Constrained Flow Control in Storage Networks: Capacity Maximization and Balancing *

Claus Danielson a, Francesco Borrelli a, Douglas Oliver b, Dyche Anderson b, Tony Phillips b

a University of California, Berkeley, CA 94720-1740, USA.
b Ford Research and Advanced Engineering, Dearborn, MI 48124-5029, USA.

Abstract

This paper studies the control of distributed storage networks with guarantees of constraints satisfaction and asymptotic stability. We consider two problems: the network capacity maximization and the network balancing problems. In the first part of the paper we describe the two problems, highlight their importance in a wide number of engineering applications, and compare them by analyzing the properties of their solutions. In the second part we present algorithms for solving both problems by using a convex one-step model predictive controller (MPC) which guarantees persistent state and flow constraints satisfaction. We present simple conditions which link the network topology, the MPC weights and the asymptotic stability of the closed-loop system. A numerical example illustrates the effectiveness of the proposed approach.

Key words: model predictive control, storage networks, load balancing, capacity maximization, reachable sets

1 Introduction

A storage network is a collection of devices for storing some type of resource. Examples include water resource management [40], [33], [31], [21], [45], [17], [39], distributed energy storage [43], [34], [1], warehouse inventory management [41], [19], [42], [2], [28], [30] and storage area networks for data storage [27], [46], [15], [47], [38], [16]. The purpose of a storage network is to store and supply resource to meet local demands. Network balancing and network capacity maximization address the problem of controlling the flow of resources through the storage network to minimize the likelihood that a local storage device cannot satisfy local demand to store or supply resources.

In a balanced state each of the storage devices has the same percentage level of the resource relative to its capacity. In a well-designed storage network, the storage devices will be sized to reflect local demand. Therefore balancing the storage devices ensures that no local storage device becomes the “weak-link” in the network. Balancing is important in many applications. In water resource management balancing ensures that each reservoir is equal prepared to handle a local drought or sudden rain-fall [33],[45]. In distributed energy storage a balanced network is better prepared to handle local fluctuations in power demand or supply from renewable sources [1]. In inventory management a balanced network is less likely to experience stock-out or excess warehousing costs [28], [30]. In cloud computing balancing ensures faster average access to data and safer systems [38],[15],[47],[16]. Balancing is also known as consensus in the literature [36],[35],[13],[44],[29].

The ability of the storage network to meet demand is limited by the storage device with the smallest capacity. The smallest capacity storage device should be the first to run out of the resource or storage space. However if the flow through the storage network is poorly controlled, larger storage elements may run out of resource or space before the smallest element. In a capacity maximizing state every storage device can supply and store at least as much resource as the smallest storage device [18].

This paper presents the theory of balancing and capacity maximization for storage networks. In particular, we adopt a model where each node of the network represents a discrete-time single-state integrator whose state

Footnote 1: This research was supported by Ford Research Laboratories and the National Science Foundation under Grant n. 0844456
Email address: claws@berkeley.edu (Claus Danielson).
corresponds to the amount of resource stored within the node. The amount of resource is regulated by the control input associated to the node. Two types of control inputs can be used: redistributive and dissipative control inputs. The redistributive control inputs transfer an amount of resource stored in the node to other nodes in the network. Equally a node can receive some amount of resource from other nodes through network links. Using redistributive control inputs, a control system can transfer resources from storage devices that are relatively full into storage devices that are relatively empty. The method of redistributive balancing depends on the application. For instance in water resource management, systems of canals and pumping stations are used to transfer water between reservoirs [33], [21]. In distributed energy storage, energy is transmitted passively or actively between storage devices [1]. For inventory management, trucks are used to move goods between warehouses [2], [19], [28]. In cloud computing data is transfer between storage devices over a network [38], [15], [47].

Through dissipative control inputs, a control system can balance the networks by discarding excess resource in certain storage devices. For instance in water resource management, reservoirs can be discharged into local bodies of water [31], [33]. In distributed energy storage a demand response signals incentives the discharge local storage devices [43], [34]. In inventory management, excess inventory is disposed of by sales promotions, price discounts, and selling to outlet retailers [42]. In cloud computing, redundant data can be archived or if necessary deleted [38], [46]. In most applications excessive dissipative balancing is undesirable but often necessary.

Every control action, whether redistributive or dissipative, has an associated cost. In water resource management canals leak and pumping stations require power to operate. Water discharged into the ocean or neighboring jurisdictions through rivers cannot be recovered. In inventory management, there are literal costs associated transporting goods and selling inventory at a reduced rate. In distributed energy storage a portion of the stored energy is lost during transmission and all of the energy is lost during discharge. In cloud computing there are cost in the form of increased network traffic from moving data and delays for recovering archived data. And obviously deleting data is an option of last resort. It is important that the control act intelligently to balance the storage network.

In each application there are constraints on the control action. In particular there are state constraints restricting the storage devices from being less than zero and more than one-hundred percent filled. There are input constraints that limit the redistribution and dissipation of resource from the storage network. These input constraints are potentially time-varying.

The closely related problem of flows in static networks has been treated extensively, starting from early classical works [25], [23] to more recent monographs [37], [4], [24]. In [25], [23], [37], [24] the nodes are not dynamical systems but represent merely topological elements. Robust control of dynamic networks subject to uncertain demands is treated in a series of papers by Blanchini and co-authors [12], [10], [8], [9], [7], both for the discrete-time as well as for continuous-time case, where the disturbances are modeled as non-stochastic and of bounded magnitude. The approach taken in [12], [11], [10], [8] is to characterize a robust control invariant (RCI) set, i.e., the set of network states for which there exists a network flow that guarantees the demand satisfaction at all time instants.

In this paper we address the problems of Network balancing and network capacity maximization in a unified way using constrained optimal control. We use minimal assumptions on the nature of the input constraint sets thus our method is applicable to a diverse variety of storage network flow problems. The paper is structured in three parts. In the first part we formally define the capacity maximization and network balancing problems. We show that the set of states maximizing capacity includes the the set of balanced states. In the second part we present model predictive control algorithms to solve the capacity maximization and network balancing problems. We show that the proposed controllers guarantee persistent feasibility and present simple conditions which link the network topology, the MPC weights and the asymptotic stability of the closed-loop system. Finally we present a numerical example to demonstrate the relevant features of the algorithms.

1.1 Notation and Basic Definitions

Let $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{N}$ denote the set of real numbers, non-negative real numbers, and non-negative integers respectively.

