\infty^k, \epsilon_\infty^k\} \leq \alpha \max\{\overline{\delta}_\infty^{k-1}, \epsilon_\infty^{k-1}\}$, which implies that $\max\{\overline{\delta}_\infty^k, \epsilon_\infty^k\}$ is contractive. As a result, $\{\max\{\overline{\delta}_\infty^k, \epsilon_\infty^k\}\}_{k \geq 0}$, and hence also $\{\epsilon_\infty^k\}_{k \geq 0}$ and $\{\overline{\delta}_\infty^k\}_{k \geq 0}$ will be converging to zero. Since $\{\epsilon_\infty^k\}_{k \geq 0}$ converges, then each term in the summation in the definition of ϵ_∞^k will also converge, implying that $\{|r_{\ell,m}^{k+1} - r_{\ell,m}^k|\}_{k \geq 0}$ is convergent. This implies that for all $\ell = 1, \ldots, q$ there exists $\bar{r}$ such that $\lim_{k \to \infty} \|r_\ell^k - \bar{r}\| = 0$, while convergence of $\{\overline{\delta}_\infty^k\}_{k \geq 0}$ (due to (2), and the definition of $\delta_\ell^k(i)$) implies that for all $\ell = 1, \ldots, q$, for all $i \in \mathcal{X}$, $\{V_\ell^k(i)\}_{k \geq 0}$ would be convergent, thus concluding the proof. ∎

Theorem 3.1 implies consensus among agents to a common $\bar{r}$, and also establishes convergence of $\{V_\ell^k(i)\}_{k \geq 0}$ to some $V_\ell^*(i)$. Moreover, the convergence rate is linear since $\max\{\overline{\delta}_\infty^k, \epsilon_\infty^k\}$ is shown to be contractive in the proof of Theorem 3.1. The exact convergence rate for $\{|r_{\ell,m}^{k+1} - r_{\ell,m}^k|\}_{k \geq 0}$ will depend on B.

Next, we consider error bounds between the limiting $V_\ell^*(i)$ and the optimal $J^*(i)$, satisfying the Bellman equation. For the subsequent result we assume that $B = 0$, i.e., agents communicate with all other agents at all iterations.

Theorem 3.2: Consider Assumption 2.1 with $B = 0$ and Algorithm 1. For all $\ell = 1, \ldots, q$, $i \in \mathcal{X}$, we have that the

limit point $V_\ell^*(i)$ of $\{V_\ell^k(i)\}_{k \geq 0}$ satisfies

$$|V_\ell^*(i) - J^*(i)| \leq \alpha \frac{\delta}{1 - \alpha}, \quad (5)$$

where $\delta := \max_{\ell \in \{1,\ldots,q\}} \max_{i,j \in I_\ell} |J^*(i) - J^*(j)|$ and J^* is the solution of the Bellman equation.

Proof: The proof is inspired by [4]. Fix $\ell = 1, \ldots, q$ and $i \in \mathcal{X}$. We only show that $V_\ell^*(i) \leq J^*(i) + \alpha \frac{\delta}{1-\alpha}$, as the other side in (5) follows symmetric arguments. To this end, for each $i \in I_\ell$, define $\overline{V}_\ell(i) := J^*(i) + \alpha \frac{\delta}{1-\alpha}$. It follows then that the aggregate of $\overline{V}_\ell(i)$ can be constructed as

$$\overline{r}_{\ell,\ell} := \sum_{j \in I_\ell} d_{\ell j} \overline{V}_\ell(j) = \sum_{j \in I_\ell} d_{\ell j}(J^*(j) + \alpha \frac{\delta}{1 - \alpha})$$

$$\leq \sum_{j \in I_\ell} d_{\ell j}(\min_{i \in I_\ell} J^*(i) + \delta + \alpha \frac{\delta}{1 - \alpha})$$

$$= \min_{i \in I_\ell} J^*(i) + \delta + \alpha \frac{\delta}{1 - \alpha} = \min_{i \in I_\ell} J^*(i) + \frac{\delta}{1 - \alpha}, \quad (6)$$

where the inequality is since $J^*(j) \leq \min_{i \in I_\ell} J^*(i) + \delta$, while the second last equality is due to $\sum_{j \in I_\ell} d_{\ell j} = 1$.

