Optimal Transportation on Directed Tree Graphs

Heyden, Martin; Pates, Richard; Rantzer, Anders

Published in:
Under Review

2020

Document Version:
Early version, also known as pre-print

Link to publication

Citation for published version (APA):
Heyden, M., Pates, R., & Rantzer, A. (2020). Optimal Transportation on Directed Tree Graphs. Manuscript submitted for publication.

Total number of authors:
3

Optimal Transportation on Directed Tree Graphs

Martin Heyden, Richard Pates, and Anders Rantzer, *Fellow, IEEE*

***Abstract*—We consider the problem of optimal transportation and production of a quantity throughout a network, where the transportation is subject to delay. It is shown that for a simple network model the optimal policy is sparse and highly structured. The results hold for a broad class of convex cost functions, and in the quadratic case we give closed form expressions for the optimal flows and for the optimal production. The optimal controller can be both synthesized and implemented using distributed communication, making it suitable for large scale applications. The performance of the optimal controller is studied for networks of different sizes and topologies, and compared to a local controller designed using off-the-shelf methods. The optimal controller gives a significant increase in performance for non-zero initial conditions.**

I. Introduction

Whether it be to maximize throughput in a traffic network, minimize losses in an electrical power system, or improve fairness when managing Internet congestion, large-scale systems are typically operated with some notion of performance in mind. Many such networks are also constructed out of dynamical components, and a fundamental challenge is to operate them in a manner that maximizes performance. In many cases, including those listed above, this balance is struck by designing control schemes to stabilize the system about *an equilibrium point* that is chosen to optimize the network's given measure of performance (e.g. [1]–[3]).

However in most cases it could be argued that these large-scale systems are in fact never in equilibrium for long, instead shifting from operating point to operating point to balance the current demands of the users. There is of course still a value to shifting toward equilibria which optimize performance, but it is also clear that the notion of *dynamic performance* is an an important one. Take for example electrical power systems; the objective is really to minimize losses throughout operation, rather than the losses associated with particular operating points.

In this work we address this aspect by studying a simple model for production and transportation of some quantity, as illustrated in Fig. 1. We will consider just one commodity, however the results presented could be extended to the case of multiple commodities. The objective is to design an optimal controller to solve a natural *dynamic* extension of the classical welfare maximization problem in economics.

This work was supported by the Swedish Foundation for Strategic Research through the project SSF RIT15-0091 SoPhy.

The authors are members of the LCCC Linnaeus Center and the ELLIIT Excellence Center at Lund University.

The authors are with the Department of Automatic Control, Lund University, Box 118, SE-221 00 Lund, Sweden. (e-mail: {martin.heyden, richard.pates, rantzer}@control.lth.se).

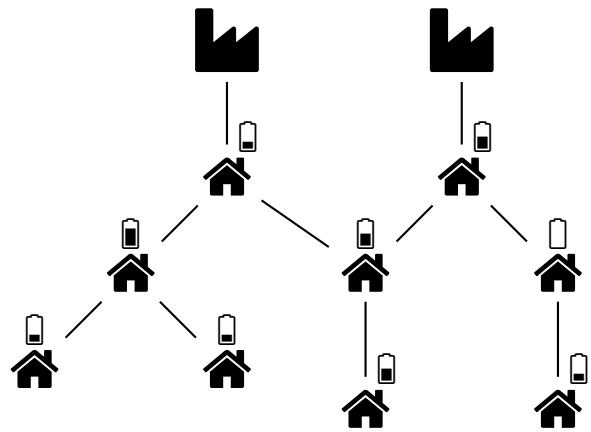


Fig. 1: An illustration of the type of problems studied in this paper. At the top of the network is a production plant that produces some commodity. The commodity should be optimally distributed to all the users in the system.

A. Problem Formulation

In welfare maximization, the goal is to optimally distribute a number of commodities among a set of agents so that the total social welfare is maximized. There are many ways of defining the social welfare (see [4] for an introduction). In this work we will consider the classical utilitarian welfare functions, where the total welfare is the sum of the utility for all agents. Furthermore the individual welfare functions are individualistic, meaning that the utility of the agents only depends on the agents own consumption (the amount of the quantity it has at each point in time). To capture dynamic aspects, we introduce constraints that model the effect of transportation delays between the agents, as structured by a graph.

Assume that each agent i starts with an initial amount w_i of the commodity, and the utility for node i in having an amount z_i of the commodity is given by $U_i(z_i)$. Then the welfare maximization problem can be formulated as

$$\begin{aligned} \underset{z_i}{\text{maximize}} \quad & \sum_i U(z_i) \\ \text{subject to} \quad & \sum_i z_i = \sum_i w_i. \end{aligned} \tag{1}$$

As we can see, the objective is to maximize the collective utility, subject to the constraint that the new levels respects the initial endowments.

We will now present a natural dynamic extension of this standard welfare maximization problem by only allowing for the commodity to be transported with a delay between agents that are connected. To describe the topology of the network

we introduce a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of a set of vertices $\mathcal{V} = \{1, \ldots, N\}$ and directed edges $\mathcal{E}$, where $(i, j) \in \mathcal{E}$ if there exists a path from node j to node i. We also introduce the standard notation of parent set $\mathcal{P}(j)$ and children set $\mathcal{C}(j)$ for every node j. That is $k \in \mathcal{P}(j)$ if $(j, k) \in \mathcal{E}$ and $i \in \mathcal{C}(j)$ if $(i, j) \in \mathcal{E}$.

Let $z_i[t]$ be the amount of the quantity held by agent i at time t. We will consider the following dynamical system, which captures the essence of transportation subject to delay throughout a network. At its heart this model is a conservation equation, and the idea is that this simple system can describe the basic features of transportation in a wide range of settings (it can for example be used as a simple model for supply chains [5] and water irrigation networks [6]):

$$z_i[t+1] = \alpha \left(z_i[t] + \sum_{j \in \mathcal{P}(i)} u_{ij}[t-1] \right) - \sum_{h \in \mathcal{C}(i)} u_{hi}[t] \quad (2)$$

This equation describes how $z_i[t]$ evolves over time. Each input $u_{ij}[t] \in \mathbb{R}$ denotes the amount of the quantity being transported from node j to node i (associated with the edge $(i, j) \in \mathcal{E}$). All variables are relative to an equilibrium. The constant $\alpha,\ 0 < \alpha \leq 1$, is the decay rate of the stored quantity. The first summation therefore gives the amount of the quantity arriving at the *i*-th node, and the second summation gives the amount leaving the *i*-th node. The one step delay in the terms in the first summation captures the delay in transportation.

We will also allow for production at some nodes in the network. We denote the set of all such nodes as I and denote the production as $u_{i0},\ i \in I$. The zero node is not part of the network, $0 \notin \mathcal{V}$. However, in a slight abuse of notation, we let $0 \in \mathcal{P}(i)$ if $i \in I$. This allows external production to be described in a manner consistent with the dynamics in (2). We will assume that only nodes that are in the top of the network can be producers. This will be made precise in Section II. Note that (2) is not a state-space model, however a realization with states $\{z_i[t], u_{ij}[t-1],\ j \in \mathcal{P}(i)\},\ \forall i$, can be introduced.

We study the following dynamic extension of the welfare maximization problem in (1):

$$\begin{aligned} \underset{u,z}{\text{minimize}} \quad & \sum_{t=0}^{\infty} \left(\sum_{i \in \mathcal{V}} f_i(z_i[t]) + \sum_{i \in I} g_i(u_{i0}[t]) \right) \\ \text{Subject to} \quad & \text{Dynamics in (2).} \end{aligned} \quad (3)$$

In the above $f_i(z_i[t])$ is the cost (or negative utility) for node i to have access to $z_i[t]$ goods. The problem corresponds to minimizing the total cost[1] for the entire system, which is a welfare maximization problem for one commodity. The function $g_i(u_{i0})$ is the cost for node i to produce u_{i0}. Note the major difference between the problems in (1) and (3) is the static constraint in (1) has been replaced by its dynamic analogue from (2).

To ensure that we obtain a unique solution, we make the following assumptions.

[1]Note that we choose to minimize the total cost instead of maximizing the total utility.

Assumption 1. *The functions f_i and g_i are strictly convex and satisfy $f_i(0) = 0$ and $g_i(0) = 0$.*

Assumption 2. *The graph is a directed tree (also known as a poly-tree). This means that it is a directed graph with no undirected cycles (the undirected version of the graph is a tree).*

We will show that the problem in (3) can be simplified for general strictly convex cost functions. However, for quadratic cost functions the solution simplifies greatly. Our main contribution is to show that the solution to the optimal control problem

$$\begin{aligned} \underset{u,z}{\text{minimize}} \quad & \sum_{t=0}^{\infty} \left(\sum_{i \in \mathcal{V}} q_i z_i[t]^2 + \sum_{i \in I} r_i u_{i0}[t]^2 \right) \\ \text{Subject to} \quad & \text{Dynamics in (2),} \end{aligned} \quad (4)$$

where $q_i > 0$ and $r_i > 0$, admits a highly structured solution (contrary to the standard linear quadratic regulator problem), and explain how these results can be generalized to the case of general convex functions. Note that the cost function in (4) satisfies Assumption 1.

B. Preview of Results

The key feature of our results is that they show how the structure in the underlying graph can be expoloited when solving the dynamic welfare problem described in the previous subsection. In particular, we will show that the optimal control on each edge (i, j) will be dependent on two sets $U(i, j)$ and $D(i, j)$, which we will call the upstream set and the downstream set. These sets are subsets of the nodes of the graph. They are different for each edge, and reflect the local structural properties of the graph. The optimal controller can then be calculated according to the formula

$$\begin{aligned} u_{ij}[t] = & \frac{\gamma_{U(i,j)}}{\gamma_{U(i,j)} + \gamma_{D(i,j)}} \alpha m_{U(i,j)}[t] \\ & - \frac{\gamma_{D(i,j)}}{\gamma_{U(i,j)} + \gamma_{D(i,j)}} \alpha m_{D(i,j)}[t]. \end{aligned} \quad (5)$$

The constants $\gamma_{U(i,j)}$ and $\gamma_{D(i,j)}$ only depend the problem data within their respective set, and can be computed ahead of time. The variable $m_{U(i,j)}$ is the total quantity stored in the nodes in the upstream set, and $m_{D(i,j)}$ is the total quantity stored in the nodes in the downstream set. The definition of the upstream and downstream sets will be made precise in the next section. However to get the intuition, consider Fig. 2. Taking u_{84} as an example, the the upstream set is indicated by blue and the downstream set by pink. Observe in particular that some nodes are neither in the upstream nor the downstream set. This implies the optimal control law is sparse (with sparsity defined by the elements in the up and downstream sets). The structure of the resulting controller for the graph in Fig. 2 is illustrated in Fig. 3. Furthermore, both the γ parameters and the aggregate levels m can be calculated via a sweep through the graph, relying on only local communication. This allows for efficient synthesis and implementation of the control law in (4).