The following definitions are taken from [3], [6]. A polytope $\mathcal{X} \subset \mathbb{R}^n$ is the intersection of a finite number of half-spaces. The Chebyshev ball of a polytope $\mathcal{X} \subset \mathbb{R}^n$ is the largest radius ball $B(c, r) \subset \mathbb{R}^n$ contained in $\mathcal{X}$ where $c$ is the ball center and $r$ is the radius. If its Chebyshev ball has zero radius $r = 0$ then the set $\mathcal{X}$ is called lower-dimensional. Otherwise it is called full-dimensional. Let $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subset \mathbb{R}^n$ the Minkowski sum is defined as $\mathcal{X} \oplus \mathcal{Y} = \{x + y \mid x \in \mathcal{X}, y \in \mathcal{Y}\}$. Let $M \in \mathbb{R}^{n \times m}$ be a matrix and $\mathcal{X} \subset \mathbb{R}^n$ be a set, we define their composition as $M \circ \mathcal{X} = \{Mx \in \mathbb{R}^n \mid x \in \mathcal{X}\}$. Let $M \in \mathbb{R}^{n \times m}$ be a matrix then the convex cone of $M$ is $\text{cone}(M) = \{M\lambda \in \mathbb{R}^n \mid \lambda \geq 0 \in \mathbb{R}^m\}$.

The following definitions are taken from [26], [5]. A network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a set of vertices $\mathcal{V}$ together with a list of unordered pairs $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ called edges. Vertices $i, j \in \mathcal{V}$ are called adjacent if $(i, j) \in \mathcal{E}$ is an edge. A path
is a sequence of adjacent vertices. Two vertices \( i, j \in \mathcal{V} \) are called connected if there exists a path connecting them. A network is connected if every pair of vertices in the network are connected. The distance \( \Delta(i, j) \) between two vertices \( i, j \in \mathcal{V} \) is the length of the shortest path connect them. With abuse of notation we define the distance matrix \( \Delta : \mathbb{N}^{|\mathcal{V}| \times |\mathcal{V}|} \) where the \( ij \)-th entry is the distance between the \( i \)-th and \( j \)-th vertices \( \Delta_{ij} = \Delta(i, j) \). An orientation \( \sigma \) assigns a positive direction to each edge in the network. The contraction of a set of vertices \( \mathcal{U} \subset \mathcal{V} \) merges all the vertices in \( \mathcal{U} \) into a single vertex \( k \) such that \( k \) is adjacent to the union of vertices to which \( i \in \mathcal{U} \) are adjacent. Formally let \( f \) be a function that maps vertices in \( \mathcal{V} \setminus \mathcal{U} \) to itself and vertices in \( \mathcal{U} \) to \( k \). Then the vertex contraction of \( \mathcal{U} \) is the network \( \mathcal{G}' = (\mathcal{V}', \mathcal{E}') \) where \( \mathcal{V}' = f(\mathcal{V}) \) and \( \mathcal{E}' = f(\mathcal{E}) \) with self-loops removed.

The power set of the set \( \mathcal{X} \) is the set of all subsets of \( \mathcal{X} \) and is denoted by \( 2^\mathcal{X} \). A set-valued function \( F : \mathcal{X} \rightrightarrows \mathcal{Y} \) maps points from \( \mathcal{X} \) to subsets in \( 2^\mathcal{Y} \). A set-valued function \( F : \mathcal{X} \rightrightarrows \mathcal{Y} \) is upper semi-continuous at \( x \in \mathcal{X} \) if for every open neighborhood \( U_y \) of \( F(x) \subset \mathcal{Y} \) there exists a neighborhood \( U_x \subset \mathcal{X} \) of \( x \) such that \( F(z) \subset U_y \) for all \( z \in U_x \).

Consider the general discrete-time system

\[
x(t+1) = f(x(t), u(t))
\]

define on \( \mathcal{X} \subseteq \mathbb{R}^n \). The set of equilibrium points \( \mathcal{X}^e \subseteq \mathcal{X} \) is Lyapunov stable if for all \( \epsilon > 0 \) and for all \( x_1 \in \mathcal{X}^e \) there exists \( \delta > 0 \) and there exists \( x_2 \in \mathcal{X}^e \) such that

\[
\|x(0) - x_1\| < \delta \Rightarrow \|x(t) - x_2\| < \epsilon
\]

for all \( t \geq 0 \). The set of equilibria \( \mathcal{X}^e \) is asymptotically stable if it is stable and \( x(t) \) converges to some point \( x^e \in \mathcal{X}^e \). The following theorem on set-valued Lyapunov functions provides a method for proving the stability of systems with sets of equilibrium points.

**Theorem 1** Let \( \mathcal{X}^e \) be a set of equilibrium points and let \( W : \mathcal{X} \rightrightarrows \mathcal{X} \) be an upper semi-continuous set-valued Lyapunov function satisfying

1. \( x \in W(x) \) for all \( x \in \mathcal{X} \) and \( W(x^e) = \{x^e\} \) for all \( x^e \in \mathcal{X}^e \)
2. \( W(x(t + 1)) \subseteq W(x(t)) \) for all \( x(t) \in \mathcal{X} \)
3. There exists a function \( \mu : \text{Im}(W) \to \mathbb{R}_{\geq 0} \), bounded on bounded sets, such that

\[
\mu(W(x(t+1))) < \mu(W(x(t)))
\]

for all \( x(t) \in \mathcal{X} \setminus \mathcal{X}^e \).

Then \( \mathcal{X}^e \) is stable and \( x(t) \to x^e \) for some \( x^e \in \mathcal{X}^e \) as \( t \to \infty \).

**PROOF.** See Theorem 4 in [32]. □

## 2 Network Storage System

In this Section we describe the system dynamics, constraints, and control objectives of the network storage system.

We consider a collection of storage devices which exchange resources through a network. We consider two classes of storage devices; permanent and temporary. The temporary storage devices serve as intermediary storage for transferring resources between the permanent storage devices. The state \( x_i \in [0, 1] \) of a storage device is defined as the amount of stored resource normalized by the total storage capacity \( C_i \) of the device.

The \( i \)-th storage device is modeled as a discrete-time integrator

\[
C_i x_i(t+1) = C_i x_i(t) + \phi_i(t) \text{ for } i = 1 \ldots n
\]

where \( C_i x_i \) is the total amount of resource stored in the \( i \)-th storage device and \( \phi_i \) is the total resource added or removed from the storage device. Equation (4) is the mass-balance of stored resource in the \( i \)-th storage device. \( S \subseteq \{1, \ldots, n\} \) is the set of permanent storage device indices and \( T = \{1, \ldots, n\} \setminus S \) is the set of temporary storage device indices with \( |S| = n_s \) and \( |T| = n_t \).

There are two methods to control the state \( x_i \) through \( \phi_i \). The first method is through redistribution of the resources in the collection of storage devices. The second method is through dissipation of excess resources.

In this paper we consider three types of systems; systems with both redistributive and dissipative balancing, system with only redistributive balancing, and systems with only dissipative balancing.