Denote the Bellman operator induced by the Bellman equation as T, such that we can compactly write the Bellman equation as $J^*(i) = (TJ^*)(i)$, where effectively by $(TJ^*)(i)$ we imply the right-hand side of the Bellman equation which depends on i and on $J^*(j)$ for all $j = 1, \ldots, n$. For all $i = 1, \ldots, n$, we now consider the application of the Bellman operator to $\overline{V}_\ell$, i.e.,

$$(T\overline{V}_\ell)(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_\ell(i, u, j) \Big(g_\ell(i, u, j) + \alpha[\phi_{j\ell} \overline{V}_\ell(j)$$
$$+ \sum_{\substack{m=1 \\ m \neq \ell}}^{q} \phi_{jm} \overline{r}_{\ell,m}] \Big)$$

$$\leq \min_{u \in U(i)} \sum_{j=1}^{n} p_\ell(i, u, j) \Big(g_\ell(i, u, j) + \alpha[\phi_{j\ell}(J^*(j)$$
$$+ \alpha \frac{\delta}{1 - \alpha}) + \sum_{\substack{m=1 \\ m \neq \ell}}^{q} \phi_{jm}(\min_{j \in I_m} J^*(j) + \frac{\delta}{1 - \alpha})] \Big)$$

$$\leq \min_{u \in U(i)} \sum_{j=1}^{n} p_\ell(i, u, j) \Big(g_\ell(i, u, j)$$
$$+ \alpha \sum_{m=1}^{q} \phi_{jm}(J^*(j) + \frac{\delta}{1 - \alpha}) \Big)$$

$$\leq \min_{u \in U(i)} \sum_{j=1}^{n} p_\ell(i, u, j) \Big(g_\ell(i, u, j) + \alpha J^*(j) \Big)$$
$$+ \alpha \frac{\delta}{1 - \alpha} = J^*(i) + \alpha \frac{\delta}{1 - \alpha} = \overline{V}_\ell(i), \quad (7)$$

where the first inequality is due to the definition of $\overline{V}_\ell$ and utilizes $\overline{r}_{\ell,m} = \overline{r}_{m,m}$, which holds since $B = 0$. The second inequality follows, since $\min_{j \in I_m} J^*(j) \leq J^*(j)$ for all $j \in \mathcal{X}$, and $\alpha \frac{\delta}{1-\alpha} \leq \frac{\delta}{1-\alpha}$, that in turn allows us to combine the terms multiplied with $\phi_{j\ell}$ into the last summation. The last
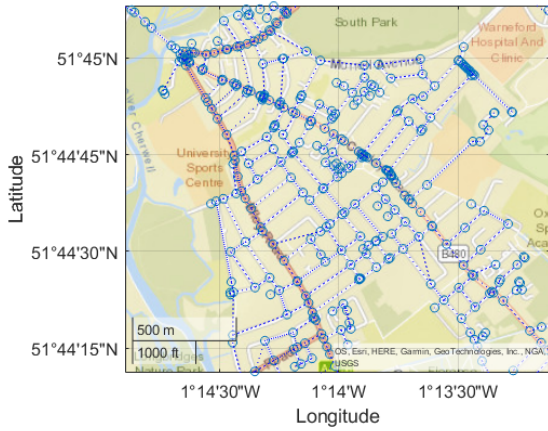
Fig. 1. Graph representation of the East-Oxford road network. Junctions are represented by vertices and edges represent roads.

Fig. 2. The complete Oxford road network is subdivided into 5 subgraphs representing the aggregation sets.

inequality follows from $\sum_{m=1}^{q} \phi_{jm} = 1$, while the second last equality is due to the Bellman equation.

By (7) we have that $(T\overline{V}_\ell)(i) \leq \overline{V}_\ell(i)$, for all $i = 1, \ldots, n$, which implies that $\{\overline{V}_\ell^k(i)\}_{k\geq 0}$ is a non-increasing sequence. Moreover, T is contractive and as such it will converge to its (unique) fixed point; a direct consequence of Theorem 3.1 is that this fixed point is $V_\ell^*(i)$ (as the latter was constructed by successive applications of the Bellman operator). Therefore, for all $i = 1, \ldots, n$, $V_\ell^*(i) = \lim_{k\to\infty}(T^k\overline{V}_\ell)(i) \leq \overline{V}_\ell(i)$, thus establishing $V_\ell(i) \leq J^*(i) + \alpha\frac{\delta}{1-\alpha}$, concluding the proof. ∎

It follows from Theorem 3.2, that if the aggregation sets $I_1, \ldots, I_q$ are chosen such that the cost function J^* is expected to vary moderately between states within an aggregation set, then the maximum error compared to $V_1, \ldots, V_q$ will be moderate as well. One method to aggregate states with similar costs is to use feature-based aggregation, whereby the aggregation is performed on a set of representative features instead of representative states. This form of aggregation has been thoroughly investigated; we refer to [9, pg. 322], [12, pg. 68] and references therein. For applications with a high discount factor, i.e., $\alpha << 1$, the maximum error will also be moderate.