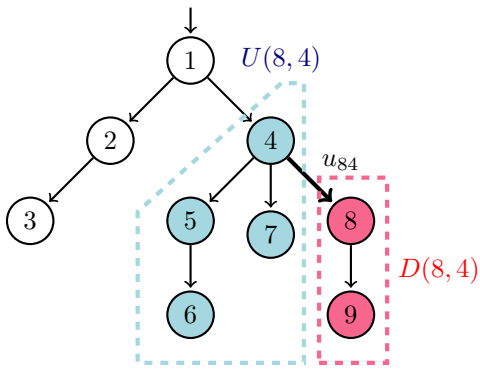


Fig. 2: An example of a directed tree. Node Four has one parent node in node one, and three children nodes in node five seven and eight. For the highlighted edge $(8,4)$ we have illustrated the upstream set U in light blue and downstream set D in pink. The incoming arrow into node one indicates that there is production in that node.

$$\begin{bmatrix} u_{10}[t] \\ u_{21}[t] \\ u_{32}[t] \\ u_{41}[t] \\ u_{54}[t] \\ u_{65}[t] \\ u_{74}[t] \\ u_{84}[t] \\ u_{98}[t] \end{bmatrix} = \begin{bmatrix} * & * & * & * & * & * & * & * & * \\ \star & \blacklozenge & \blacklozenge & \star & \star & \star & \star & \star & \star \\ 0 & \star & \blacklozenge & 0 & 0 & 0 & 0 & 0 & 0 \\ \star & \star & \star & \blacklozenge & \blacklozenge & \blacklozenge & \blacklozenge & \blacklozenge & \blacklozenge \\ 0 & 0 & 0 & \star & \blacklozenge & \blacklozenge & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \blacklozenge & 0 & 0 & 0 \\ 0 & 0 & 0 & \star & \star & \star & \blacklozenge & \star & \star \\ 0 & 0 & 0 & \star & \star & \star & \star & \blacklozenge & \blacklozenge \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \star & \blacklozenge \end{bmatrix} \begin{bmatrix} u_{10}[t-1] + z_1[t] \\ u_{21}[t-1] + z_2[t] \\ u_{32}[t-1] + z_3[t] \\ u_{41}[t-1] + z_4[t] \\ u_{54}[t-1] + z_5[t] \\ u_{65}[t-1] + z_6[t] \\ u_{74}[t-1] + z_7[t] \\ u_{84}[t-1] + z_8[t] \\ u_{98}[t-1] + z_9[t] \end{bmatrix}$$

Fig. 3: Structure of optimal feedback law for the polytree in Fig. 2. $\star$ corresponds to the upstream set and $\blacklozenge$ to the downstream set.

Similarly, the optimal production for (4) is given on the simple form

$$u_{i0}[t] = cm_{\mathcal{V}}[t]. \tag{6}$$

In the above c is a constant depending on the problem data for the entire graph and $m_{\mathcal{V}}[t]$ is the total level in the graph at time t.

C. Literature Review

While we give economic motivations for the problem studied, the main contribution lies in showing that the optimal controller is structured. It is these structural features that allows for an efficient implementation, even for large-scale systems. This is in contrast with standard methods for control synthesis, such as the linear quadratic controller, which in general scales poorly with size, both in terms of synthesis and implementation.

The lack of scalability of standard control methods has prompted a great deal of research within control community on optimal control under constraints on the controller structure (e.g. sparsity). Early work includes team decision problems, where multiple decisions that have access to different information needs to be made. See for example [7]. One approach to finding structured control is to enforce the structure on the controller. An important contribution to this approach was given in [8], where the notation of Quadratic Invariance is used to give a condition under which the structured controller synthesis problem can be recasted as a convex optimization problem. The role of convexity has been a recurring theme, consider for example the work on system level synthesis [9], [10] and network realizability [11], where the problem of finding the optimal control can again be recast as a convex optimization problem.

In [12] it is shown that for spatially interconnected systems, the optimal distributed controller can be found by solving linear matrix inequalities. In [13] an optimal controller for partially ordered sets is found by solving a set of independent Riccati equations. The structure of the controller is similar to ours, however the information flow is in the opposite direction. In [14] a method for distributed synthesis of a distributed LQ controller is derived by constructing local estimates for the gradients of the global cost function. A slightly different approach is taken in [15], where structured controllers are synthesized by posing an optimization problem with a sparsity promoting term.

Another approach to finding structured controllers is to look for systems were the unconstrained optimal controller is structured in such a way that it is suitable for large-scale implementation. In [16] it is shown that for spatially invariant systems, the optimal controller will be spatially localized and decentralized. Other examples of distributed optimal controller includes [17] where it is shown that for systems with symmetric and Hurwitz state matrix, an optimal H_∞ controller can be found using a simple calculation involving the matrices of the systems state-space representation. Furthermore, if the plant has a sparsity pattern, the optimal controller will be distributed. This result is similar to ours in that it is the plant that gives the sparsity, without adding additional constraints to the controller. Another interesting structured controller is presented in [18], where the controller consists of a decentralized part, and a rank-one coordination term.

II. The Optimal Controller

In this section we will demonstrate that the solution to the optimal control problem in (4) does indeed have the structure as hinted at in (5) and (6). This will be presented as Theorem 1 and Theorem 2. We will also show how the internal flows for the more general problem (3) can be found by solving a static convex optimization problem in Theorem 3.

However, we start by introducing the required graph theoretic notions to define the upstream $U(i,j)$ and downstream $D(i,j)$ sets of a given edge (i,j) in a graph. We will begin with the definition for a rooted-tree. For that case the definitions simplify greatly, which will hopefully make the presentation easier to follow. We also believe that rooted-trees, corresponding to a single producer, will occur naturally in applications. After stating the theorems, we will discuss how the optimal controller can be implemented using local communication. Finally, we will give the necessary definitions to apply the theorems to arbitrary directed trees.

A. Graph Structuring

We begin with the definition of a directed rooted-tree.

Definition 1. *A directed graph is a directed rooted tree if it contains a single vertex such that there exists a unique directed path from this vertex to every other vertex in the graph. Equivalently a directed tree is rooted if it contains a single node with no parents.*

The upstream and downstream sets are easy to define for a rooted-tree.

Definition 2. *For a rooted-tree we define for every edge (i,j) the upstream and the downstream sets as follows:*

1) *The upstream set $U(i,j)$ is the source node j and all its decendents when the edge (i,j) is removed.*
2) *The downstream set $D(i,j)$ is the destination node i and all its decendents.*

The definition is illustrated in Fig. 2. When it is clear from context we sometimes drop the arguments (i,j) from the upstream and downstream sets.

In addition to the upstream and downstream sets, we also need to define the depth of every node to calculate the constants γ_U and γ_D.

Definition 3. *For a rooted-tree we define the depth $d(i)$ of a node i as follows*

- *The root has depth* 0.
- *For any other node i with parent j it holds that $d(i) = d(j) + 1$.*

We assume that only the root can be a producer.

Assumption 3. *All nodes in the producer set I has depth zero, i.e*

$$i \in I \Rightarrow d(i) = 0.$$

The appropriate definition of the depth of a node for the non-rooted case will be given in Definition 4. This will also generalize Assumption 3 for non-rooted trees.

B. Theorem Statements

With the necessary definitions in place for the rooted trees, we can now present our results. Note that these results hold for general directed trees by replacing Definition 2 with Definition 8 (see subsection II-D). We will first consider the case that the cost functions are quadratic, corresponding to the problem in (4). As already alluded to in the introduction, the optimal controller is highly structured in that it only needs the aggregate levels in the upstream and downstream sets to be implemented. This is demonstrated through Theorems 1 and 2 We then show in Theorem 3 how the optimal internal flows can be found for general convex cost functions. Here the controller retains the structure, in that it needs information from the same nodes as in the quadratic case. However, finding the optimal flow on a link will require solving a static convex optimization problem. The proofs will be given in Section III.

To simplify the description of the aggregate levels we make the following definition for a set of nodes S. The variable $m_S[t]$ describes the total amount of the quantity in the set S, including the quantity currently in transit towards a node in the set.

$$m_S[t] = \sum_{i \in S} \left(z_i[t] + \sum_{j \in \mathcal{P}(i)} u_{ij}[t-1] \right) \tag{7}$$

Now we can give a formal statement of the optimal internal flows.

Theorem 1. *Consider the problem in (4) under Assumption 2-3. For every edge $(i,j) \in \mathcal{E}$ with upstream set $U(i,j)$ and downstream set $D(i,j)$ as defined in Definition 2, let*

$$\gamma_{U(i,j)} = \left(\sum_{k \in U} \frac{1}{\alpha^{2(d(k)-d(j))} q_k} \right)^{-1}$$
$$\gamma_{D(i,j)} = \left(\sum_{k \in D} \frac{1}{\alpha^{2(d(k)-d(j))} q_k} \right)^{-1},$$

where $d(k)$ is the depth of node k as defined in Definition 3. Then the optimal value of $u_{ij}[t]$ is given by

$$u_{ij}[t] = \frac{\gamma_{U(i,j)}}{\gamma_{U(i,j)} + \gamma_{D(i,j)}} \alpha m_{U(i,j)}[t] - \frac{\gamma_{D(i,j)}}{\gamma_{U(i,j)} + \gamma_{D(i,j)}} \alpha m_{D(i,j)}[t]. \tag{8}$$

where $m_{U(i,j)}[t]$ and $m_{D(i,j)}[t]$ are the aggregate quantities of the upstream and downstream sets as defined in (7).

