Abstract: The performance of multiagent systems depends heavily on information flow. As agents are populated more densely, some information flow can be redundant. Thus, there can be a tradeoff between communication overhead and control performance. To address this issue, the optimization of the communication topology for the consensus network has been studied. In this study, three different suboptimal topology algorithms are proposed to minimize the linear quadratic regulator (LQR) cost considering the communication penalty, since the optimal solution requires a brute-force search, which has exponential complexity. The first two algorithms were designed to minimize the maximum eigenvalue of the Riccati matrix for the LQR, while the third algorithm was designed to remove edges sequentially in a greedy manner through evaluating the LQR cost directly. The first and second algorithms differ in that the active edges of a consensus network are determined at the end of the iterations in the first, while sequentially in the second. Numerical evaluations show that the proposed algorithms reduce the LQR cost significantly by optimizing communication topology, while the proposed algorithm may achieve optimal performance with a properly chosen parameterization for a small consensus network. While the three algorithms show similar performance with the increasing number of agents, the quantized terminal cost matrix optimization (QTCMO) algorithm shows significantly less complexity within the order of several tenths than those of the other two algorithms.

Keywords: LQR; consensus; topology; optimization

1. Introduction

With advances in networking technology, communication speed and delay have been greatly improved. While the control system over networks used to focus on controlling a single device remotely over the network, a large number of devices are cooperatively operating together, which is often referred to as a multiagent system [1]. Multiagent systems are employed in many different industries, such as power networks [2], smart factories [3], and smart cities [4]. The control of this type of system can be classified as centralized, decentralized, and distributed. Even though centralized control can provide the best performance under ideal conditions, its operation is often limited by the computational complexity and communication overhead. While decentralized control is free from these limitations, its performance is possibly limited by the lack of sufficient information. Thus, distributed control, which uses partial information, has garnered attention recently [5].

Distributed control algorithms are known to have a tradeoff between performance and communication overhead. Some known benefits of distributed algorithms are enforced security, robustness to the failure of each agent, and distributed parallel computation [6]. Many different types of distributed algorithms have been developed from the constrained optimization theory using Lagrangian decomposition or Karush–Kuhn–Tucker (KKT) necessary conditions [6]. One particularly interesting field of a distributed control is consensus control, where each agent collectively determines its action based on the information from neighbors to achieve a common goal.

It has been extensively applied to the cooperative coordination of multiagent systems or a collective distributed estimation over multi-sites [7]. A linear quadratic regulator (LQR)
control has often been studied to find an optimal or suboptimal consensus algorithm. The optimal LQR control for consensus problems with single integrator dynamics and a single leader was shown to be given by the Laplacian matrix associated with a complete directed graph [8]. A suboptimal consensus LQR control for a hierarchical leader–follower network, where the leader sends information to the followers unidirectionally, was developed with state feedback and output feedback. The proposed algorithm was shown to synchronize nine rolls of a paper converting machine, where the first roll was set as a leader [9]. To save further communication costs, event-triggered control was also introduced to the cyber physical security problem [10].

The consensus problem has also been applied to the development of distributed estimation. An optimal and suboptimal Kalman Consensus Filter (KCF) was developed in the form of a distributed Kalman filter to have a consensus on a state estimate [11]. To overcome the limitation of KCF due to the lack of information on cross covariance and inconsideration of the naive node, which does not have valid observation, a generalized KCF (GKCF) was proposed in the form of achieving a weighted average consensus [12]. Surveys on consensus problems can be found in [13] for the static consensus problem, [14] for the dynamic consensus problem, and [15] for optimal consensus protocols.

Previous research focuses on the optimization of the network topology to minimize the convergence speed of the consensus or to sparsify the current topology to minimize communication overhead or improve the control characteristics, which is summarized in Table 1.

One of the major control problems associated with multiagent systems is to optimize the topology to improve control performance and/or communication overhead. Consensus algorithms have been developed to be tailored to specific topologies, such as the switching topology [16] and the general directed topology [17]. The optimal topology for the consensus problem of MASs with the single leader being stationary was shown to be the star topology for the first and second order dynamics [18]. Control issues associated with the interaction topology of the agents were surveyed for the formation control schemes [19]. Sufficient conditions for the mean square consensusability were formulated into mean square stabilization problems in consideration of delay, packet erasure, and network topology [20]. A sufficient condition for the asymptotic stability of a sparse networked control system with distributed observer-based controllers was given as a feasibility problem of a convex binary mixed margin program [21]. A delay-aware controller was also proposed to deal with communication delays and communication overhead, such that it can promote sparsity on information transmission and set the elements of control gain associated with nonarrival information due to delay [22]. A novel approach to deal with convergence speed was proposed from exploiting the community decomposition, such that topology optimization can be done through identification with the hub nodes of the decomposed topology [23].