In redistributive balancing the storage devices are connected in a network \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \) where the set of vertices \( \mathcal{V} = S \cup T (|\mathcal{V}| = n) \) are the storage devices and set of edges \( E \subset V \times V (|E| = m) \) are physical links that allow the temporary storage devices to exchange resources. The control variable is the resource flow \( u_{ij}(t) \) between adjacent storage devices \( (i, j) \in \mathcal{E} \). Thus the model (4) of the \( i \)-th cell becomes

\[
C_i x_i(t + 1) = C_i x_i(t) + \sum_{j \in N_i} u_{ij}(t) - u_{di}(t)
\]

where \( N_i = \{ j | (i, j) \in \mathcal{E} \} \) is the set of storage devices neighboring the \( i \)-th storage device and \( u_{di}(t) \) is the dissipation of resources from the \( i \)-th device.

We assign a label \( k \) to each edge \( (i, j) \in \mathcal{E} \). The bijection \( c(k) \) maps each label \( k \) to an unordered pair \( (i, j) \in \mathcal{E} \).
We also assign an orientation $\sigma_k$ to each edge $e(k) = (i, j) \in E$. The orientation specifies whether $i \rightarrow j$ or $j \rightarrow i$ is the positive direction for resource transfer. For $e(k) = (i, j) \in E$ with orientation $i \rightarrow j$ we have $\sigma_k(i) = -1$ and $\sigma_k(j) = +1$.

The resource flows are collected in a vector $u_r \in \mathbb{R}^m$ where $u_{r,k}(t) = u_{i,j}(t)$ for $e(k) = (i, j)$. With this notation the dynamics of the $i$-th node are

$$C_ix_i(t + 1) = C_ix_i(t) + B_iu_r(t) - u_{d,i}(t)$$

(6)

where $B_i^T \in \mathbb{R}^{m}$ is the $i$-th row of the incidence matrix $B$ associated to the network $\mathcal{G}$ with entries

$$B_{ik} = \begin{cases} \sigma_k(i) & \text{if } e(k) = (i, j) \text{ for some } j \in \mathcal{V} \\ 0 & \text{otherwise.} \end{cases}$$

The collective dynamics for the system are

$$x(t + 1) = x(t) + C^{-1}(Bu_r(t) - u_d(t))$$

(8)

where $x = [x_1 \ldots x_n]^T \in \mathbb{R}^n$ collects the state of the storage devices and $C = \text{diag}([C_1 \ldots C_n]) \in \mathbb{R}^{n \times n}$ is the matrix with the storage capacities on the diagonal.

2.1 Constraints

The storage device states must satisfy the box constraints $x(t) \in \mathcal{X} \subseteq \mathbb{R}^n$ for all $t \in \mathbb{N}$ where

$$\mathcal{X} = \left\{ x \in \mathbb{R}^n \mid \underline{\xi}_s \leq x_i(t) \leq \overline{\xi}_s \text{ for } i \in \mathcal{S} \right\}$$

(9)

Here $\underline{\xi}_s$ and $\overline{\xi}_s$ are the lower and upper bounds on the permanent storage device states respectively, and $\underline{\xi}_t$ and $\overline{\xi}_t$ are the lower and upper bounds on the temporary storage states respectively. These bounds account for uncertainty due to state estimation errors and unmodeled dynamics.

The constraints on the network flow $u_r(t)$ and dissipation $u_d(t)$ are potentially coupled and time varying

$$\begin{bmatrix} u_r(t) \\ u_d(t) \end{bmatrix} \in \mathcal{U}(t) \subseteq \mathbb{R}^m \times \mathbb{R}^n_{\geq 0}$$

(10)

where $\mathcal{U}(t)$ is polytopic for all $t \in \mathbb{N}$. The dissipations can only remove resource from the network thus $\text{Proj}_{u_d}(\mathcal{U}) \subseteq \mathbb{R}^n_{\geq 0}$ is a subset of the positive cone. In some application the redistributive $u_r(t)$ and dissipative $u_d(t)$ inputs are coupled.

We consider three types of systems:

- Case 1: Systems with redistributive and dissipative balancing.
- Case 2: Systems with only redistributive balancing.
- Case 3: Systems with only dissipative balancing.

The specific properties of $\mathcal{U}(t)$ depend on the type of system we are considering. The following assumptions are made in each of the cases.

**Assumption 1 (Redistribution and Dissipation)**

For systems with both redistributive and dissipative balancing the following properties hold:

a) There exists $\mathcal{U}_r \subseteq \mathbb{R}^m$ and $\mathcal{U}_d \subseteq \mathbb{R}^n_{\geq 0}$ such that $\mathcal{U}_r \times \mathcal{U}_d \subseteq \mathcal{U}(t)$ for all $t \in \mathbb{N}$.

b) $\mathcal{U}_r$ is full-dimensional and contains the origin in its interior.

c) $\mathcal{U}_d$ contains the origin and there exists a subspace $\mathbb{R}^{n' \prime} \subset \mathbb{R}^n$ ($0 < n' \prime \leq n$) for which $\mathcal{U}_d$ is full-dimensional.

Assumption a) states that the time-varying set $\mathcal{U}(t)$ is under-bounded by a fixed set for all $t \in \mathbb{N}$. This ensures that any control inputs $u_r(t) \in \mathcal{U}_r$ and $u_d(t) \in \mathcal{U}_d$ are available for all $t \in \mathbb{N}$. In b) the requirement that $\mathcal{U}_r$ is full-dimensional and contains the origin in its interior ensures that bidirectional flow is allowed on every edge at every time $t \in \mathbb{N}$. In c) the requirement that $\mathcal{U}_d$ contain the origin ensures that it is always possible to not dissipate any resource. The requirement that $\mathcal{U}_d$ is full-dimensional in $\mathbb{R}^{n' \prime}$ ensures that it is always possible to dissipate some resource $u_d(t) \neq 0 \in \mathcal{U}_d$. Note that $\mathcal{U}_d$ is not assumed to be full-dimensional in $\mathbb{R}^n$. This means that only certain nodes in the network can dissipate resource.

**Assumption 2 (Only Redistribution)**

For systems with only redistributive balancing the following properties hold:

a) There exists $\mathcal{U}_r \subseteq \mathbb{R}^m$ such that $\mathcal{U}_r \subseteq \mathcal{U}(t)$ for all $t \in \mathbb{N}$ where $\mathcal{U}_r$ is full-dimensional and contains the origin in its interior.

b) $\text{Proj}_{u_d}(\mathcal{U}(t)) = \{0\}$ for all $t \in \mathbb{N}$.

Assumption a) ensures that any control $u_r(t) \in \mathcal{U}_r$ is available for all $t \in \mathbb{N}$. Bidirectional flow is allowed on all edges since $\mathcal{U}_r$ is full-dimensional and contains the origin in its interior. Assumption b) states that dissipative balancing is not possible in systems with only redistributive balancing.

**Assumption 3 (Only Dissipation)**

For systems with only dissipative balancing the following properties hold:

a) There exists $\mathcal{U}_d \subseteq \mathbb{R}^n_{\geq 0}$ such that $\mathcal{U}_d \subseteq \mathcal{U}(t)$ for all $t \in \mathbb{N}$ where $\mathcal{U}_d$ is full-dimensional and contains the origin.
b) \( \text{Proj}_{u_d}(U(t)) = \{0\} \) for all \( t \in \mathbb{N} \).

c) \( \mathcal{T} = \emptyset \).