Note that the optimal flow on link (i,j) only depends on the aggregate levels in the upstream and downstream set. This gives a sparse and highly structured controller. Similarly, the optimal gains also only depends on the local cost function in the upstream and downstream set, allowing for efficient synthesis.

In Fig. 3 the structure of the feedback law is illustrated for the graph in Fig. 2. Also note that (8) can be implemented as a state feedback law, by letting $\{z_i[t], u_{ij}[t-1]\}$ be the state of the system.

Remark 1. *The result is of course compatible with the results presented for a string graph in [19]. Then the upstream set is just the source node, and the the downstream set is all the descendants of the source node, as follows from Definition 2.*

We will show in the next subsection how the aggregate levels and the γ parameters can be calculated recursively through the graph, allowing for an efficient implementation.

The optimal production is also structured and only needs to know the aggregate level for the entire graph.

Theorem 2. *Consider the problem in (4) under Assumption 2-3. Let*

$$R = \left(\sum_{i \in I} \frac{1}{r_i} \right)^{-1},$$

and

$$\gamma_{\mathcal{V}} = \left(\sum_{i \in \mathcal{V}} \frac{1}{\alpha^{2d(i)} q_i} \right)^{-1},$$

where $d(i)$ is the depth of node i as defined in Definition 3. Furthermore, let

$$X = -\frac{1}{2}\left[(1-\alpha^2)R - \alpha^2\gamma_{\mathcal{V}}\right] + \sqrt{\alpha^2\gamma_{\mathcal{V}}R + \frac{1}{4}\left[(1-\alpha^2)R - \alpha^2\gamma_{\mathcal{V}}\right]^2}.$$

Then the optimal total production $U[t] = \sum_{i\in I} u_{i0}[t]$ is given by

$$U[t] = -\frac{X}{X+R}\alpha m_{\mathcal{V}}[t],$$

where $m_{\mathcal{V}}$ is defined as in (7). The optimal production for the individual producers $i \in I$ is given by

$$u_{i0}[t] = \frac{R}{r_i}U[t].$$

The constant X is the solution to a scalar Riccati equation defined by the aggregate costs $\gamma_{\mathcal{V}}$ and R.

Remark 2. *Note that for rooted trees there can only be one node with production, and thus $U[t] = u_{i0}$ where i is the unique producing node. We state the theorem in this general way so that it holds for non-rooted trees as well.*

We will now state the results for general convex cost functions. The main difference is that more computations are required to find the optimal flows.

Theorem 3. *Consider the problem in (3) under Assumption 1-3. Then for every edge $(i,j) \in \mathcal{E}$ with upstream set $U(i,j)$ and downstream set $D(i,j)$ as defined in Definition 2, the optimal value of $u_{ij}[t]$ is given by the optimal u for the optimization problem*

$$\begin{aligned} \underset{u,x_k}{\text{minimize}} \quad & \sum_{k\in U(i,j)} f_k(x_k) + \sum_{k\in D(i,j)} f_k(x_k) \\ \text{subject to} \quad & \sum_{k\in U(i,j)} \alpha^{d(j)-d(k)} x_k = \alpha m_{U(i,j)}[t] - u \\ & \sum_{k\in D(i,j)} \alpha^{d(j)-d(k)} x_k = \alpha m_{D(i,j)}[t] + u. \end{aligned}$$

In the above $m_{U(i,j)}[t]$ and $m_{D(i,j)}[t]$ are defined by (7) and $d(k)$ is the depth of node k as defined in Definition 3.

Again the optimal flow only requires the aggregate levels in the upstream and downstream sets. However, each link must now solve a convex optimization problem with the aggregate levels of the upstream set and downstream set as input. These optimization problems gain more and more decision variables as we move up the graph. However, these type of problems are well studied and there exists plenty of software that can solve such problem even for a quite large number of decision variables, for example CVX [20].

We also give the general case of Theorem 2 in Proposition 1 in Section III. However, to our knowledge, the resulting problem would be very hard to solve even for small graphs, and we state it as a mechanism for proving Theorem 2.

C. Controller Implementation

Theorems 1 and 3 limit the information each node needs to the upstream set and the downstream set of each of its outgoing links. We will now show that by exploiting that the aggregate levels can be calculated by iterating through the graph, the controller can be implemented using only local communication. We only cover rooted trees in this subsection. Non rooted trees can be handled similarly, as covered in the next subsection.

For all nodes with no children nodes we define $m_i[t] = z_i[t] + u_{ij}[t-1]$, where j is the unique parent for node i. Then for all other nodes we define the node aggregate level $m_i[t]$ to be

$$m_i[t] = z_i[t] + u_{ij}[t-1] + \sum_{k\in\mathcal{C}(i)} m_k[t].$$

Where j again is the unique parent of node i, except for the root, where $u_{ij} = u_{i0}$. Note that the calculation of $m_i[t]$ only requires information from the neighboring nodes.

For link (i,j) the upstream and downstream levels can then by computed from the node aggregate levels,

$$m_{U(i,j)}[t] = m_j[t] - m_i[t], \quad m_{D(i,j)}[t] = m_i[t]. \tag{9}$$

This allows for a localized scheme for calculating the quantities needed to calculate the optimal flows. This is illustrated in Fig. 4, where we consider how the upstream and downstream level for the input considered in Fig. 2 can be calculated.

The aggregate costs γ_U and γ_D can be calculated in a similar way. Let

$$\gamma_i = \left(\frac{1}{q_i} + \sum_{j\in\mathcal{C}(i)} \frac{1}{\alpha^2\gamma_j}\right)^{-1}.$$

Then γ_U and γ_D are given by

$$\gamma_{U(i,j)} = \left(\frac{1}{\gamma_j} - \frac{1}{\alpha^2\gamma_i}\right)^{-1}, \quad \gamma_{D(i,j)} = \alpha^2\gamma_i.$$

Where again each node only needs to communicate with its neighbors.

The formulas proposed here gives an efficient way to implement the controller that greatly reduces the communication requirements for each node. This allows for a scalable implementation as the number of communication channels for each node remains low, even as the graph grows large. As the communication is in series, the time required for communication will grow as the depth of the graph increases. However, the depth is typically sub-linear in the number of nodes.

The distributed iteration of the γ parameter also allows for the controller to be efficiently updated when the graph changes. If a node i is added to or removed from the network, then only those nodes that has i as an ancestor needs to update their feedback law. How many nodes this is will depend on depth of node i and on the graph topology.

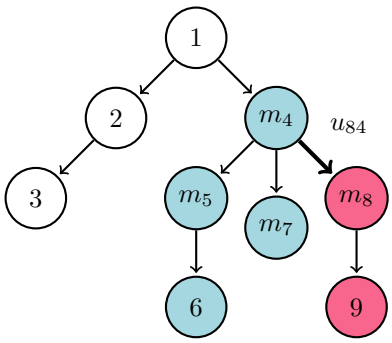


Fig. 4: The example in Fig. 2 revisited. The upstream and downstream sets for $(8, 4)$ are highlighted in blue and pink. However, by utilizing the aggregation of inventory level in (9) node 4 can find all the information it needs from its neighbors. Node fours aggregate level is given by $m_4 = m_5 + m_7 + m_8$. Then $m_{U(8,4)} = m_4 - m_8$ and $m_{D(8,4)} = m_8$. Thus no direct communication with node 6 and 9 is needed for node 4.

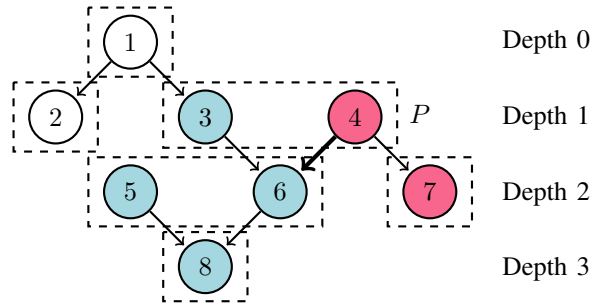


Fig. 5: Illustration of the graph structuring. Each row of nodes in the picture of the graph corresponds to nodes that have the same depth. All blocks in the graph has been marked by dashed squares. For the block P containing node three and four the set $\mathcal{F}(P)$ are all the colored nodes. For the link $(6, 4)$ the upstream set has been marked in pink and the downstream set in blue.

D. Non-Rooted Trees

We will now give the necessary definitions for defining the upstream and the downstream sets for non-rooted trees. We will also show how the implementation in this case can still be implemented using local communication, as in the previous subsection.

As there is no unique root in non rooted trees we have to generalize the definition of the depth of a node.

Definition 4. *For a non-rooted tree we define the depth $d(i)$ of a node i as follows: Pick any node in the graph and define the depth for that node to be d_0. Then define the depth of all other nodes by the following rules.*

- *For a node i with parent j it holds that $d(i) = d(j) + 1$.*
- *For a node i with child k it holds that $d(i) = d(k) - 1$.*

d_0 is then chosen so that $min_{i \in \mathcal{V}} d(i) = 0$.

Note that now multiple nodes can satisfy $d(i) = 0$ and thus there can be multiple producers while Assumption 3 is satisfied.

We define the existence of an undirected path in the following way.

Definition 5. *There exists an undirected path between node n_1 and node n_k if there exists a sequence of nodes $(n_1, n_2, \dots n_k)$ such that $(n_i, n_{i+1}) \in \mathcal{E}$ or $(n_{i+1}, n_i) \in \mathcal{E}$ for all $1 \leq i \leq k - 1$.*

To describe the upstream and downstream sets we split the nodes on the same depth into blocks. The definition is illustrated in Fig 5.

Definition 6. *Two nodes i and j are in the same block if they satisfy the following:*

1) *Node i and Node j has the same depth, i.e. $d(i) = d(j)$.*
2) *There exists a undirected path between node i and node j when the incoming links to node i and node j are removed.*

Let the block of node i be denoted $B(i)$. For rooted trees each block contains only one node. Each block P can be associated with a set of nodes $\mathcal{F}(P)$, which we call its family, in the following way.