Table 1. The objective and the criterion in existing literature.

| References | Objective | Criterion |
|------------|-----------|-----------|
| [18]       | Topology optimization | Minimization of the sum of weighted squared consensus error and weighted squared control input |
| [20]       | Feasibility condition of topology | Mean-square consensusability |
| [21]       | Sparsest topology for a distributed observer-controller network | Minimization of the number of edges with constraints on the control gain and stabilization conditions |
| [22]       | Sparsest topology for damping the power oscillations of wide area power systems | Enhancement of the closed-loop damping factor in consideration of participation factor |
| [23]       | Sparsification of the existing topology | Acceleration of the convergence speed of the consensus |
| Proposed Algorithms | Sparsification of the existing topology | Minimization of sum of the LQR cost and the number of edges |
Previous research focuses on the optimization of the network topology to minimize the convergence speed of the consensus, or simply to a communication overhead of the problem without considering the control cost. However, control cost is one of the most important design parameters to be considered in the design of a controller. Thus, we present three algorithms to optimize a network topology for a consensus problem through removing existing neighbor relations, taking into account the control cost. These algorithms reduce the number of communication links for information flows without creating a completely new topology. To this end, the proposed algorithms reduce the number of edges in consideration of the LQR cost, which is likely to provide a good tradeoff between the control cost and the communication overhead. This will be critical when a large number of multiagents are densely located since redundant information may be exchanged between neighbors, which results in the waste of communication and computing resources. The proposed algorithms optimize the tradeoff between control cost and communication overhead by minimizing the LQR cost. This is the main original contribution of this work.

This paper is organized as follows. In Section 2, a multiagent system with a single integrator dynamic and the problem formulation with the LQR cost are presented. Two iterative neighbor decision algorithms are proposed in Section 3. The numerical evaluations of the proposed algorithms are made in Section 4. Finally, concluding remarks are made in Section 5.

2. System Model and Problem Formulation

We consider a distributed control system consisting of \( N \) agents where each agent can communicate with neighbors only. It is assumed that the dynamics system of each agent follows an integrator system. The corresponding system model of the \( i \)th agent in a discrete-time system can be given as

\[
x_{i,k+1}^j = x_{i,k}^j + T_s u_{i,k}^j
\]

where \( T_s \) is the sampling period and \( u_{i,k}^j \) is the control signal of the \( i \)th agent. When \( u_{i,k}^j \) can only use the neighbor state information, it can be represented as

\[
u_{i,k}^j = - \sum_{j=1}^{N} a_{ij} h_{ij} (x_{i,k}^j - x_{j,k}^j)
\]

where \( a_{ij} \) has a nonzero value if \( j \) belongs to the set of neighbors of the agent \( i \), otherwise 0. \( h_{ij} \) is a binary variable which determines the structure of information flow. Stacking up its agent in a single vector, Equation (2) can be collectively represented as

\[
u_k = - \bar{L} x_k
\]

where \( \bar{L} = \sum_{j=1}^{N} a_{ij} h_{ij} \bar{L} \) and \( \bar{L} \) is the element in the \( i \)th row and the \( j \)th column of the matrix in the bracket. Similarly, the state vector in a stacked form can be found from Equations (1) and (3) as

\[
x_{k+1} = (I - T_s \bar{L}) x_k
\]

To determine \( h_{ij} \), we consider LQR control cost \( J_k \) which is defined as

\[
J_k = \sum_{k=0}^{k_T} \sum_{i=1}^{N} \sum_{j=1}^{N} (x_{i,k}^j - x_{j,k}^j)^T Q_{ij} (x_{i,k}^j - x_{j,k}^j) + \sum_{k=0}^{k_T} \sum_{i=1}^{N} u_{i,k}^j T R_{i,i} u_{i,k}^j + \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} h_{ij}
\]

where \( k_T \) is the terminal time, and \( \lambda \) is a parameter which controls the effect of controlling the number of neighbors.