Assumption a) ensures that any control \( u_d(t) \in U_d \) is available for all \( t \in \mathbb{N} \). Every storage device can dissipate resource since \( U_d \) is full-dimensional. This assumption is required to balance networks with only dissipative balancing. Assumption b) states that redistributive balancing is not possible in system with only dissipative balancing. Assumption c) states that storage networks with only dissipative balancing do not contain temporary storage. Temporary intermediate storage is not needed for systems without the ability to redistribute resource.

3 Network Reachable Sets

In this section we introduce the set of all network states which can be reach from an initial state \( x_0 = x(0) \) by choosing an appropriate control action. This will be used in the next section to study the properties of the feedback controller.

**Definition 1 (N-step Reachable Set)** The \( N \)-step reachable set from \( x_0 \in \mathcal{X} \) for system (8) subject to the constraints (9) and (10) is defined recursively as

\[
R_{k+1}(x_0) := \{ x^+ \in \mathcal{X} \mid \exists x \in R_k(x_0) \text{ and } \exists [u_r, u_d]^T \in U(k) \text{ s.t. } x^+ = x + C^{-1}(Bu_r - u_d) \}
\]

for \( k = 1, \ldots, N-1 \) and \( R_0(x_0) = \{ x_0 \} \).

**Definition 2 (∞-step Reachable Set)** The \( \infty \)-step reachable set from \( x_0 \in \mathcal{X} \) for system (8) subject to the constraints (9) and (10) is defined as

\[
R_\infty(x_0) := \lim_{N \to \infty} R_N(x_0).
\]

The set of reachable states for system (8) subject to the constraints (9) and (10) is characterized by Proposition 1 presented next. The proposition provides the reachable sets under Assumptions 1, 2 and 3 defined in Section 2.1.

**Proposition 1** Let \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \) be connected.

1. The \( \infty \)-step reachable set for the system (8) subject to the constraints (9) and (10) satisfying Assumption 1 is

\[
R_\infty(x_0) = \{ x \in \mathcal{X} \mid 1^T C x \leq 1^T C x_0 \}.
\]

2. The \( \infty \)-step reachable set for the system (8) subject to the constraints (9) and (10) satisfying Assumption 2 is

\[
R_\infty(x_0) = \{ x \in \mathcal{X} \mid 1^T C x = 1^T C x_0 \}.
\]

3. The \( \infty \)-step reachable set for the system (8) subject to the constraints (9) and (10) satisfying Assumption 3 is

\[
R_\infty(x_0) = \{ x \in \mathcal{X} \mid C x \leq C x_0 \}.
\]

The proof of Proposition 1 can be found in the Appendix. Proposition 1 says that system (8) under Assumption 1 can reach all states with less or equal the initial amount of resource. This is possible because the network is connected and it is always possible to remove a minimum amount of resource at each time step. Under Assumption 2 resource can only be move around the network. Thus the total amount of resource remains constant. Under Assumption 3 the amount of resource can only be decreased.

4 Control Problem Statements

In this section we define two control problems; capacity maximization and network balancing. In the capacity maximization problem the redistribution \( u_r(t) \) and dissipation \( u_d(t) \) are selected to maximize the capacity of the storage network. In the network balancing problem the redistribution \( u_r(t) \) and dissipation \( u_d(t) \) are selected to balance the states of the storage devices. We show that balancing the network maximizes the capacity.

4.1 Capacity Maximization

The purpose of the permanent storage devices is to supply or store resource for some application. The storage network fails if it cannot store or supply the requested resource. The smallest demand for resource that the network cannot satisfy is

\[
\min_{i \in \mathcal{S}} C_i (x_i - \bar{x}_i).
\]

The smallest demand to store resource that the network cannot satisfy is

\[
\min_{i \in \mathcal{S}} C_i (\bar{x}_i - x_i).
\]

In this paper the effective capacity of the storage network is defined as the smallest demand to supply resource plus the smallest demand to store resource that the network cannot accommodate:

\[
\bar{c}(x) = \min_{i \in \mathcal{S}} C_i (x_i - \bar{x}_i) + \min_{i \in \mathcal{S}} C_i (\bar{x}_i - x_i).
\]

The objective of the capacity maximization problem is to maximize the effective capacity (18) and minimize the
amount of capacity resource in temporary storage, therefore the following capacity cost function is introduced

\[ J_{\text{cap}}(x) = \min_{i \in S} C_i(x_i - \overline{x}_s) + \min_{i \in S} C_i(\overline{x}_a - x_i) - \sum_{i \in T} C_i x_i. \]  

(19)

We define the subset of reachable states that maximize the capacity cost function (19) as

\[ R^*_\infty(x_0) = \arg \max_{x \in R_\infty(x_0)} J_{\text{cap}}(x) \]  

(20)

where the definition of \( R_\infty \) depends on which case we are considering according to Proposition 1. We formally define the capacity maximization problem as follows.

**Problem 1 (Capacity Maximization)** At each time instance \( t \in \mathbb{N} \) select feasible flow \( \nu_i(t) \) and dissipation \( d_i(t) \) such that for \( x(t) \in \mathcal{X} \) the future state \( x(t+1) \in \mathcal{X} \) is feasible and the cost is maximized at steady-state i.e. \( x(t) \to x^* \in R_\infty(x_0) \) as \( t \to \infty \).

In Section 5 we propose an algorithm that solves this problem.

4.2 Network Balancing

A common approach to solving Problem 1 is to balance the network. Network balancing consists of driving all the permanent storage devices to a common state \( x_i = x_j \) for all \( i, j \in S \). The temporary storage devices are driven to the minimum state \( x_i = \overline{x}_s \) for \( i \in T \). We define the set of reachable balanced states as

\[ R_\infty(x_0) = \{ x \in R_\infty(x_0) \mid x_i = x_j \forall i, j \in S \}
\]

and \( x_i = \overline{x}_s \forall i \in T \}. \]

(21)

The set of initial conditions \( x_0 \) for which this set \( R_\infty \) is non-empty is characterized by the following proposition.

**Proposition 2**

1. Let the set of reachable states \( R_\infty(x_0) \) be given by (13) under Assumption 1. Then the set \( R_\infty(x_0) \) is non-empty for all initial conditions \( x_0 \in \mathcal{X}_0 = \mathcal{X} \).

2. Let the set of reachable states \( R_\infty(x_0) \) be given by (14) under Assumption 2. Then the set \( R_\infty(x_0) \) is non-empty for all initial conditions \( x_0 \in \mathcal{X}_0 \) where

\[ \mathcal{X}_0 = \left\{ x_0 \in \mathcal{X} \mid \sum_{i \in V} C_i x_{0i} \leq \sum_{i \in S} C_i \overline{x}_s + \sum_{i \in T} C_i \overline{x}_t \right\}. \]  

(22)

3. Let the set of reachable states \( R_\infty(x_0) \) be given by (15) under Assumption 3. Then the set \( R_\infty(x_0) \) is non-empty for all initial conditions \( x_0 \in \mathcal{X}_0 = \mathcal{X} \).