Definition 7. *For a block P, $i \in \mathcal{F}(P)$ if*

- *$i \in P$*

or if

- *There exists an undirected path from i to a node in P when all incoming links to the block P, that is $\forall (k, l),\ k \in P$, are removed.*

For a node $i \in P$ we also let $\mathcal{F}(i) = \mathcal{F}(P)$. This definition is illustrated in Fig. 5.

We are now ready to define the upstream and downstream sets for general directed trees. The definition is illustrated in Fig. 5 and Fig. 6.

Definition 8. *For a non-rooted directed tree we define the upstream and the downstream set as follows*

A node k is in the upstream set $U(i, j)$ if the following holds

1) *$k \in \mathcal{F}(j)$*
2) *$k = j$ or there exists an undirected path between k and the source j when (i, j) is removed from $\mathcal{G}$.*

A node k is in the downstream set $D(i, j)$ if the following holds

1) *$k \in \mathcal{F}(j)$*
2) *$k = i$ or there exists an undirected path from node k to the destination i when (i, j) is removed from $\mathcal{G}$.*

Remark 3. *Definition 8 is a generalization of Definition 2. The proofs of Theorem 1 and Theorem 3 use this generalized definition and thus those theorems holds also for non rooted trees.*

We will now consider how the optimal controller for non rooted trees can be implemented. To do this we introduce the following notation.

Definition 9. *Consider an edge (i, j) where $j \in P$ and with upstream set $U(i, j)$ and downstream set $D(i, j)$. We let the nodes in $U(i, j)$ that are in P be denoted $U_P(i, j)$ and the set of children blocks of P who's nodes are in $U(i, j)$ as $U_C(i, j)$.*

Similarly define $D_P(i,j)$ as the set of nodes in P that are in $D(i,j)$ and the set of children blocks of P who's nodes are in $D(i,j)$ as $D_C(i,j)$.

The definition is illustrated in Fig. 6. Note that the nodes in each children block is either all in $U(i,j)$ or all in $D(i,j)$. Also note that both $U_C(i,j)$ and $D_P(i,j)$ might be empty. For a rooted tree $U_p(i,j)$ is just the source node j and $C_p(i,j)$ is the empty set.

Starting with the blocks B without children, which thus contains only one node $i \in B$, we define

$$m_B[t] = z_i[t] + \sum_{j \in \mathcal{P}(i)} u_{ij}[t-1].$$

Note that for non rooted trees each node might have multiple parents. By iterating through the graph, we define for block B with children blocks $\mathcal{C}(B)$

$$m_B[t] = \sum_{i \in B} \left(z_i[t] + \sum_{j \in \mathcal{P}(i)} u_{ij}[t-1] \right) + \sum_{c_i \in \mathcal{C}(B)} m_{c_i}[t]. \tag{10}$$

We can now, for every edge (i,j), calculate the aggregate upstream and downstream level as

$$m_{D(i,j)}[t] = \sum_{k \in D_P(i,j)} \left(z_k[t] + \sum_{l \in \mathcal{P}(i)} u_{kl}[t-1] \right) + \sum_{c_k \in D_C(i,j)} m_{c_k}[t]$$
$$m_{U(i,j)}[t] = m_{B(j)}[t] - m_{D(i,j)}[t].$$

We can also similarly define the aggregate cost for blocks B with no children nodes as $\gamma_B = q_i$ and for all other blocks as

$$\gamma_B = q_i$$
$$\gamma_B = \left(\sum_{i \in b} \frac{1}{q_i} + \sum_{c_i \in \mathcal{C}(b)} \frac{1}{\alpha^2 \gamma_{c_i}} \right)^{-1}.$$

This allows for the upstream and downstream aggregate cost to be calculated in the following way

$$\gamma_{D(i,j)} = \left(\sum_{k \in D_P(i,j)} \frac{1}{q_k} + \sum_{c_k \in D_C(i,j)} \frac{1}{\alpha^2 \gamma_{c_k}} \right)^{-1}$$
$$\gamma_{U(i,j)} = \left(\frac{1}{\gamma_{B(j)}} - \frac{1}{\gamma_{D(i,j)}} \right)^{-1}.$$

In the rooted tree case, each node only needed to communicate with its direct neighbors. Now each node need to communicate with all the nodes in its block, and all its children blocks. This is illustrated in Fig. 6, where all the quantities needed to calculate the flow on the highlighted link are shown. This will in general require more communication channels, and also some abstraction in handling the block to node communication. Still, the communication requirement will for most graphs grow sub-linearly in the number of nodes in the graph.

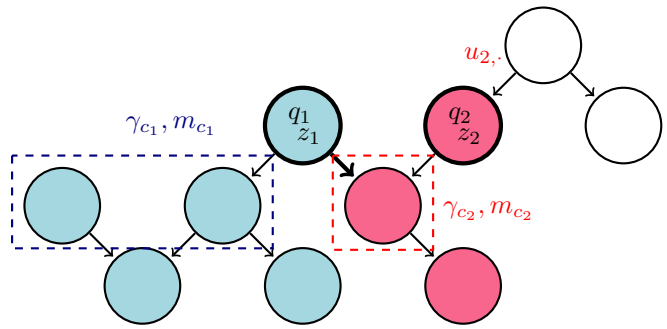


Fig. 6: Illustration of the calculation for the flow on the highlighted link. The upstream set is highlighted in blue and the downstream set in pink. Furthermore U_P and D_P are highlighted as the two nodes with a thick border. The two children blocks have also been highlighted as dashed squares. Finally all the quantities needed to calculate the flow has been written out. That is the children block aggregate levels m_{c_i} and aggregate cost γ_{c_i}. And for the nodes in the parent block, node level z_i, node inflow $u_{2,\cdot}$, and node costs q_i.

III. Derivation of Optimal Controller

The goal of this section is to prove the results presented in the previous section. We will begin by discussing the proof idea, before giving the proof. Throughout this section we will use the layer of a node instead of its depth, as that is notationally more convenient. We define the layer of a node by first finding the maximum layer,

$$l_{\max} = \max_{i \in \mathcal{V}} d(i).$$

The layer of a node is then defined as $l(i) = l_{\max} - d(i)$. As can be seen from the definition, the layers starts from the bottom of the graph, while the depth starts from the top. We also let $l(B)$ denote the layer of the nodes in block B, and T as the block in the maximum layer. Then it holds that $l(T) = l_{\max}$. There can only be one such block as the graph is assumed to be connected.

A. Proof Idea

The proof relies on decomposing the problem into independent subproblems. To do so a shifted level vector, defined for a block P as

$$\Big\{ z_i[t - l(i)], \quad i \in \mathcal{F}(P) \Big\},$$

will be studied. See Fig. 7 for a graphical illustration. In Fig. 7 it can be noted that the highlighted flow $u_{34}[t-3]$ will only directly affect the levels $z_3[t-1]$ and $z_4[t-2]$ which are both in the same shifted level. More generally every flow, $u_{ij}[t]$ will only directly affect the nodes within one shifted level vector.

We define the cost associated with a shifted level for a block P

$$F_P(z,t) := \sum_{i \in \mathcal{F}(P)} f_i(z_i[t - l(i)]).$$

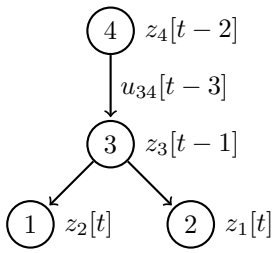


Fig. 7: Illustration of a shifted level vector. The shift of each node is proportional to the layer of the node. The highlighted link flow $u_{34}[t-3]$ only directly affect the source $z_4[t-2]$ and the destination $z_3[t-1]$, which are both in the same shifted level.

This allows for the first term in the cost function in (4) to be written in terms of the shifted levels in the following way,

$$\sum_{t=0}^{\infty}\sum_{i\in\mathcal{V}} f_i(z_i[t]) = \sum_{i\in\mathcal{V}} f_i(z_i[0]) + \sum_{t=1}^{l_{\max}} \sum_{P:l(P)=t-1} F_P(z,t) + \sum_{t=l_{\max}+1}^{\infty} F_T(z,t). \quad (11)$$

This formula is illustrated for a three node string graph in Fig. 8. Recall that T is the block consisting of the top layer, and $F_T(z,t)$ is then the cost for a shifted level for the entire graph.

In Fig. 8 we see that all but one of the levels within a shifted level can be chosen freely by choosing the outflow from each node appropriately. The last level will then be decided by the total level of the previous shifted level. Thus the possible shifted levels are only constrained by the sum of the previous shifted level. Furthermore, from the figure it is clear that sum of the shifted levels are independent of the internal flows u_{ij}. This implies that each shifted level can be optimized independently, as each flow decision only affects one shifted level. We will show that this holds for general directed trees by introducing a weighted sum of the shifted levels

$$S_P[t] = \sum_{i\in\mathcal{F}(P)} \alpha^{l(i)-l(P)} z_i[t-l(i)],$$

and showing that it satisfies a simple conservation law. The weighting reflects the fact that if we move some quantity c from a node to one of its children, then only αc will arrive.

B. Decomposition

The time shift in the shifted level is not suitable for controller implementation. However there is a relationship between the shifted level and aggregate level.

Definition 10. *Define the aggregate level $m_P[t]$ for block P as*

$$m_P[t] = \sum_{i\in\mathcal{F}(P)} \left(z_i[t] + \sum_{j\in\mathcal{P}(i)} u_{ij}[t-1] \right).$$

This definition is different from (7) as it is defined on a block, and not on a set of nodes. However, it holds that $m_P[t] = m_{\mathcal{F}(P)}[t]$.