It is noted that the first term in Equation (5) will be 0 when every agent reaches consensus. Thus, this term represents the quadratic difference in states between all possible
pairs of states. Since adding a constant term for a given $\lambda$ does not change the optimal solution, we consider the following constrained LQR cost as follows:

$$I_k = \sum_{k=0}^{k_T} \sum_{i=1}^{N} \sum_{j=1}^{N} (x_k^i - x_k^j)^T Q_{ij} (x_k^i - x_k^j) + \sum_{k=0}^{k_T} \sum_{i=1}^{N} u_k^i R_k u_k^i + \lambda \left( \sum_{i=1}^{N} \sum_{j=1}^{N} h_{ij} + \sum_{i=1}^{N} d_i \right)$$

(6)

$I_k$ can be rewritten with a vector form as

$$I_k = \sum_{k=0}^{k_T} x_k^T Q x_k + \sum_{k=0}^{k_T} u_k^T R u_k + \lambda \left( \sum_{i=1}^{N} \sum_{j=1}^{N} h_{ij} + \sum_{i=1}^{N} d_i \right)$$

(7)

where $[H]_{ij} = h_{ij}$, $[D]_{ij} = d_i > 0$, $[Q]_{ij} = \sum_{i=1}^{N} Q_{li} + \sum_{j=1}^{N} Q_{lj} + Q_{ij} = -Q_{ji}$, $[Q]_{ij,ji} = -Q_{ij}$, $[R]_{ij} = R_{ii}$, and $1$ is a column vector of length $N$ with all elements being 1.

Since $H$ has binary elements on nondiagonal positions, and 0 on the diagonal positions, the smallest eigenvalue will be larger than $-\sqrt{N}$. Thus, as long as $d_i \geq \sqrt{N}$, all $D + H$ will be a positive definite matrix. Even though $H$ seems to appear on the last term in Equation (6), it also has an effect on the control signal $u_k$, which is evident from Equation (2). Thus, an algorithm to minimize the LQR cost efficiently will be developed in the next section.

3. Topology Optimization Algorithms

In this section, we derive a neighbor survival algorithm which keeps some of the neighbors as they are while discarding others. To this end, the finite LQR cost can be represented through a value function for dynamic programming, which can be posed as

$$V_k = x_k^T Q x_k + \min_{u_k} (u_k^T R u_k + V_{k+1})$$

(8)

For the optimal $u_k$, $V_k$ is a quadratic function of the state $x_k$, which is given by

$$V_k = x_k^T P_k x_k$$

(9)

where $P_k$ is a nonnegative definite matrix for all $k$. From Equations (8) and (9), the LQR Riccati equation can be given as

$$P_k = Q + A^T P_{k+1} A - A^T P_{k+1} B (R + B^T P_{k+1} B)^{-1} B^T P_{k+1} A$$

(10)

In Equation (7), the last term can be considered as the terminal cost without a loss of generality, with the assumption that the average consensus is achieved. When the assumption holds, $x_k^{N+1} = \mu A$, which results in $\lambda 1^T (D + H) 1 = \lambda u^{-2} x_k^{N+1} (D + H) x_k^{N+1}$. Then, the LQR Riccati Equation (11) at the terminal step can be written as

$$P_k = Q + \bar{\lambda} A^T Y A - \bar{\lambda} A^T Y B (R + \bar{\lambda} B^T Y B)^{-1} B^T \bar{\lambda} Y A$$

(11)

where $\bar{\lambda} = \lambda u^{-2}$, and $Y > 0$. In addition, an upper bound of $V_k$ can easily be derived from the trace inequality of $tr(XY) \leq tr(X) tr(Y)$ for $X \geq 0$ and $Y \geq 0$ as

$$V_k \leq N p(\rho) tr \left( (x_k x_k^T) \right)$$

(12)

where $\rho(\cdot)$ is the spectral radius of the matrix in the bracket.