**Proof.** For \( R_\infty(x_0) \) to be non-empty for each \( x_0 \in \mathcal{X}_0 \) there must exist at least one state \( \hat{x} \) satisfying the conditions

a) Balanced: \( \hat{x} = \hat{x}_i \forall i, j \in S \) and \( \hat{x} = \overline{x}_s \forall i \in T \)

b) Feasible: \( \hat{x} = [\overline{x}_s, \overline{x}_s] \) for all \( i \in S \) and \( \hat{x} = [\overline{x}_t, \overline{x}_t] \) for all \( i \in T \)

c) Reachable: \( \hat{x} \in R_\infty(x_0) \).

1. For systems with both redistributive and dissipative balancing we show that the state

\[ \hat{x}_i = \begin{cases} \overline{x}_s & i \in S \\ \overline{x}_t & i \in T \end{cases} \]  

(23)

satisfies conditions a)-c). Clearly the state in (23) satisfies the balance condition a) and feasibility condition b). It also satisfies the reachability condition c) for all \( x_0 \in \mathcal{X} \)

\[ 1^T C \hat{x} = \sum_{i \in S} C_i \overline{x}_s + \sum_{i \in T} C_i \overline{x}_t \leq 1^T C x_0 \]  

(24)

since \( x_0 \) is feasible i.e. \( x_{0i} \geq \overline{x}_s \forall i \in S \) and \( x_{0i} \geq \overline{x}_t \forall i \in T \).

2. For systems with only redistributive balancing let us parameterize all \( \hat{x} \) that satisfy condition a) by

\[ \hat{x}_i = \begin{cases} \alpha & i \in S \\ \overline{x}_t & i \in T \end{cases} \]  

(25)

where \( \alpha \) is the common state of the permanent storage devices. In order to satisfy condition c) where \( R_\infty(x_0) \) is given by (14), \( \alpha \) must satisfy

\[ \sum_{i \in S} C_i \alpha + \sum_{i \in T} C_i \overline{x}_t = 1^T C x_0. \]  

(26)

In order to satisfy condition b) we need \( \alpha \in [\overline{x}_s, \overline{x}_s] \). Since \( x_0 \) is feasible \( (x_{0i} \geq \overline{x}_s \forall i \in S \) and \( x_{0i} \geq \overline{x}_t \forall i \in T) \) we have

\[ \sum_{i \in S} C_i \alpha + \sum_{i \in T} C_i \overline{x}_t = 1^T C x_0 \]  

(27)

\[ \geq \sum_{i \in S} C_i \overline{x}_s + \sum_{i \in T} C_i \overline{x}_t. \]  

(28)

Subtracting \( \sum_{i \in T} C_i \overline{x}_t \) from each side and noting that \( \alpha \) and \( \overline{x}_s \) are constant over \( i \in S \) we conclude that \( \alpha \geq \overline{x}_s \).
For $\alpha \leq \bar{\pi}$, we need the following inequality to hold

$$\sum_{i \in S} C_i \alpha = \sum_{i \in V} C_i x_{0i} - \sum_{i \in T} C_i \bar{x}_i \leq \sum_{i \in S} C_i \bar{\pi}.$$  \hfill (29)

This is the condition in (22).

3. For systems with only dissipative balancing there is no temporary storage $T = \emptyset$. Let $\bar{x}_i = \min_{i \in S} x_{0i}$ for each $i \in S$. This state satisfies the balance condition a). It is feasibility since $x_0 \in \mathcal{X}$ is feasible. It is also reachable $\bar{x} \in \mathcal{R}_\infty(x_0)$ where the set of reachable state $\mathcal{R}_\infty(x_0)$ is given by (15) since $C\bar{x} \leq Cx_0$.

Proposition 2 describes the restrictions on the initial conditions such that the storage network can reach a balanced state. In cases where dissipative balancing is available it is always possible to reach a balanced state from any initial condition. However when only redistributive balancing is available there is a restriction that the total initial amount of resource in the network must be smaller than the total amount that can be stored in the permanent storage devices.

We formally define the network balancing problem as follows.

**Problem 2 (Network Balancing)** At each time instance $t \in \mathbb{N}$ select feasible flow $u_r(t)$ and dissipation $u_d(t)$ such that for $x(t) \in \mathcal{X}$ the future state $x(t+1) \in \mathcal{X}$ is feasible and the state converges to a point in the set of balanced states i.e. $x(t) \to \bar{x} \in \mathcal{R}_\infty(x_0)$ as $t \to \infty$.

Clearly Problem 2 is only feasible for $x_0 \in \mathcal{X}_0$ as defined in Proposition 2. In Section 5 we propose an algorithm that solves this problem.

In the following theorem we will show that the set of balanced states $\mathcal{R}_\infty(x_0)$ is a subset of the capacity maximizing states $\mathcal{R}_\infty^*(x_0)$. For an algorithm that solves Problem 2 can be used to solve Problem 1. We will show that the converse is not necessarily true through a numerical example in Section 6.

**Theorem 2** The set of balanced states is a subset of the capacity maximizing states: $\mathcal{R}_\infty(x_0) \subseteq \mathcal{R}_\infty^*(x_0)$.

**PROOF.** Note that this proof holds for each cases described in Section 2.1. The structure of $\mathcal{R}_\infty$ is incoherent.

In the case $\mathcal{R}_\infty(x_0) = \emptyset$ the proof is trivial. Let us consider the case when $\mathcal{R}_\infty(x_0) \neq \emptyset$. Let $\bar{x} \in \mathcal{R}_\infty(x_0) \subset \mathcal{R}_\infty(x_0)$. Then $\bar{x} \in \mathcal{R}_\infty^*$ if $\bar{x}$ maximizes the cost (19) over $\mathcal{R}_\infty(x_0)$. The cost $J_{\text{cap}}(x)$ at $\bar{x}$ is

$$J_{\text{cap}}(\bar{x}) = \min_{i \in S} \{C_i(\alpha - \bar{x}_i)\} + \min_{i \in S} \{C_i(\bar{\pi}_i - \bar{x}_i)\} - \sum_{i \in T} C_i \bar{x}_i$$

where $x_i = \alpha$ is the common state of the permanent storage devices. We will show that $J_{\text{cap}}(\bar{x})$ is an upper-bound on $J_{\text{cap}}(x)$ for all $x \in \mathcal{X} \supset \mathcal{R}_\infty$.