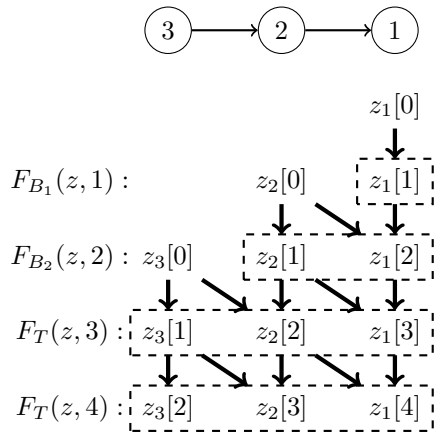


Fig. 8: Illustration of the dynamics for the three node graph at the top of the picture. The graph has three blocks. The block B_1 contains node one, the block B_2 contains node two, and the top block T contains node three. Each node has been shifted proportional to its layer, making each horizontal slice a shifted level. The arrows indicates were the quantities in a node can go, i.e either stay in the node or be transported, with a delay, to a child. Each term included in (11) has been highlighted by dashed rectangles. It can be seen that the total level in every row is independent of the internal flows. It can also be seen that $z_3[t+1]$ and $z_2[t+1]$ can be chosen freely by picking $u_2[t] = z_3[t] - z_3[t+1]$ and $u_1[t] = z_2[t] + u_2[t-1] - z_2[t+1]$.

We will now show some important properties of the shifted level vectors and its relationship to the weighted sum. These properties are related to the dynamics, and are independent of the optimization problems.

Lemma 1. *For any block P, define for $t \geq l(P)+1$*

$$S_P[t] = \sum_{i\in\mathcal{F}(P)} \alpha^{l(i)-l(P)} z_i[t-l(i)].$$

Assume that $z[t]$ satisfies the dynamics in (2)*. Then*

(a) $S_P[t]$ satisfies the conservation law

$$S_P[t] = \alpha\left(S_P[t-1] + \sum_{\substack{i\in P\\ j\in\mathcal{P}(i)}} u_{ij}[t-l(i)-2] \right). \quad (12)$$

(b) Given a shifted level for P and the inflow to P,

$$\begin{aligned} z_i[t-l(i)-1], &\quad i\in\mathcal{F}(P) \\ u_{ij}[t-l(i)-2], &\quad i\in P, \end{aligned} \quad (13)$$

then the only constraint on the next shifted level

$$z_i[t-l(i)], \quad i\in\mathcal{F}(P) \quad (14)$$

is that (12) *is satisfied. In addition the flows*

$$u_{ij}[t-l(j)-1],\ j\in\mathcal{F}(P)$$

are unique for every pair of values for (13) *and* (14)*.*

(c) Neither $S_P[\tau]$ for $\tau = l(P)+1$, nor $S_T[\tau]$ for $\tau > l_{max}$, depend of the internal flows $u_{ij}[t]$, for $t \geq 0$.

(d) It holds that

$$\alpha m_P[t-l(P)-1] = S_P[t].$$

Proof. Part a), c) and d) are proved in the appendix. We prove b) here. Recall that the dynamics are given by

$$z_i[t-l(i)] = \alpha\Big(z_i[t-l(i)-1]+\sum_{h\in\mathcal{P}(i)} u_{ih}[t-l(i)-2]\Big)-\sum_{j\in\mathcal{C}(i)} u_{ji}[t-l(i)-1]. \tag{15}$$

A node level $z_i[t-l(i)], i \in P$ is determined when all of the node's inflows and outflows directly affecting the level, and the previous level $z_i[t-l(i)-1]$, are determined . By assumption, the inflows to nodes in P,

$$u_{ij}[t-l(j)-2],\ j \in P,$$

are given as well as the previous level

$$z_i[t-l(i)-1],\ i \in \mathcal{F}(P).$$

Thus all that remains is the flows going between any pair of nodes in $\mathcal{F}(P)$. Consider the graph containing the nodes that have not yet fixed their level $z_i[t-l(i)]$, that is $i \in \mathcal{F}(P)$. And the edges that have not yet fixed their flow $u_{ij}[t-l(j)-1]$, that is (i,j) $j \in \mathcal{F}(P)$. This graph is a directed tree and thus contains a node that is incident to only one edge (as otherwise the graph would contain a cycle). The flow on this edge is the only term that is unspecified in (15) for the node and must be chosen according to the unique value so that the target node level in (14) is achieved.

Now the graph of edges with undecided flows and nodes with non fixed levels contain one edge and one node less. However, it is still a directed tree as all the remaining nodes stay connected and no cycles have been introduced. We can once again find a node with only a single edge with unspecified flow. The flow on this edge must again be uniquely chosen so that the wanted node level in (14) is achieved.

This can be continued until the graph of edges with undecided flows and nodes with non fixed levels only contains two nodes and one edge. Then the flow on the last edge can be chosen so that at least one of the nodes get the correct value. By (a), both nodes will have the correct level if and only if (12) is satisfied. This flow will be unique, as there is only one flow for each node that gives the correct level. □

Before we continue we will show that the problems in (3) and (4) have a unique solution. First we show that the problems have a bounded minimizer. This is easily done by constructing a set of inputs that takes all the node levels to zero in finite time with finite control effort, as $f_i(0) = g_i(0) = 0$. Let the production at time $t = 0$ be such that,

$$\sum_{i\in I} u_{i0}[0] = -\sum_{i\in\mathcal{V}}\Big(z_i[0] + \sum_{j\in\mathcal{P}(u)} u_{ij}[-1]\Big)$$

and $u_{i0}[t] = 0,\ t \geq 1$. Then $m_{\mathcal{V}}[t] = 0,\ \forall t \geq 1$. Now it only remains to find internal flows so that all levels are zero in finite time. That this is possible follows from (b) and (d) in Lemma 1. The problems in (3) and (4) both have a strictly convex cost function, with a cost bounded from above by the previous argument, and a cost bounded from below by zero. Thus the optimal value for $z_i[t]$ and $u_{i0}[t]$ are unique. The internal flows are not part of the cost function. However, it follows from (b) in Lemma 1, that they will also be unique.

We now use Lemma 1 to decompose the problem by showing that each shifted level, corresponding to a row in Fig. 8, can be optimized independently.

Lemma 2. *Assume that z is the minimizer for (4). Then for $1 \leq t \leq l_{\max}$ and all blocks P s.t. $l(P) = t$, z is also the minimizer for*

$$\begin{array}{ll} \underset{z_i[t-(i)],i\in\mathcal{F}(P)}{\text{minimize}} & F_P(z,t) \\ \text{subject to} & S_P[t] = \alpha m_P[0] \end{array}$$

And for $t > l_{\max}$, z is the minimizer for

$$\begin{array}{ll} \underset{z_i[t-l(i)],i\in\mathcal{V}}{\text{minimize}} & F_T(z,t) \\ \text{subject to} & S_T[t] = \alpha m_T[t-l_{\max}-1] \end{array}$$

Proof. Recall that

$$F_P(z,t) = \sum_{i\in\mathcal{F}(P)} f_i(z_i[t-l(i)])$$

and thus corresponds to a shifted level. By (b) in Lemma 1, the only constraint on a shifted level is that is satisfies (12). By (c) in Lemma 1 the shifted level corresponding to the constraint for each term in (11) is unaffected by the internal flows. Thus each term in (11) can be optimized independently, without affecting the optimal value of the other terms. Applying Lemma 1d to (12) completes the proof. □

C. Proof of the Theorems

Lemma 2 can be used to find the optimal levels in a centralized way, relying on the method in the proof of lemma 1b to then find the optimal flows. However, to achieve the distributed implementation of the controller described in the previous section, one needs a more direct approach to finding the optimal link flow. To do this we need to understand how the flow $u_{ij}[t]$ affects the upstream and downstream sets of the corresponding link.

Lemma 3. *Assume that $z[t]$ satisfies the dynamics in* (2)*. Then for every edge (i,j) where the source node j is in block P, and with upstream set $U(i,j)$ and downstream set $D(i,j)$, it holds that:*

$$\begin{aligned} \sum_{k\in U(i,j)} \alpha^{l(k)-l(P)} z_k[t-l(k)] &= \\ &\alpha m_{U(i,j)}[t-l(P)-1] - u_{ij}[t-l(j)-1] \\ \sum_{k\in D(i,j)} \alpha^{l(k)-l(P)} z_k[t-l(k)] &= \\ &\alpha m_{D(i,j)}[t-l(P)-1] + u_{ij}[t-l(j)-1] \end{aligned} \tag{16}$$

For the proof, see the appendix. We are now ready to prove Theorem 3.

Proof of Theorem 3. We will first show that a necessary condition for z_i to be a minimizer for (3) is to also be a minimizer for

$$\begin{aligned} \underset{z_k[t-(l(k)-1)],k\in\mathcal{F}(B)}{\text{minimize}} \quad & \sum_{k\in\mathcal{F}(B)} f_k(z_k[t-l(k)]) \\ \text{subject to} \quad & S_B[t] = \alpha m_B[t-l(B)-1], \end{aligned} \tag{17}$$

for all blocks B and all $t \geq l(B)+1$.

For any block B and time $t \geq l(B)+1$ we pick a block P in the following way: For $t \leq l_{\max}$ there exists a block P s.t. $\mathcal{F}(B) \in \mathcal{F}(P)$ and $t = l(P)+1$. For $t > l_{\max}$ take $P = T$, then $\mathcal{F}(B) \in \mathcal{F}(P)$. By Lemma 2 it then holds for that block P that the optimal z must be a minimizer for

$$\begin{aligned} \underset{z_k[t-l(k)],k\in\mathcal{F}(P)}{\text{minimize}} \quad & \sum_{k\in\mathcal{F}(P)} f_k(z_k[t-l(k)]) \\ \text{subject to} \quad & S_P[t] = \alpha m_P[t-l(P)-1]. \end{aligned} \tag{18}$$

We have from the definition of $S_P[t]$ that

$$S_P[t] = \sum_{k\in\mathcal{F}(P)\setminus\mathcal{F}(B)} \alpha^{l(k)-l(P)} z_k[t-l(k)] + \alpha^{l(B)-l(P)} S_B[t],$$

and from Lemma 1d that $S_B[t] = \alpha m_B[t-l(B)-1]$. Thus z_i must be the minimizer of (17) to be the minimizer of (18). If not, the value of the objective function in (18) could be improved by using the minimizer for (17) for all $i \in \mathcal{F}(B)$ and using the same z_i for $i \in \mathcal{F}(P) \setminus \mathcal{F}(B)$. And it follows that for every block B that the optimal $z_i[t]$ for (3) must satisfy (17).