Furthermore, the method of representing the Riccati operator for the LQR problem in [24] is exploited:

$$\psi(F, X) = (A + F B) X (A + F B) + Q + FRF^T$$

(13)
When \( F = \overline{A}^T \overline{X} \overline{B} (R + \overline{B}^T P_N \overline{B})^{-1} \) and \( X = P_N \), it corresponds to the Riccati inequality for the LQR problem at the \( N-1 \)th step. For given \( X, F_X \), a solution for minimizing \( \psi(F, X) \) can easily be calculated from a necessary condition for the quadratic equation as

\[
F_X = \overline{A}^T \overline{X} \overline{B} (R + \overline{B}^T P_N \overline{B})^{-1}
\]  

(14)

It is noted that \( \psi(F_X, X) \) is a monotonically increasing function of \( X \), such that \( \psi(F_X, X) \leq \psi(F_X, Y) \) for \( X \leq Y \). From this property, we have the following inequality:

\[
\psi(F_X, X) \leq \psi(F_Y, X) \leq \psi(F_Y, Y)
\]  

(15)

This implies that \( \min_F \psi(F, X) \) is a monotonically increasing function of \( X \) in the following way [24]:

\[
\psi(F_X, X) \leq \psi(F_Y, X)
\]  

(16)

The upper bound of \( J_{kT} \) is given by the upper bound of \( V_0 \) from Equations (11) and (15) as

\[
J_{kT} \leq N \rho(P_0) \operatorname{tr} \left( x_0 x_0^T \right) \leq N \rho(P_0) \operatorname{tr} \left( x_0 x_0^T \right)
\]  

(17)

Thus, we are interested in minimizing the upper bound of \( J_{kT} \) in Equation (16), rather than minimizing \( J_{kT} \) directly, since the original minimization problem is a set of functions involved over time. The corresponding optimization problem can be posed as

\[
\min_{Y, \alpha} \quad Q + \lambda \overline{A}^T \overline{Y} \overline{A} - \lambda \overline{A}^T \overline{Y} \overline{B} (R + \overline{B}^T \overline{Y} \overline{B})^{-1} \overline{B}^T \overline{Y} \overline{A} \leq \alpha I
\]

\[
Y = D + H, [H]_{ij} \in \{0,1\}, i,j \in \{1,2,\ldots,N\}
\]  

(18)

One direct method to find the solution is to try every possible \( H \), which results in excessive complexity as the number of agents increases. Thus, we propose an algorithm which iteratively updates the matrix and applies binary decisions to the matrix at the end of each iteration. The proposed algorithm, which we call a quantized terminal cost matrix optimization (QTCMO) algorithm, is summarized in Figure 1. It starts with initializing a nonnegative definite matrix \( Y \) as \( D \). It also defines a set \( S \), such that all indices corresponding to the edge between neighbor nodes can be included. That is, \( S \) represents a set of indices of the elements in \( Y_0 \) which are optimizing variables. For a given \( Y, F \) is selected to minimize \( \psi(F, Y) \) in step 1. For a given \( F, Y \) is determined to minimize \( \psi(F, Y) \) further in step 2.

In step 2, several constraints are made so that the performance of the proposed algorithm can be improved through limiting the feasible regions. \( \sum_{l_2=1}^{N} [H]_{l_1,l_2} \geq 1 \) is imposed so that each node can have at least one neighbor. \( [H]_{l_1,l_2} \geq 0, [H]_{l_1,l_2} \leq 1, (l_1, l_2) \in S \) is imposed to reduce the quantization error in later stages. Repeating steps 1 and 2 monotonically decreases \( \psi(F, Y) \). When it converges, an element-wise rounding operation denoted as \( \text{round}() \) is applied so that the resulting element in \( H^* \) is 0 or 1. The condition that the number of edges is at least 1 does not guarantee the connectivity of \( H^* \). Thus, it sequentially adds edges. To this end, \( H_S = \sum_{l=1}^{N-1} H^{\alpha l} \) is computed. The nonzero elements of \( H_S \) represent the connectivity through direct or indirect neighbors. Thus, the column of \( H_S \) with the largest number of zeros and the row having 0 at the intersection with that column are selected so that the corresponding position of \( H_S \) is set to 1. This procedure continues until \( H^* \) belongs to the set of connected subgraphs of \( A \).
Figure 1. A quantized terminal cost matrix optimization algorithm.

Proposition 1. The Algorithm in Figure 1 Converges Locally.