Since $\max(A - B) \leq \max A - \min B$ the cost can be upper-bounded

$$J_{\text{cap}}(x) \leq \max_{x \in \mathcal{X}} \left( \min_{i \in S} \{C_i(x_i - \bar{x}_i)\} + \min_{i \in S} \{C_i(\bar{\pi}_i - x_i)\} \right) - \min_{x \in \mathcal{X}} \sum_{i \in T} C_i x_i.$$ \hfill (31)

Clearly the second term in (31) is minimized by $x_i = \bar{x}_i$ for all $i \in T$ since the storage capacities are positive $C_i > 0$. Since $\min A - \min B \leq \min (A + B)$ the first term in (31) can be upper-bounded

$$\min_{i \in S} \{C_i(x_i - \bar{x}_i)\} + \min_{i \in S} \{C_i(\bar{\pi}_i - x_i)\} \leq \min_{i \in S} \{C_i(x_i - \bar{x}_i) + Q_i(\bar{\pi}_i - x_i)\}$$

Thus the cost (19) is upper-bounded by

$$J_{\text{cap}}(x) \leq \min_{i \in S} \{C_i(\bar{\pi}_i - x_i) - \sum_{i \in T} C_i x_i\} = J_{\text{cap}}(\bar{x})$$ \hfill (35)

for all $x \in \mathcal{X}$. Since $\bar{x} \in \mathcal{R}_\infty \subset \mathcal{R}_\infty^* \subset \mathcal{X}$ it follows that $\bar{x}$ maximizes $J_{\text{cap}}(x)$ over $\mathcal{R}_\infty$. Therefore $\bar{x} \in \mathcal{R}_\infty^*$ and thus $\mathcal{R}_\infty \subseteq \mathcal{R}_\infty^*$.

5 **Control Design: Algorithm and their Feasibility and Stability Properties**

In this section we introduce three model predictive control algorithms; one algorithm to solve Problem 1 and two algorithms, one based on linear programming and one based on quadratic programming, to solve Problem 2. The algorithms use a one-step prediction horizon. We prove these are asymptotically stable and persistently feasible under each of the Assumptions defined in Section 2.1.
5.1 Control Invariant Sets

In this section we define and derive the control invariant sets for the network storage system (8).

Definition 3 (Control Invariant Set) A set $C \subseteq X$ is a control invariant set for the system $x(t+1) = Ax(t) + Bu(t)$ subject to the constraints $x(t) \in X$ and $u(t) \in U$ if $x(t) \in C$ implies that there exists $u(t) \in U$ such that $Ax(t) + Bu(t) \in C$ for all $t \in \mathbb{N}$.

Definition 4 (Maximal Control Invariant Set) The set $C_\infty \subseteq X$ is the maximal control invariant set for the system $x(t+1) = Ax(t) + Bu(t)$ subject to the constraints $x(t) \in X$ and $u(t) \in U$ if it is control invariant and contains all control invariant sets.

Proposition 3 characterizes the maximal control invariant set for the network storage system (8).

Proposition 3 The maximal control invariant set for the system (8) subject to the constraints (9) and (10) under Assumption 1, 2, or 3 is $X$.

PROOF. Since (8) is a network of integrators, the inputs $u_r(t) = 0 \in \text{Proj}_{u_r}(U(t))$ and $u_d(t) = 0 \in \text{Proj}_{u_d}(U(t))$ are feasible under each set of assumptions in Section 2.1. Thus the set $X$ is control invariant. Furthermore $X$ is the largest control invariant set such that $C \subseteq X$.

5.2 Capacity Maximization Algorithm

In this section we present a one-step MPC algorithm for solving Problem 1. We show that the redistribution $u_r^*(t)$ and dissipative $u_d^*(t)$ control inputs produced by the algorithm are zero if and only if the system state is in the equilibrium set. We use this fact to prove that the algorithm solves Problem 1.

Algorithm 1 At each time instant $t \in \mathbb{N}$ the flow $u_r^*(t)$ and dissipation $u_d^*(t)$ are obtained by solving the optimization problem

\[
\begin{aligned}
\text{maximize} & \quad J_{\text{cap}}(x(t+1)) - \rho_r \|u_r(t)\|_2^2 - \rho_d \|u_d(t)\|_2^2 \\
\text{subject to} & \quad x(t+1) = x(t) + C^{-1}(Bu_r(t) - u_d(t)) \\
& \quad x(t+1) \in X \\
& \quad \begin{bmatrix} u_r(t) \\ u_d(t) \end{bmatrix} \in U(t).
\end{aligned}
\]

(37a)

(37b)

(37c)

(37d)

where $\rho_r > 0$ and $\rho_d > 0$.

Here $J_{\text{cap}}(x)$ has been defined in (19) and is restated below

\[
J_{\text{cap}}(x) = \min_{i \in S} \{ C_i(x_i - \bar{x}_i) \} + \min_{i \in S} \{ C_i(\bar{x}_i - x_i) \} - \sum_{i \in T} C_i x_i.
\]

(38)

Before proving this algorithm solves the capacity maximization problem 1 we need the following Lemma.

Lemma 1 Let $G = (V, E)$ be connected. Then under Assumption 1, 2, or 3, $\mu^*(t) = 0$ and $\mu^*_d(t) = 0$ if and only if if $x(t) \in R^\infty_{\infty}(x_0)$.

PROOF. See Appendix.

We now prove that Algorithm 1 solves Problem 1.

Theorem 3 Let $G = (V, E)$ be connected. Then Algorithm 1 solves Problem 1 under Assumption 1, 2, or 3.

PROOF. This proof holds for all three cases defined in Section 2.1.

In order to solve Problem 1, Algorithm 1 must maintain feasibility $x(t+1) \in X$ for $x(t) \in X$ and guarantee asymptotic convergence $x(t) \rightarrow x^* \in R^\infty_{\infty}(x_0)$ as $t \rightarrow \infty$. Persistent feasibility is guaranteed by the constraint (37c) since $X$ is a control invariant set.

We now prove asymptotic convergence $x(t) \rightarrow x^* \in R^\infty_{\infty}(x_0)$ as $t \rightarrow \infty$ using Theorem 1. Here $X^* = R^\infty_{\infty}(x)$ is the set of equilibrium states. Let $J(\cdot) = J_{\text{cap}}(\cdot)$ and $R^\infty_{\infty} = R^\infty_{\infty}(x)$ for notational simplicity. Consider the candidate set-valued Lyapunov function $W : X \Rightarrow X$ given by

\[
W(x) = \left\{ z \in R^\infty_{\infty}(x) \mid \exists v_r \in \mathbb{R}^m \text{ and } \exists v_d \in \mathbb{R}^n \text{ such that } z = x + C^{-1}(Bv_r - v_d) \text{ and } J(z) - J(x) \geq \epsilon_r \|v_r\|_2^2 + \epsilon_d \|v_d\|_2^2 \right\}
\]

(39)

where $\epsilon_r \in (0, \rho_r]$ and $\epsilon_d \in (0, \rho_d]$ are chosen such that $W(x) \cap R^\infty_{\infty} \neq \emptyset$ for any $x \in X$. Such values $\epsilon_r$ and $\epsilon_d$ exists since $R^\infty_{\infty} \subseteq R^\infty_{\infty}$ is reachable and $J(x)$ is continuous. $W(x)$ is the set of states $z$ such that the improvement is cost $J(z) - J(x)$ outweighs the unconstrained control effort required to move the state from $x$ to $z$.