Adding the two equalities in (16) gives the constraint in (17). Thus if (16) holds, then the constraint in (17) must also hold. Also if the constraint in (17) holds, then so must (16) for any $u_{ij}[t-l(j)-1]$. This can be seen by adding and subtracting u_{ij} to (17) and then splitting the equality into two. Thus any outgoing link from B, that is $(i,j),\ j \in B$, z is a minimizer for (17) if and only if z and u are a minimizer for for

$$\begin{aligned} &\underset{u,z_k}{\text{minimize}} \\ &\quad \sum_{k\in U} f_k(z_k[t-l(i)]) + \sum_{k\in D} f_k(z_k[t-l(k)]) \\ &\text{subject to} \\ &\quad \sum_{k\in U} \alpha^{l(k)-l(B)} z_k[t-l(k)] = \alpha m_U[t-l(B)-1] - u \\ &\quad \sum_{k\in D} \alpha^{l(k)-l(B)} z_k[t-l(k)] = \alpha m_D[t-l(B)-1] + u. \end{aligned}$$

Let $x_i = z_i[t-l(i)]$, shift the time by $l(B)-1$ and use that $l(k)-l(B) = l(k)-l(j) = d(j)-d(k)$ and the theorem statement is achieved. Which then must be a necessary condition for optimality.

We previously showed that the optimal flows exists and are unique. Thus the necessary condition is also sufficient. □

We can use Lemma 2, i.e. the fact that the shifted level vectors are optimally distributed to simplify the problem of finding the optimal production.

Proposition 1. *Let $Q(m_T[t-l_{max}-1])$ denote the optimal value for the optimization problem*

$$\begin{aligned} \underset{z_i[t-l(i)],i\in\mathcal{V}}{\text{minimize}} \quad & \sum_{i\in\mathcal{V}} f_i(z_i[t-l(i)]) \\ \text{subject to} \quad & \sum_{i\in\mathcal{V}} \alpha^{l(i)-l_{max}} z_i[t-l(i)] = \alpha m_T[t-l_{max}-1] \end{aligned}$$

for $t \geq l_{max}+1$. Then the optimal production $u_{i0}[t]$ for problem (3) is given by the minimizer for

$$\begin{aligned} \underset{u_{i0}}{\text{minimize}} \quad & \sum_{t=0}^{\infty} \left[Q(m_T[t]) + \sum_{i\in I} g_i(u_{i0}[t]) \right] \\ \text{subject to} \quad & m_T[t+1] = \alpha m_T[t] + \sum_{i\in I} u_{i0}[t]. \end{aligned}$$

Proof. Recall that the cost can be written as (11). From (d) in Lemma 1 we have that for $P : l(p) = t-1$

$$S_P[t] = \alpha m_P[0],$$

and thus the production only affect the terms

$$\sum_{t=l_{\max}+1}^{\infty} F_T(z,t),$$

in (11). If the internal flows are chosen optimally, then by Lemma 2, the total cost that the inflows affect are then given by

$$\sum_{t=0}^{\infty} \left[Q(m_T[t]) + \sum_{i\in I} g_i(u_{i0}[t]) \right].$$

The fact that

$$m_T[t+1] = \alpha m_T[t] + \sum_{i\in I} u_{i0}[t]$$

follows from (a) and (d) in Lemma 1, since

$$\begin{aligned} &\alpha m_T[t-l_{\max}-1] = S_T[t] \\ &\quad = \alpha \left(S_T[t-1] + \sum_{i\in I} u_{i0}[t-l_{\max}-2] \right) \\ &\quad = \alpha \left(\alpha m_T[(t-1)-l_{\max}-1] + \sum_{i\in I} u_{i0}[t-l_{\max}-2] \right). \end{aligned}$$

□

We are almost ready to prove Theorem 1 and 2. However, first we need the following lemma, which can be used to solve the problem in Theorem 3 and find the function Q in Proposition 1 when the cost functions are quadratic.

Lemma 4. *Consider the problem*

$$\begin{aligned} \text{minimize} \quad & \sum_{i\in P} q_i m_i^2 \\ \text{subject to} \quad & \sum_{i\in P} \alpha^{l(i)-l(P)} m_i = m. \end{aligned}$$

Let

$$\gamma = \left(\sum_{i\in P} \frac{\alpha^{2(l(i)-l(P))}}{q_i} \right)^{-1}.$$

Then the optimal m_i is given by

$$m_i = \frac{\alpha^{l(i)-l(P)}}{q_i}\gamma m$$

and the optimal value of the objective function is given by

$$\gamma m^2.$$

For the proof, see the Appendix.

Proof of Theorem 1 and 2. Using Lemma 4 and that the cost functions are quadratic, we can rewrite the optimal u_{ij} from Theorem 3 as

$$\begin{aligned} u_{ij}[t] = \arg\min_u \quad & \gamma_{U(i,j)}(\alpha m_{U(i,j)}[t] - u)^2 \\ & + \gamma_{D(i,j)}(\alpha m_{D(i,j)}[t] + u)^2. \end{aligned}$$

Differentiating with respect to u gives the optimal link flow as in Theorem 1.

By Lemma 4 we have than for quadratic cost functions, the function $Q(m_T[t])$ in Proposition 1 is given by

$$Q(m_T[t]) = \gamma_T \alpha^2 m_T^2[t].$$

Let the total production at time t be denoted $U = \sum_{i\in I} u_{i0}[t]$. Then if the total production is split optimally along the producers, the cost is

$$\sum_{i\in I} g_i(u_{i0}[t]) = RU^2,$$

and each individual production is given by $u_i = RU/r_i$ (by application of Lemma 4 with $\alpha = 1$). This allows us to formulate the problem in Proposition 1 as a standard linear quadratic control problem:

$$\begin{aligned} \underset{U}{\text{minimize}} \quad & \sum_{t=0}^{\infty} \gamma_T \alpha^2 m_T[t]^2 + RU[t]^2 \\ \text{subject to} \quad & m_T[t+1] = \alpha m_T[t] + U[t] \end{aligned}$$

This problem can be solved using the Riccati equation, see for example [21]. Let

$$A = \alpha, \quad B = 1, \quad Q = \alpha^2 \gamma_T.$$

Then the solution to the scalar Riccati equation

$$A^T X A - (A^T X B)(B^T X B + R)^{-1}(A^T X B)^T + Q = X,$$

is as given in the theorem statement. The optimal total production then follows from

$$U[t] = -(B^T X B + R)^{-1} B^T X A m_T[t].$$

□

IV. Simulation Examples

In this section we will study the performance of the optimal controller in networks of different sizes and topologies. We aim to illustrate the closed loop behavior, and study how the control performance scales with the size of the graph. We restrict ourselves to quadratic cost functions and study two cases designed to illustrate dynamic performance in the face of changes in equilibrium point, and random disturbances. For both cases the simulations were run on two different types of graphs; a directed string graph (every node except the last has one child), and a binary tree graph (every node has two children, except for those at the maximum depth).

First we consider performance when the network is subject to non-zero initial conditions. This could for instance be of interest if the underlying equilibrium is changing (the initial conditions correspond to the difference between the old equilibrium point, and the new one). From the derivation of the optimal controller we saw that this case can be split into two parts. The optimal distribution of the available goods, and the optimal production. The optimal production essentially corresponds to solving a first order system, and we will focus on the optimal distribution here. Thus we will normalize the initial conditions to sum to zero. To ensure a wide range of initial conditions were considered, the initial node levels $z_i[0]$ and the initial transport levels $u_{ij}[-1]$ were drawn from a multivariable Gaussian distribution

$$\mathcal{N}\left(0, \frac{M}{M-1} \cdot \left[I - \frac{\mathbf{1}\mathbf{1}^T}{M}\right]\right),$$

where M is the total number of nodes and links. This choice guarantees that the initial values sum to zero, i.e.

$$\sum_{i\in\mathcal{V}} \left(z_i[0] + \sum_{j\in\mathcal{P}(i)} u_{ij}[-1] \right) = 0,$$

and keeps the diagonal elements of the covariance matrix the same for all graph sizes. For simplicity, the cost function for each node was set to $q_i = 1$, the decay factor to $\alpha = 0.99$. The simulations were run 1000 times for each graph size.

Secondly we consider the case of persistent Gaussian disturbances acting on the node levels $z_i[t]$. This case could be motivated for example by having an unknown demand on the nodes of the underlying system. We again set $q_i = 1$ for all nodes, and $\alpha = 0.99$. The state disturbances were set to be normally distributed with zero mean and a standard deviation of 0.01. We considered the optimal controller both with and without production. For the case of production, we set the production cost to be $r = 10$. 1000 simulations were run for each graph size, and each simulation was run for 100 time steps.

To provide some sort of comparison, a local controller was also designed with off-the-shelf computational tools and tested on the same simulation cases. By local, we mean that the local controller was forced to satisfy certain structural constraints restricting which variables were available for feedback. In particular, the transportation on each link was decided based only on the levels in the source node, the children of the source node, and the goods in transit towards those nodes. For simplicity, the control law was further restricted to be static, and the Matlab function `fmincon` was used to conduct the non-convex optimization corresponding to optimizing over such controllers gains. Of course there are no guarantees that this process will yield even a optimal controller subject to the constraints (indeed, for larger graphs not even a local minima was found). However, since the controllers designed using the tools from this paper are globally optimal (they give a structured solution to a standard unconstrained LQ problem),

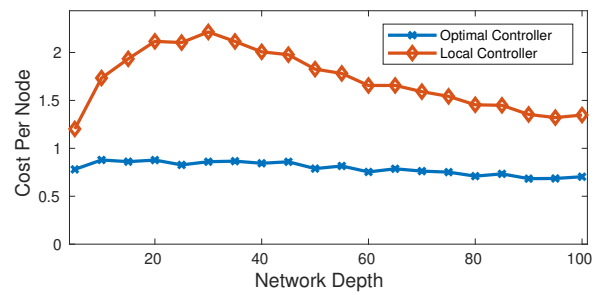
the performance of any local controller will necessarily be worse, and it's purpose is more to provide perspective than represent a 'good way' to design decentralized controllers.