IV. APPLICATION TO TRAFFIC ROUTING

A. Simulation Set-up

We demonstrate the proposed algorithm in a traffic routing case study. To this end, we begin by modeling the traffic network as a graph, with vertices representing junctions and edges representing roads connecting junctions. An illustration of such a graph representation for the Oxford road network is shown in Figure 1.

We consider a low-energy antenna with a limited range and computation power to be situated at the center of each network partition, gathering the current speed and location of nearby vehicles. The transition cost of an edge is the expected travel time along that edge and is computed based on the data received by cars, i.e.,
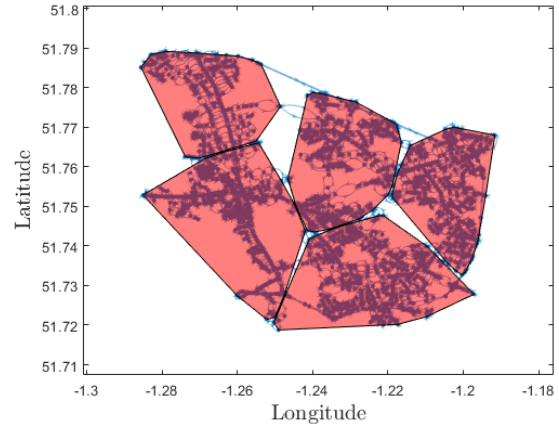
$$g_\ell(i, u, j) = \frac{\text{length of edge}}{\text{average speed of cars on edge}}, \quad (8)$$

where u is used to determine the selected edge between node i and j. Due to the nature of the low-energy application, we utilize the proposed algorithm to limit the amount of data that needs to be sent over longer distances.

As is common in transit node routing, vehicles are trying to reach a common access node, such as a freeway used for long-distance routing. The goal is to find the fastest path to the access point, thus the cost at each vertex is the discounted expected travel time to the access point. In our example related to the Oxford traffic network, the access point will be London Road, leading long-distance travelers toward London.

B. Simulation results

The Oxford road network is partitioned into 5 subgraphs using K-means clustering of the vertices based on their euclidean distance, as shown in Figure 2.

The cost-to-go is calculated as in (8), where the average car speed is randomly generated from a uniform distribution to lie between 25% and 100% of the speed limit of the relevant road. For each vertex, the outgoing edges are enumerated and the transition probability between two vertices is set to either 1 or 0, depending on whether an edge directly connects the vertices and the input selects that edge.

The disaggregation probabilities are zero for all nodes that have no edge connected to a vertex in another aggregation set I_ℓ. The remaining vertices are given a normalized non-zero disaggregation probability. The discount factor, $\alpha = 0.9$, is chosen close enough to 1 so as to reflect the desire to reach the final node and avoid loops, yet smaller than 1, so as to weight costs further off as less important due to the constantly evolving traffic situation.

Solving the aggregate problem and comparing it to what is considered to be the "true solution" J^* obtained via conventional value iteration, we notice
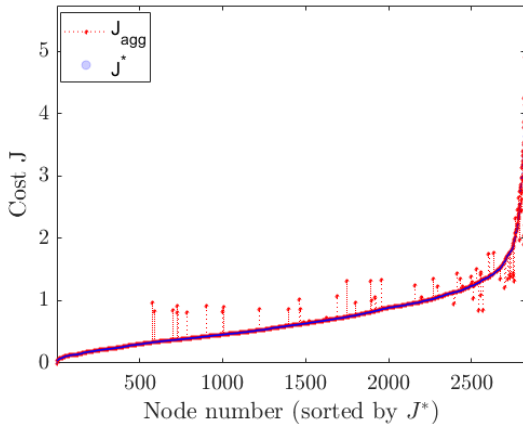
Fig. 3. The node cost sorted by size. The blue line indicates the "true cost" computed using conventional value iteration, the red markers represent the cost obtained by the proposed distributed algorithm.

Fig. 4. The aggregated cost r evolving over time with minimal communication between agents.

that the normalized average error of the expected cost-to-go, i.e., $\frac{1}{n} \sum_{\ell=1,\ldots,q} \sum_{i \in I_\ell} \frac{|\phi_{i\ell} V_\ell(i) - J^*(i)|}{|J^*(i)|}$, is 0.94%, and the normalized maximum error i.e., $\max_{\ell=1,\ldots,q} \max_{i \in I_\ell} \frac{|\phi_{i\ell} V_\ell(i) - J^*(i)|}{|J^*(i)|}$, is 190.83%.