First we prove that $W(x)$ is upper semi-continuous. Consider the graph of $W(x)$

\[
\text{graph}(W) = \{(x, z) \mid z \in W(x)\}.
\]

(40)
This set is closed since it is the pre-image of a continuous function. This implies $W$ is upper semi-continuous by the Closed Graph Theorem [22].

Now we show $W(x)$ satisfies condition 1 from Theorem 1. Plugging $v_r = 0$ and $v_d = 0$ into (39) we can confirm $z = x \in W(x)$. For $x^* \in \mathcal{R}_\infty^s$ we have $J(x^*) = \max_{x \in \mathcal{R}_\infty} J(x)$ thus $W(x^*) = \{x^*\}$ since there is no $z$ such that $J(z) > J(x^*)$. Thus $W(x)$ satisfies condition 1 in Theorem 1.

Next we show $W(x(t+1)) \subseteq W(x(t))$ using the optimality of $x(t+1)$ with respect to (37). Take $z \in W(x(t+1))$ then there exist $v_r^1$ and $v_d^1$ such that $z = x(t+1) + C^{-1}(Bu_r^1 - v_d^1)$ and $J(z) - \epsilon_r ||v_r^1||^2 - \epsilon_d ||v_d^1||^2 \geq J(x)$. By the definition of $x(t+1)$ there exists control inputs $v_r^2 = u_r^2(t)$ and $v_d^2 = u_d^2(t)$ such that $x(t+1) = x(t) + C^{-1}(Bu_r^2 - v_d^2)$. Thus there exists $v_r^1 + v_r^2$ and $v_d^1 + v_d^2$ such that $z = x(t) + C^{-1}(Bv_r^1 + v_r^2) - (v_d^1 + v_d^2)$. The point $z$ satisfies

$$J(z) - \epsilon_r ||v_r^1 + v_r^2||^2 - \epsilon_d ||v_d^1 + v_d^2||^2 \geq J(z) - \epsilon_r ||v_r^1||^2 - \epsilon_d ||v_d^1||^2 \geq J(x(t+1)) - \epsilon_r ||v_r^2||^2 - \epsilon_d ||v_d^2||^2 \geq J(x(t))$$

where the first inequality follows from the triangle inequality, the second inequality follows from the definition of $z \in W(x(t+1))$, third inequality follows from the choices of $\epsilon_r \in (0, \rho_r]$ and $\epsilon_d \in (0, \rho_d]$, and forth inequality follows from the optimality of (37a). Thus $z \in W(x(t))$ for each $z \in W(x(t+1))$ which implies $W(x(t+1)) \subseteq W(x(t))$. Therefore $W(x)$ satisfies condition 2 of Theorem 1.

Finally we show that $W(x)$ satisfies condition 3 from Theorem 1. Define the function

$$\mu(W(x)) = J^* - \inf_{z \in W(x)} J(z).$$

where $J^* = \max_{z \in \mathcal{R}_\infty} J(z)$. This function is bounded on bounded sets since $W(x) \subset \mathcal{R}_\infty$ is bounded and $J(z)$ is continuous. By the definition of $W(x)$ we have $\inf_{z \in W(x)} J(z) = J(x)$. Therefore $\mu(W(x)) = J^* - J(x)$.

The optimal cost (37a) satisfies

$$J(x(t)) + \rho_r ||u_r(t)||^2 + \rho_d ||u_d(t)||^2 \geq J(x(t+1))$$

where $J(x(t))$ is the cost (37a) for $u_r(t) = 0$ and $u_d(t) = 0$. From Lemma 1 we conclude $J(x(t+1)) > J(x(t))$ since $u_r(t) \neq 0$ and $u_d(t) \neq 0$. Therefore $\mu(W(x))$ satisfies condition 3 $\mu(W(x(t+1))) < \mu(W(x(t)))$ for all $x(t) \in \mathcal{R}_\infty \setminus \mathcal{R}_\infty^*$. Finally we conclude by Theorem 1 that $x(t) \to x^* \in \mathcal{R}_\infty^*$. ■

**Remark 1** Typically the stability of an equilibrium in closed-loop with a model predictive controller is proven using the value function $J^*(x)$ as a Lyapunov function. It would seem a natural extension of this method to system with sets of equilibrium points would be to use the level set $W(x) = \{z \in \mathbb{R}^n | J^*(z) \leq J^*(x)\}$ as a set-valued Lyapunov function. However this set-valued function does not satisfy the condition $W(x_e) = \{x_e\}$ for $x_e \in \mathcal{X}_e$.

### 5.3 Linear Network Balancing Algorithm

In this section we present a one-step MPC algorithm based on linear programming for solving Problem 2. We show that the redistribution $u_r^*(t)$ and dissipative $u_d^*(t)$ control inputs produced by the algorithm are zero if and only if the system state is in the equilibrium set. We use this fact to prove that the algorithm solves Problem 2.

**Algorithm 2** At each time instant $t \in \mathbb{N}$ the flow $u_r^*(t)$ and dissipation $u_d^*(t)$ are obtained by solving the optimization problem

$$\min_{u_r(t), u_d(t)} J_{bal}(x(t+1)) + \frac{\rho_r}{2} ||u_r(t)||^2 + \frac{\rho_d}{2} ||u_d(t)||^2$$

subject to $x(t+1) = x(t) + C^{-1}(Bu_r(t) - u_d(t))$

$$x(t+1) \in \mathcal{X}$$

$$u_r(t), u_d(t) \in \mathcal{U}(t).$$

Here $J_{bal}(x)$ is defined as

$$J_{bal}(x) = \max_{i \in S} \{x_i(t+1)\} - \min_{i \in S} \{x_i(t+1)\} + \sum_{i \in T} x_i(t+1).$$

The term $\max_{i \in S} x_i(t+1) - \min_{i \in S} x_i(t)$ is the difference between maximum and minimum state. It quantifies the amount of imbalance in the network. The term $\sum_{i \in T} x_i(t+1)$ is the amount of resource stored in temporary storage.

Before proving this algorithm solves the network balancing problem 2 we need the following Lemma.

**Lemma 2** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be connected, $x(t) \in \mathcal{X}_0$ defined in Proposition 2, $\rho_4 \leq \text{tr}(C)^{-1}$, and $\rho_r < \|C\Delta\|_1^{-1}$ where $\Delta$ is the network distance matrix. Then under Assumption 1, 2, or 3, $u_r^*(t) = 0$ and $u_d^*(t) = 0$ if only if $x(t) \in \mathcal{R}_\infty$.
**PROOF.** See Appendix.

We now prove that Algorithm 2 solve Problem 2.

**Theorem 4** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be connected, $x(t) \in X_0$, $\rho_u < \text{tr}(C)^{-1}$, and $\rho_c < \|CA\|_1^{-1}$. Then Algorithm 2 solves Problem 2 under Assumption 1, 2, or 3.