The simulation results for the first case (random initial conditions) are presented in Figs. 9–10. The cost per node for different graph sizes can be seen in Fig. 9. We can see in (a) that the optimal controller get slightly better performance as the number of nodes increases. This is to be expected, since otherwise it would suggest that different parts of the string could be controlled independently. For the local controller the performance first decreases, and then increases. This might be due to two competing effects. On the one hand, it should be easier to control a longer string for the same reasons as for the optimal controller. However, as the graph gets shorter each local controller has access to a larger subset of the total information, which should increase performance, For the tree case in Fig. 9b we again see that the performance of the local controller decreases as the depth of the graph increases. We could expect similar behavior as for the string case, however it is not feasible to synthesize the local controller for deeper graphs due to the exponential increase in the number of nodes. We can also note that the string structure seems more efficient than the tree structure for this problem, in that it yields lower cost per node. The state trajectories for a few nodes in a string graph of length 20 has been plotted in Fig. 10. Note that the optimal controller brings the system to the optimal configuration at time 20, which can be compared to the depth of the network which is 19.

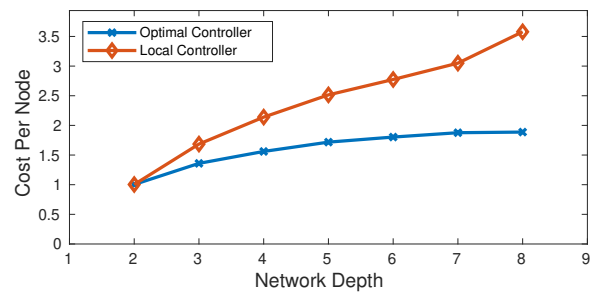
The simulation results for the second case (persistent Gaussian disturbances) for different graph-sizes can be seen in Figs. 11–12. Fig. 11 shows that the performance for the optimal controller without production and the local controller both improve as the graph grows. However this feature all but disappears when production is allowed. This is likely due to the fact that the larger the graph is, the more likely it is that the total level is zero, meaning the need for production declines as the graphs become larger. Secondly, while the optimal controller still outperforms the local one, the difference is much smaller now. We can also note that the difference in performance between the string case and the tree case is negligible now. State trajectories for the different control strategies can be seen in Fig. 12.

V. Conclusion

We have studied a class of optimal transportation problems over a network, where the transportation is subject to delay. It was shown that under simple modeling assumptions the optimal control policy has a sparse and structured solution, suitable for large scale systems. In the case of quadratic cost functions, closed form expressions for the optimal control were derived. The controller was compared to a local controller designed with off-the-shelf methods. The optimal controller showed a significant improvement in performance for the case of non-zero initial conditions, corresponding to the dynamical adjustment from one equilibrium point to another.



(a) Comparison of the performance of the local and the optimal controller for a string graph, with an initial disturbance.



(b) Comparison of the performance of the local and the optimal controller for a rooted-tree, with an initial disturbance. Every node except the ones as max depth has two children.

Fig. 9: Performance comparison for the local and the optimal controller for a string graph and a rooted-tree graph with non-zero initial conditions. In both cases the initial values has been chosen so that they sum to zero.

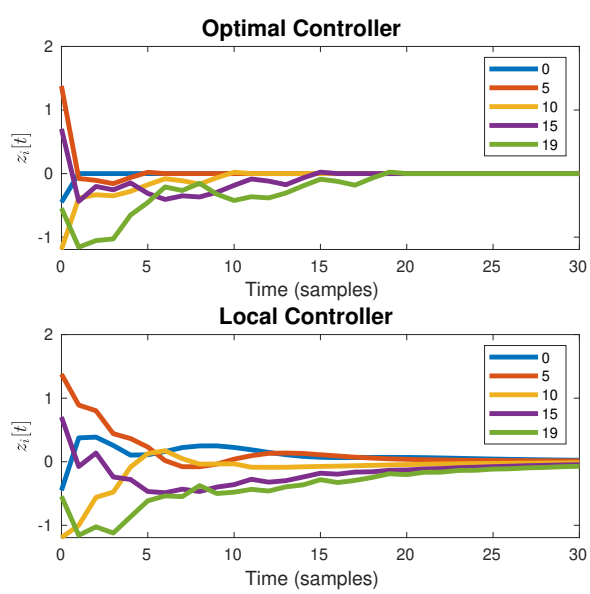
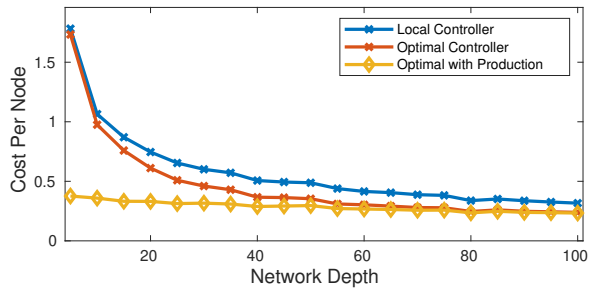
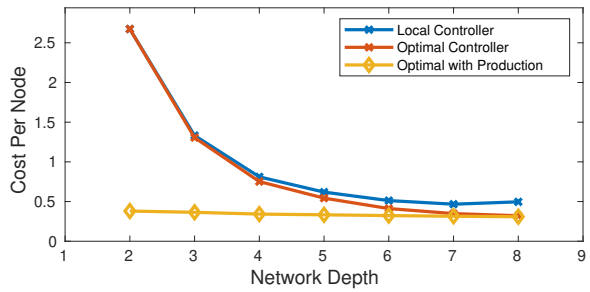


Fig. 10: State trajectories of the optimal and the local controller for a string graph with 20 nodes and non-zero initial conditions. The legend indicate the depth of the plotted node. We can see that the optimal controller reaches the optimal level ($z = 0$) at time 20, compared to the network depth of 19.



(a) Comparison of the performance of the local and the optimal controller for a string graph subject to persistent disturbance from Gaussian noise.



(b) Comparison of the performance of the local and the optimal controller for a rooted-tree subject to persistent disturbances from Gaussian noise. Every node except those at max depth has two children.

Fig. 11: Performance comparison for the local and the optimal controller for a string graph and a rooted-tree graph. In both cases the the system is subjected to Gaussian noise.

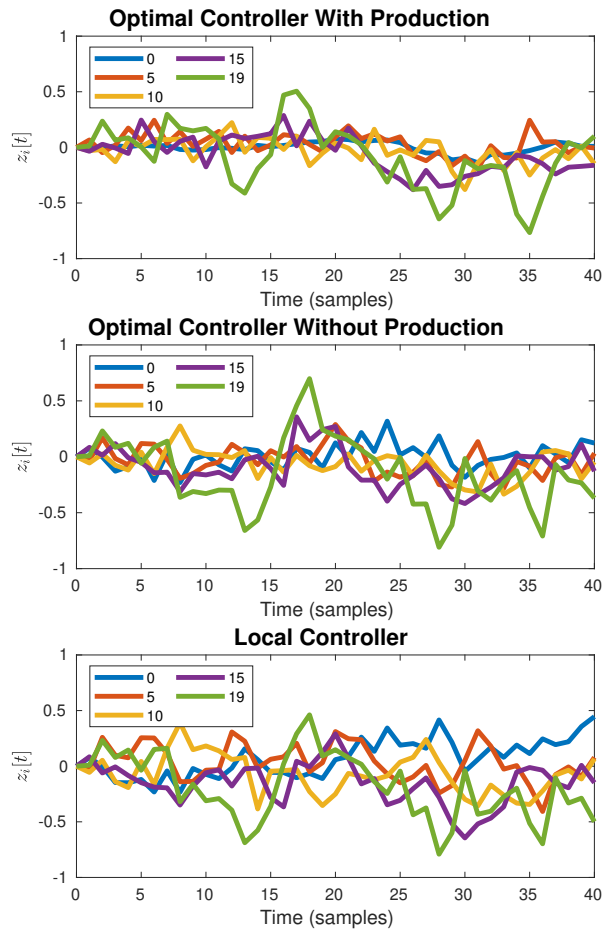


Fig. 12: Illustration if the optimal controller with and without production, and the local controller for a string graph with 20 nodes subject to Gaussian noise. The legend indicate the depth of each plotted node.

Acknowledgment

The authors gratefully acknowledge financial support from Swedish Foundation for Strategic Research through the project SSF RIT15-0091 SoPhy.

Appendix

Proof of Lemma 1. **(a)** For any block without children, which is just a node without children, the lemma follows form the dynamics. Now assume that for a block P the Lemma holds for all the children blocks $\{C \in \mathcal{C}(P)\}$. Then we have that

$$S_C[t] = \alpha\Big(S_C[t-1] + \sum_{\substack{i\in C\\ j\in\mathcal{P}(i)}} u_{ij}[t-l(i)-2]\Big). \tag{19}$$

We can rewrite the shifted level for P as

$$S_P[t] = \sum_{j\in P} z_j[t-l(j)] + \frac{1}{\alpha}\sum_{C\in\mathcal{C}(P)} S_C[t]. \tag{20}$$

Using the dynamics the sum over the nodes in the parent block can be rewritten as

$$\sum_{j\in P} z_j[t-l(j)] = \sum_{j\in P}\Bigg[\alpha\Big(z_j[t-l(j)-1] + \sum_{k\in\mathcal{P}(j)} u_{jk}[t-l(j)-2]\Big) - \sum_{i\in\mathcal{C}(j)} u_{ij}[t-l(j)-1]\Bigg]. \tag{21}$$

As $\mathcal{P}(i) = P$ for $i \in \mathcal{C}(P)$ and $l(i) = l(j) - 1$ for $j \in \mathcal{P}(i)$ it holds that

$$\sum_{\substack{i\in C\\ j\in\mathcal{P}(i)}} u_{ij}[t-l(i)-2]\Big) = \sum_{j\in P}\sum_{i\in\mathcal{C}(j)} u_{ij}[t-l(j)-1].$$

Inserting (19) and (21) into (20) proves the statement.