The value function is shown in Figure 3. The evolution over time of the aggregated costs is shown in Figure 4, where the colored dots represent when the change to an agent's local value function compared to the last broadcast is greater than the chosen communication threshold (0.1 in this example). At this point, the agent will broadcast its updated aggregated cost. If since the last broadcast, no agents value function has changed significantly, the agents signal each other that convergence is reached and the algorithm terminates. For comparison, we show how, on average, the normalized error increases as we increase the number of agents (see Table below). This is to be expected, as agents will rely more heavily on the aggregated values the smaller their respective partitions become.

Number of Agents	4	8	12	16
Normalized average error	0.67%	1.63%	2.84%	4.46%

To reproduce the numerical results the associated code has been made available in [13], with the ability to upload any OpenStreetMap file, which is then converted to a graph and subsequently a discounted Markov decision process.

V. Conclusion

We presented a multi-agent extension of aggregated value iteration which was shown to be able to solve large-scale dynamic programming problems in a fully distributed manner. The presented methodology finds application in problems where each agent has only partial knowledge of the transition probabilities and costs. To this end, we demonstrated its efficacy in a distributed traffic routing problem, for which the code has been made available in [13].

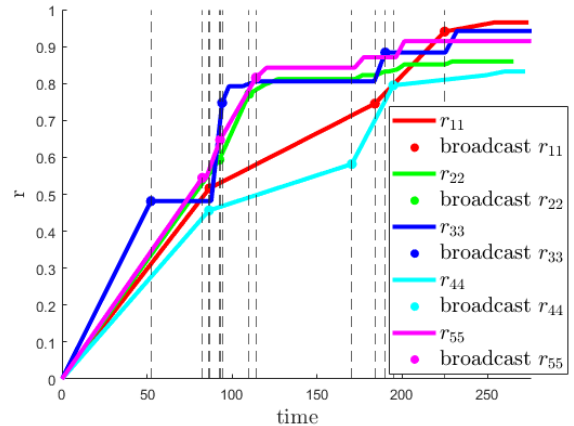Future work aims at extending Theorem 3.2 that is based on the additional assumption of full network connectivity at all iterations to the more general case of Assumption 2.1, as well as to the case of hard aggregation (where each state is clearly assigned to no more than one agent).

References

[1] F. Chatelin and W. L. Miranker, "Acceleration by aggregation of successive approximation methods," *Linear Algebra and its Applications*, vol. 43, pp. 17–47, 1982.

[2] D. F. Rogers, R. D. Plante, R. T. Wong, and J. R. Evans, "Aggregation and disaggregation techniques and methodology in optimization," *Operations Research*, vol. 39, no. 4, pp. 553–582, 1991.

[3] J. C. Bean, J. R. Birge, and R. L. Smith, "Aggregation in dynamic programming," *Operations Research*, vol. 35, no. 2, pp. 215–220, 1987.

[4] J. N. Tsitsiklis and B. van Roy, "Feature-based methods for large scale dynamic programming," *Machine Learning*, vol. 22, no. 1, pp. 59–94, Mar 1996.

[5] D. Bertsekas, "Multiagent value iteration algorithms in dynamic programming and reinforcement learning," *Results in Control and Optimization*, vol. 1, p. 100003, 2020.

[6] N. Paul, T. Wirtz, S. Wrobel, and A. Kister, "Multi-agent neural rewriter for vehicle routing with limited disclosure of costs," *ArXiv*, 2022.

[7] X. Guo and B. Hu, "Exact formulas for finite-time estimation errors of decentralized temporal difference learning with linear function approximation," *ArXiv*, 2022.

[8] T. Doan, S. Maguluri, and J. Romberg, "Finite-time analysis of distributed TD(0) with linear function approximation on multi-agent reinforcement learning," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97. PMLR, June 2019, pp. 1626–1635.

[9] D. Bertsekas, *Reinforcement Learning and Optimal Control*, ser. Athena Scientific optimization and computation series. Athena Scientific, 2019.

[10] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[11] K. Gatsis, "Federated reinforcement learning at the edge: Exploring the learning-communication tradeoff," in *2022 European Control Conference (ECC)*, 2022, pp. 1890–1895.

[12] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[13] N. Vertovec, 2022. [Online]. Available: https://github.com/nikovert/distributed_traffic_routing