Proof of Proposition 1. $\psi(F, Y)$ monotonically decreases with repeating step 1 and step 2 due to Equation (15). $\psi(F, Y)$ is lower bounded by $\psi(F_0, Y^*)$ where $Y^*$ is the optimal solution for Equation (18). Thus, the local convergence is guaranteed. □

Even though the proposed algorithm is guaranteed to converge, the quantization process may degrade the performance significantly when the resulting elements of $H_{i+1}$ at the end of each iteration are distributed uniformly. To deal with this issue, a greedy terminal cost matrix optimization (GTCMO) algorithm is proposed in Figure 2. It initializes and updates $F$ and $H_{i+1}$ in exactly the same way as in the QTCMO algorithm. Then, the element of $H_{i+1}$ with the largest magnitude and its symmetric element are set as 1 in step 3. Then, step 4 enforces the elements of $H_{i+1}$ having a value of less than $\tilde{\varepsilon}$ to be 0. Thus, iterating these steps reduces the number of optimizing variables in a greedy way. The algorithm stops when the number of optimizing variables becomes 0. Even though this algorithm is guaranteed to stop with finite iterations, there is no guarantee on the monotonicity of $Y^*$ due to the nonlinearity associated with step 3 and step 4. While the QTCMO algorithm applies quantization at the end of the iteration, the GTCMO algorithm is considered to apply the quantization sequentially where each quantization is made adaptively to prior
quantization. The same procedure to add an edge to make \( H^* \) a connected graph as the one in the QTCMO is applied.

**Figure 2.** A greedy terminal cost matrix optimization algorithm.

\[
\text{step 0: Initialization} \\
Y_0 = D, \ i = 0 \\
S = \{(l_i, l_j) \in \mathbb{N}^2 | a_{l_i, l_j} \neq 0, l_i, l_j \in \{1, 2, \cdots N\}\}
\]

\[
\text{step 1: update } F \\
F = \tilde{A} \tilde{Y} \tilde{B} (R + \tilde{A} \tilde{B}^T \tilde{Y} \tilde{B})^{-1}
\]

**step 2:** update \( Y_{i+1} \)

\[
H_{i+1} = \arg_{\alpha} \min \psi(F, \tilde{Y}_i, Y_i, \alpha) \geq Y \geq 0, \\
\sum_{l_i, l_j} [H]_{l_i, l_j} \geq 1, \forall (l_i, l_j) \in S, \\
[H]_{l_i, l_j} \geq 0, [H]_{l_i, l_j} \leq 1, \forall (l_i, l_j) \in S \\
Y_{i+1} = D + H_{i+1}
\]

**step 3:** Freeze the largest element in \( H_{i+1} \)

\[
(l_i^*, l_j^*) = \max_{H_{i+1}} H_{i+1} \\
Y_{i+1} = \{ (l_i^*, l_j^*) \}
\]

\[
S = S - \{ (l_i^*, l_j^*) \}
\]

**step 4:** Freeze the elements with small magnitude in \( H_{i+1} \)

\[
S_0 = \{ (l_i, l_j) \} | [H_{i+1}]_{l_i, l_j} \leq \delta, (l_i, l_j) \in S \} \\
[H_{i+1}]_{l_i, l_j} = 0 \forall (l_i, l_j) \in S_0 \\
S = S - S_0
\]

**step 5:** Determine the completion

if \( |S| = 0 \), \( H^* = H_{i+1} \), go to step 6
else, \( i = i + 1 \), go to step 1

**step 6:** Guarantee the connectivity

\[
H_S = \sum_{i=1}^{N-1} H^{i,i} \\
\text{while } H^* \neq \mathcal{C}(A) \\
[l_i^*] = \arg \min_{H_{S}} \sum_{l_j} I(l_j)(H_{S})_{l_i, l_j} \\
[l_j^*] = \arg \min_{H_{S}} \sum_{l_i} I(l_i)(H_{S})_{l_i, l_j} \\
[H_{S}]_{l_i, l_j} = 1
\]

end

One may try to optimize the structure directly with an increase in complexity. A sequential edge removal (SEQ) algorithm is summarized in Figure 3. It starts by initializing a set of the edges for a given \( A \). \( \tilde{A} \) is an optimizing adjacent matrix, is initialized as \( A \). The LQR cost with an edge removed is calculated for each edge, guaranteeing the connectivity of a network without it. Let a set of connected subgraphs of \( A \) be denoted by \( \mathcal{C}(A) \). In step 1, the LQR cost is \( I(\tilde{A}(i,j)) \) where \( \tilde{A}(i,j) \) is the same as \( \tilde{A} \), except the edge connecting node i and node j being removed. The edge for the removal is determined so
that the removal of that edge can result in the minimization of the LQR cost. This process repeats as long as the sequentially selected edge for removal reduces the LQR cost. The maximum number of iterations is guaranteed to be less than $n^2$. The complexity associated with checking out the connectivity after the edge removal and the LQR cost is likely to be significant.