(c) Let $\tau = 1$. Then for all P s.t. $l(P) = \tau - 1$ the set $\mathcal{F}(P)$ is just a node, and for that node the lemma holds, as

$$S_P[1] = z_i[1] = \alpha\Big(z_i[0] + \sum_{j\in\mathcal{P}(i)} u_{ij}[-1]\Big).$$

Now assume the lemma holds for for all blocks B s.t. $l(B) = \tau - 2$, for some fixed $\tau \le l_{\max} + 1$. Then for any P such that $l(P) = \tau - 1$, using (20) and (12) gives

$$S_P[\tau] = \sum_{i\in P} z_i[0] + \sum_{\substack{i\in P\\ j\in\mathcal{P}(i)}} u_{ij}[-1] + \sum_{C\in\mathcal{C}(P)} S_C[\tau-1].$$

Thus (c) holds for $\tau \le l_{\max} + 1$ as $l(C) = \tau - 2$.

Now we consider $S_T[\tau]$. The case of $\tau = l_{\max} + 1$ follows from above. Let $\tau = l_{\max}+1+\bar{\tau},\ \bar{\tau} \geq 1$. Repeated application of (12) to $S_T[l_{\max} + 1 + \bar{\tau}]$ gives that

$$S_T[l_{\max} + 1 + \bar{\tau}] = \alpha^{\bar{\tau}} S_T[l_{\max} + 1] - \sum_{t=0}^{\bar{\tau}-1} \left(\alpha^{\bar{\tau}-t} \sum u_{i0}[t] \right).$$

(d) The lemma holds for blocks with no children blocks, that is nodes with no children, as the dynamics gives

$$\begin{aligned} S_i[t] &= z_i[t - l(i)] \\ &= \alpha \left(z_i[t - l(i) - 1] + \sum_{j \in \mathcal{P}(i)} u_{ij}[t - l(i) - 2] \right) \\ &= \alpha m_i[t - l(i) - 1]. \end{aligned}$$

Now assume that the lemma holds for all children blocks of some block P. $S_P[t]$ can be written as as

$$S_P[t] = \sum_{i \in P} z[t - l(i)] + \frac{1}{\alpha} \sum_{C \in \mathcal{C}(P)} S_C[t].$$

Applying the dynamics for the first term gives (21). For the second term (a) gives

$$\frac{1}{\alpha} S_C[t] = S_C[t - 1] + \sum_{\substack{i \in C \\ j \in \mathcal{P}(i)}} u_{ij}[t - l(i) - 2].$$

Using the induction assumption we have that $S_C[t - 1] = \alpha m_C[t - l(C) - 2] = \alpha m_C[t - l(P) - 1]$. The internal flows cancel and thus P also satisfies the lemma. □

Proof of Lemma 3. The edge (i, j) splits the set $\mathcal{F}(P)$ into two parts, the upstream set $U(i, j)$ and the downstream set $D(i, j)$. We will first consider the dynamics on these sets when $u_{ij}[t - l(j) - 1] = 0$. This allows Lemma 1d to essentially be applied to the upstream and the downstream sets. We will then consider the effect caused by a non zero flow on u_{ij}. This will be easy as it only affects two nodes.

Consider the nodes in the upstream set that are also in P, that is U_P (recall Definition 9). When $u_{ij} = 0$ lemma 1d can be applied to U_P as if it were a block, since it would be a block if the edge (i, j) was removed. This gives

$$\sum_{k \in U(i,j)} \alpha^{l(k)-l(P)} z_k[t - l(k)] = \alpha m_U[t - l(P) - 1], \quad (22)$$

where we have used that $U(i, j) = \mathcal{F}(U_P)$ and $m_{U_P} = m_U$.

For the downstream set there are two cases. We will show that for both cases it holds that

$$\sum_{k \in D(i,j)} \alpha^{l(k)-l(P)} z_k[t - l(k)] = \alpha m_D[t - l(P) - 1]. \quad (23)$$

If the set of nodes D_P is non empty, then Lemma 1d can be applied to D_P as long as $u_{ij} = 0$,

$$\sum_{k \in \mathcal{F}(D_P)} \alpha^{l(k)-l(P)} z_k[t - l(k)] = \alpha m_{D_P}[t - l(D_P) - 1].$$

This is equivalent to (23) as $l(D_P) = l(P)$. If D_P is empty then the set of blocks D_C contains only one block with (i, j) as its only incoming link, as otherwise D_P would not be empty. With slight abuse of notation[2] Lemma 1d gives that

$$\sum_{k \in \mathcal{F}(D_C)} \alpha^{l(k)-l(D_C)} z_k[t - l(k)] = \alpha m_{D_C}[t - l(D_C) - 1].$$

However, $m_{D_C}[t - l(D_C) - 1] = \alpha m_{D_C}[t - l(D_C) - 2]$ when the flow on (i, j) is zero, as then there is no inflow to the block m_{D_C}. As $l(D_C) = l(P) - 1$ it thus holds that

$$\frac{1}{\alpha} \sum_{k \in \mathcal{F}(D_C)} \alpha^{l(k)-l(D_C)} z_k[t - l(k)] = \alpha m_{D_C}[t - l(P) - 1].$$

Finally, using that $1/\alpha \cdot \alpha^{-l(D_C)} = \alpha^{-l(P)}$ gives

$$\sum_{k \in \mathcal{F}(D_C)} \alpha^{l(k)-l(P)} z_k[t - (l(k) - 1)] = \alpha m_{D_C}[t - l(P) - 1].$$

which is equivalent to (23).

Now consider the effect of changing to a non zero value for $u_{ij}[t - l(i) - 1]$ while keeping all other flows constant on (22) and (23). All levels other than the source j and the destination i will be unaffected by this change. The source will decrease its level by u and the destination will increase its level by αu. However, the destination node is weighted by $1/\alpha$, which gives the lemma. □

Proof of Lemma 4. Decreasing m_i by ϵ allows m_j to be increased by

$$\epsilon \frac{\alpha^{l(i)-l(P)}}{\alpha^{l(j)-l(P)}}.$$

At optimality it must thus hold that

$$q_i m_i = q_j m_j \frac{\alpha^{l(i)-l(P)}}{\alpha^{l(j)-l(P)}}.$$

Which can be rewritten as

$$m_j = \frac{q_i}{q_j} \frac{\alpha^{l(j)}}{\alpha^{l(i)}} m_i.$$

The constraint can then be rewritten as

$$\alpha^{l(i)-l(P)} m_i + \sum_{\substack{j \in P \\ j \neq i}} \alpha^{l(j)-l(P)} \frac{q_i}{q_j} \frac{\alpha^{l(j)}}{\alpha^{l(i)}} m_i = m,$$

which gives that

$$\frac{m_i q_i}{\alpha^{l(i)-l(P)}} \left(\sum_{j \in P} \frac{\alpha^{2(l(j)-l(P))}}{q_j} \right) = m.$$

From which the expression for the optimal m_i follows. The optimal value is achieved by inserting the optimal value for each m_i into the cost function:

$$\sum_{i \in P} q_i m_i^2 = \gamma^2 m^2 \sum_{i \in P} q_i \left(\frac{\alpha^{l(i)-l(P)}}{q_i} \right)^2 = \gamma m^2$$

□

[2]Note that D_C is defined to be a set of blocks. However, here the set only contains one block, and we use it to describe that block.

REFERENCES

[1] P. Kundur, *Power System Stability and Control*. McGraw-Hill Professional, 1994.

[2] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.

[3] S. V. Ukkusuri and K. M. A. Özbay, *Advances in Dynamic Network Modeling in Complex Transportation Systems*. Springer, 2013.

[4] H. R. Varian, *Intermediate Microeconomics:A Modern Approach*, 6th ed. W. W. Norton & Company, 2003.

[5] K. Subramanian, J. B. Rawlings, C. T. Maravelias, J. Flores-Cerrillo, and L. Megan, "Integration of control theory and scheduling methods for supply chain management," *Computers & Chemical Engineering*, vol. 51, pp. 4–20, 2013.

[6] R. Evans, L. Li, I. Mareels, N. Okello, M. Pham, W. Qiu, and S. K. Saleem, "Real-time optimal control of river basin networks," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11 459–11 464, 2011.

[7] R. Radner, "Team decision problems," *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 857–881, 1962.

[8] M. Rotkowitz and S. Lall, "A characterization of convex problems in decentralized control," *IEEE transactions on Automatic Control*, vol. 50, no. 12, pp. 1984–1996, 2005.

[9] J. Anderson, J. C. Doyle, S. H. Low, and N. Matni, "System level synthesis," *Annual Reviews in Control*, 2019.

[10] Y. Wang, N. Matni, and J. C. Doyle, "Separable and localized system-level synthesis for large-scale systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4234–4249, 2018.

[11] A. Rantzer, "Realizability and internal model control on networks," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3475–3477.

[12] R. D'Andrea and G. E. Dullerud, "Distributed control design for spatially interconnected systems," *IEEE Transactions on automatic control*, vol. 48, no. 9, pp. 1478–1495, 2003.

[13] P. Shah and P. A. Parrilo, "$\mathcal{H}_2$-optimal decentralized control over posets: A state-space solution for state-feedback," *IEEE Transactions on Automatic Control*, vol. 58, no. 12, pp. 3084–3096, 2013.

[14] K. Mårtensson and A. Rantzer, "Gradient methods for iterative distributed control synthesis," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*. IEEE, 2009, pp. 549–554.

[15] F. Lin, M. Fardad, and M. R. Jovanović, "Design of optimal sparse feedback gains via the alternating direction method of multipliers," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2426–2431, 2013.

[16] B. Bamieh, F. Paganini, and M. A. Dahleh, "Distributed control of spatially invariant systems," *IEEE Transactions on automatic control*, vol. 47, no. 7, pp. 1091–1107, 2002.

[17] C. Lidström and A. Rantzer, "Optimal H_∞ state feedback for systems with symmetric and hurwitz state matrix," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 3366–3371.

[18] D. Madjidian and L. Mirkin, "Distributed control with low-rank coordination," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 53–63, 2014.

[19] M. Heyden, R. Pates, and A. Rantzer, "A structured linear quadratic controller for transportation problems," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 1654–1659.

[20] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[21] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 1, no. 2.