**Figure 3.** Sequential edge removal algorithm.

The objective and summary of the implementation procedures are summarized in Table 2.

**Table 2.** The summary of the three proposed algorithms.

| Algorithms | Objective | Summary of Procedure |
|------------|-----------|----------------------|
| QTCMO      | $\min_{F,Y} \varphi(F,Y)$ | Update $F$ and $Y$ iteratively until convergence  
Quantize the element of $H$ as 1 or 0  
If the network topology is not connected, add edges |
| GTCMO      | $\min_{F,Y} \varphi(F,Y)$ | Update $F$ and $Y$  
Quantize the maximal element in $H$ as 1  
Quantize the small elements in $H$ as 0  
Repeat it until convergence  
If the network topology is not connected, add edges |
| SER        | $\min H_{k_f}$ | Find an edge for exclusion  
Repeat it until convergence |

**4. Evaluations of the Algorithms**

In this section, the proposed algorithms are evaluated through numerical simulations. Two sets of numerical experiments are considered. The first set is for characterizing the proposed algorithm with a simple network configuration shown in Figure 4. The second set is to characterize the average performance for randomly generated network configuration. The LQR cost, total number of edges, and computational complexity will be evaluated.
Figure 4. A network consisting of 5 nodes and 8 edges.

Figure 4 shows a network consisting of five nodes and eight edges. Even though it is not unique, the simplest connected subgraph has four edges. Thus, some edges can be redundant from the perspective of the LQR cost. To provide the baseline performance of the proposed algorithms, the optimal brute-force search method is considered. This method finds the connected subgraph with the minimum LQR cost through evaluating all the possible connected subgraphs. The total number of edges of the proposed algorithms are provided in Table 3. Both the optimal method and the GTCMO always find the subgraph with the smallest number of edges regardless of $\lambda$. They also provide the smallest LQR cost even though they are not discernable in Figure 4. The QTCMO algorithm is considered to work in a coarse way in the sense that the total number of edges is the same as the one in the optimal method for a large $\lambda$, while it keeps all edges for a small $\lambda$. The SER method is found to be more responsive to $\lambda$. Even though the QTCMO and SER fail to find the optimal sparse network configuration for small $\lambda$, all the proposed algorithms are shown to provide optimal performance for a large $\lambda$. Figure 5 shows that all methods provide almost the same cost. Since the QTCMO fails to find the optimal number of edges for a small $\lambda$, its cost is found to be relatively larger than the optimal cost by 24.1% maximum when $\lambda = 1$.

Table 3. The resulting total number of edges with the proposed sparsifying algorithms for $\lambda$.

| $\lambda$ | Opt. | QTCMO | GTCMO | SER |
|----------|------|-------|-------|-----|
| $10^{-4}$ | 4    | 8     | 4     | 8   |
| $10^{-3}$ | 4    | 8     | 4     | 8   |
| $10^{-2}$ | 4    | 8     | 4     | 6   |
| $10^{-1}$ | 4    | 8     | 4     | 4   |
| $10^0$    | 4    | 8     | 4     | 4   |
| $10^1$    | 4    | 4     | 4     | 4   |
| $10^2$    | 4    | 4     | 4     | 4   |
| $10^3$    | 4    | 4     | 4     | 4   |

To find out the average characteristic of the proposed algorithms, nodes were uniformly generated such that their location could be uniformly distributed over the square with a length of 1, where nodes within the squared distance of 0.5 were set as neighbors. Two hundred different realizations were made so that the average number of edges, average cost, and average processing time could be calculated. $\lambda$ was set to 1. Since the complexity of the optimal method is prohibitive, the proposed methods only are compared.

Figure 6 shows the average number of edges resulting from applying the proposed algorithms. It is observed that the average number of edges resulting from applying the proposed algorithms is reduced. All the proposed algorithms are found to sparsify the network by almost same amount, even though the resulting networks are often different. The ratio of reduction is found to increase from 28.3% to 78.7% as $N$ increases since the number of neighbors for a given node tends to increase with $N$. 
Figure 5. The LQR cost of the proposed algorithm for $\lambda$.

Figure 6. Average number of edges for different $N$ values.

The average cost resulting from each algorithm is compared in Figure 7. The larger the number of edges, the larger the cost to find the result. Since all the proposed algorithms provide almost the same average number of edges, the costs associated with the proposed algorithms are almost same. Figure 8 shows the computational complexity of the proposed algorithms in average processing time. The average processing time of the QTCMO is found to be the lowest. The number of iterations does not increase much with $N$. Thus, the increase in complexity depends on the complexity of the optimization with a linear matrix inequality. On the contrary, the complexity of the GTCMO is the largest. The number of iterations associated with the greedy procedure depends on the number of edges, which is proportional to $N$, whereas the complexity per iteration is larger than that of the QTCMO due to the additional step. Even though the complexity of the SER algorithm is in the middle, it heavily depends on the number of control steps. Thus, as the number of control steps increases, its complexity is likely to increase proportionally.
The two numerical experiments show that all proposed algorithms provide nearly similar performance as long as \( \lambda \) is chosen properly. As long as the control step is not too large, the QTCMO algorithm can be considered to perform well with the least complexity.

5. Conclusions

In this paper, three different algorithms for sparsifying the communication network for the average consensus problem were proposed. The proposed algorithms were designed to minimize the LQR cost through either minimizing the Riccati matrix at the terminal step or removing edges in a greedy manner. An experiment over a simple hypothetical network has shown that all the proposed algorithms show near optimal LQR performance. Another experiment showed that all the proposed algorithms provide almost similar performance, while the QTCMO algorithm can be implemented with the least complexity.

Even though a simple integrator model was considered in this paper, an extension to a conventional linear system model can be easily developed. In addition, by considering
disturbance and measurement noise using LQG control, this work can be extended to dynamic consensus problems, such as vehicle formations.

One interesting way to address the complexity of the topology optimization of consensus networks is likely to exploit state-of-the-art methods in a genetic algorithm (GA), which is found to provide an efficient solution to complex problems, such as the traveling salesman problem [25]. An articulated topology optimization may be developed by exploiting the selection and mutation schemes [26] or Mendel’s rule of heredity [27] in a GA, which should be given attention as a possible direction for future research.

Author Contributions: Conceptualization, J.Y. and Y.C.; methodology, J.Y.; software, Y.C.; formal analysis, J.Y.; writing—original draft preparation, J.Y.; writing—review and editing, Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Basic Science Research Program through the National Research Foundation of Korea by the Ministry of Science and ICT under Grant NRF-2017R1A2B4007398.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References
1. Deloach, S.A.; Wood, M.F.; Sparkman, C.H. Multiagent Systems Engineering. Int. J. Softw. Eng. Knowl. Eng. 2001, 11, 231–258. [CrossRef]
2. McArthur, S.D.J.; Davidson, E.M.; Catterson, V.M.; Dimeas, A.L.; Hatziargyriou, N.D.; Ponci, F.; Funabashi, T. Multi-Agent Systems for Power Engineering Applications—Part I: Concepts, Approaches, and Technical Challenges. IEEE Trans. Power Syst. 2007, 22, 1743–1752. [CrossRef]
3. Wang, S.; Wan, J.; Zhang, D.; Li, D.; Zhang, C. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. Comput. Netw. 2016, 101, 158–168. [CrossRef]
4. Roscia, M.; Longo, M.; Lazaro, C.G. Smart City by multi-agent systems. In Proceedings of the International Conference on Renewable Energy Research and Applications (ICRERA), Madrid, Spain, 26–29 September 2013.
5. Zhu, Y.; Yang, F.; Li, C.; Zhang, Y. Simultaneous Stability of Large-scale Systems via Distributed Control Network with Partial Information Exchange. Int. J. Control. Autom. Syst. 2018, 16, 1502–1511. [CrossRef]
6. Molzahn, D.K.; Dorfler, F.; Sandberg, H.; Low, S.H.; Chakrabarti, S.; Baldick, R.; Lavaei, J. A Survey of Distributed Optimization and Control Algorithms for Electric Power Systems. IEEE Trans. Smart Grid 2017, 8, 2941–2962. [CrossRef]
7. Garin, F.; Schenato, L. A Survey on Distributed Estimation and Control Applications Using Linear Consensus Algorithms. Sens. Control Auton. Veh. 2010, 406, 75–107.
8. Cao, Y.; Ren, W. Optimal Linear-Consensus Algorithms: An LQR Perspective. IEEE Trans. Syst. Man Cybern. Part. B (Cybern.) 2010, 40, 819–830.
9. Nguyen, D.H. A sub-optimal consensus design for multi-agent systems based on hierarchical LQR. Automatica 2015, 55, 88–94. [CrossRef]
10. Li, X.; Zhou, Q.; Li, P.; Li, H.; Lu, R. Event-Triggered Consensus Control for Multi-Agent Systems Against False Data-Injection Attacks. IEEE Trans. Cybern. 2019, 50, 1856–1866. [CrossRef] [PubMed]
11. Olfati-Saber, R. Kalman-Consensus Filter: Optimality, Stability, and Performance. In Proceedings of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, China, 15–18 December 2009.
12. Kamal, A.T.; Ding, C.; Song, B.; Farrell, J.A.; Roy-Chowdhury, A.K. A Generalized Kalman Consensus Filter for Wide-Area Video Networks. In Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 12–15 December 2011.
13. Olfati-Saber, R.; Fax, J.A.; Murray, R.M. Consensus and Cooperation in Networked Multi-Agent Systems. Proc. IEEE 2007, 95, 215–233. [CrossRef]
14. Kia, S.S.; Van Scoy, B.; Cortes, J.; Freeman, R.A.; Lynch, K.M.; Martinez, S. Tutorial on Dynamic Average Consensus: The Problem, Its Applications, and the Algorithms. IEEE Control. Syst. 2019, 39, 40–72. [CrossRef]
15. Sun, H.; Liu, Y.; Li, F.; Niu, X. A survey on optimal consensus of multi-agent systems. In Proceedings of the Chinese Automation Congress (CAC), Jinan, China, 22 October 2017.
16. Zhang, H.; Yang, R.; Yan, H.; Yang, F.H. ∞ consensus of event-based multi-agent systems with switching topology. Inf. Sci. 2016, 370–371, 623–635. [CrossRef]
17. Wang, H.; Yu, W.; Wen, G.; Chen, G. Fixed-Time Consensus of Nonlinear Multi-Agent Systems with General Directed Topologies. IEEE Trans. Circuits Syst. II Express Briefs 2019, 66, 1587–1591. [CrossRef]
18. Ma, J.; Zheng, Y.; Wang, L. LQR-based optimal topology of leader-following consensus. Int. J. Robust Nonlinear Control 2014, 25, 3404–3421. [CrossRef]
19. Oh, K.-K.; Park, M.-C.; Ahn, H.-S. A survey of multi-agent formation control. *Automatica* 2015, 53, 424–440. [CrossRef]
20. Zheng, J.; Xu, L.; Xie, L.; You, K. Consensusability of Discrete-Time Multiagent Systems with Communication Delay and Packet Dropouts. *IEEE Trans. Autom. Control.* 2019, 64, 1185–1192. [CrossRef]
21. Razeghi-Jahromi, M.; Seyedi, A. Stabilization of Networked Control Systems with Sparse Observer-Controller Networks. *IEEE Trans. Autom. Control.* 2015, 60, 1686–1691. [CrossRef]
22. Dibaji, S.M.; Annaswamy, A.; Chakrabortty, A.; Hussain, A. Sparse and Distributed Control of Wide-Area Power Systems with Large Communication Delays. In Proceedings of the Annual American Control Conference, Wisconsin Center, Milwaukee, WI, USA, 27–19 June 2018.
23. Zhang, M.M.; Yun, H.Q.; Ju, W. The Topology Optimization Rule for Multi-Agent System Fast Consensus. In Proceedings of the 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), Xiamen, China, 10–12 August 2018.
24. Gupta, V.; Murray, R.M.; Shi, L.; Sinopoli, B. *Networked Sensing, Estimation and Control Systems*; Technical Report; Department of Control and Dynamical Systems, California Institute of Technology: Pasadena, CA, USA, 2009.
25. Singh, G.; Gupta, N.; Khosravy, M. New crossover operators for real coded genetic algorithm (RCGA). In Proceedings of the 2015 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), Okinawa, Japan, 28–30 November 2015; pp. 135–140.
26. Gupta, N.; Patel, N.; Tiwari, B.N.; Khosravy, M. Genetic Algorithm Based on Enhanced Selection and Log-Scaled Mutation Technique. In Proceedings of the Future Technologies Conference, Vancouver, BC, Canada, 15–16 November 2018.
27. Gupta, N.; Khosravy, M.; Patel, N.; Dey, N.; Mahela, O.P. Mendelian evolutionary theory optimization algorithm. *Soft Comput.* 2020, 24, 1–46. [CrossRef]