**PROOF.** In order to solve Problem 2, Algorithm 2 must maintain feasibility $x(t+1) \in X$ for $x(t) \in X$ and guarantee asymptotic convergence $x(t) \to \bar{x} \in \mathcal{R}_\infty$ as $t \to \infty$. Persistent feasibility is guaranteed by the constraint (47c) since $X$ is a control invariant set.

We now prove asymptotic convergence $x(t) \to \bar{x} \in \mathcal{R}_\infty$ as $t \to \infty$ using Theorem 1. Here $X^e = \mathcal{R}_\infty$ is the set of equilibrium states. Let $J(\cdot) = J_{bat}(\cdot)$ and $\mathcal{R}_\infty = \mathcal{R}_\infty(x)$ for notational simplicity. Consider the candidate set-valued Lyapunov function $W : X \to \mathcal{R}_\infty$ given by

$$ W(x) = \left\{ z \in \mathcal{R}_\infty(x) \mid \exists v_r \in \mathbb{R}^m \text{ and } \exists v_d \in \mathbb{R}^n \right.$$ such that $z = x + C^{-1}(Bv_r - v_d)$ and

$$ J(x) - J(z) \geq \epsilon_r \|v_r\|_2^2 + \epsilon_d \|v_d\|_2^2 \right\} \tag{49} $$

where $\epsilon_r \in (0, \rho_c]$ and $\epsilon_d \in (0, \rho_d]$ are chosen such that $W(x) \cap \mathcal{R}_\infty \neq \emptyset$ for any $x \in X$. Such values $\epsilon_r$ and $\epsilon_d$ exists since $\mathcal{R}_\infty \subseteq \mathcal{R}_\infty$ is reachable and $J(x)$ is continuous. $W(x)$ is the set of states $z$ such that the improvement in cost $J(x) - J(z)$ outweighs the unconstrained control effect required to move the state from $x$ to $z$.

First we prove that $W(x)$ is upper semi-continuous. Consider the graph of $W(x)$

$$ \text{graph}(W) = \{(x, z) \mid z \in W(x)\}. \tag{50} $$

This set is closed since it is the pre-image of a continuous function. This implies $W$ is upper semi-continuous by the Closed Graph Theorem [22].

Now we show $W(x)$ satisfies condition 1 from Theorem 1. Plugging $v_r = 0$ and $v_d = 0$ into (49) we can confirm $z = x \in W(x)$. For $\bar{x} \in \mathcal{R}_\infty(x_0)$ we have $J(\bar{x}) = \min_{z \in \mathcal{R}_\infty} J(x)$ thus $W(\bar{x}) = \{\bar{x}\}$ since there is no $z$ such that $J(z) < J(\bar{x})$. Thus $W(x)$ satisfies condition 1 in Theorem 1.

Next we show $W(x(t+1)) \subseteq W(x(t))$ using the optimality of $x(t+1)$ with respect to (47). Take $z \in W(x(t+1))$ then there exists $v^*_r$ and $v^*_d$ such that $z = x(t+1) + C^{-1}(Bv^*_r - v^*_d)$ and $J(z) + \epsilon_r \|v^*_r\|_2^2 + \epsilon_d \|v^*_d\|_2^2 \leq J(x)$. By the definition of $x(t+1)$ there exists control inputs $v^*_r = u^*_r(t)$ and $v^*_d = u^*_d(t)$ such that $x(t+1) = x(t) + C^{-1}(Bv^*_r - v^*_d)$. Thus there exists $v^*_r + v^*_d$ and $v^*_d + v^*_d$ such that $z = x(t) + C^{-1}(Bv^*_r + v^*_d) - v^*_d - v^*_d$.

$$ J(z) + \epsilon_u \|v^*_r + v^*_d\|_1 + \epsilon_d \|v^*_d + v^*_d\|_1 \leq J(z) + \epsilon_u \|v^*_r\|_1 + \epsilon_u \|v^*_r\|_1 + \epsilon_d \|v^*_d\|_1 + \epsilon_d \|v^*_d\|_1 \leq J(x(t+1)) + \epsilon_u \|v^*_r\|_1 + \epsilon_d \|v^*_d\|_1 \leq J(x(t+1)) + \rho_r \|u_r\|_1 + \rho_d \|u_d\|_1 \leq J(x(t)) \tag{51} $$

where the first inequality follows from the triangle inequality, the second inequality follows from the definition of $z \in W(x(t+1))$, third inequality the choices of $\epsilon_u \in (0, \rho_r]$ and $\epsilon_d \in (0, \rho_d]$, and forth inequality follows from the optimality of (47a). Thus $z \in W(x(t))$ for each $z \in W(x(t+1))$ which implies $W(x(t+1)) \subseteq W(x(t))$. Therefore $W(x)$ satisfies condition 2 of Theorem 1.

Finally we show that $W(x)$ satisfies condition 3 from Theorem 1. Let us define the function

$$ \mu(W(x)) = \sup_{z \in W(x)} J(z) - J^*. \tag{55} $$

where $J^* = \min_{x \in \mathcal{R}_\infty} J(x)$. This function is bounded on bounded sets since $W(x) \subseteq \mathcal{R}_\infty$ is bounded and $J(z)$ is continuous. By the definition of $W(x)$ we have $\sup_{z \in W(x)} J(z) = J(x)$. Therefore $\mu(W(x)) = J(x) - J^*$.

The optimal cost (47a) satisfies

$$ J(x(t+1)) + \rho_r \|u_r\|_1 + \rho_d \|u_d\|_1 \leq J(x(t)) \tag{56} $$

where $J(x(t))$ is the cost (47a) for $u_r = 0$ and $u_d = 0$. From Lemma 2 we conclude $J(x(t+1)) > J(x(t))$ since $u_r \neq 0$ or $u_d \neq 0$ for $x \not\in \mathcal{R}_\infty$. Therefore $\mu(W(x))$ satisfies condition 3 $\mu(W(x(t+1))) < \mu(W(x(t)))$ for all $x(t) \in \mathcal{R}_\infty \setminus \mathcal{R}_\infty$.

Finally we conclude by Theorem 1 that $x(t) \to \bar{x} \in \mathcal{R}_\infty$.

**Remark 2** See Remark 1 about our choice of set-valued Lyapunov function.

### 5.4 Quadratic Network Balancing Algorithm

We present an algorithms for solving Problem 2 based on quadratic programming.

**Algorithm 3** At each time instant $t \in \mathbb{N}$ the flow $u^*_r(t)$ and $u^*_d(t)$ are obtained by solving the optimization prob-
Theorem 5 Let $\mathcal{G} = (V, E)$ be connected and $x(t) \in X_0$ defined in Proposition 2. Then under Assumption 1, 2, or 3, $u_r^*(t) = 0$ and $u_d^*(t) = 0$ if and only if $x(t) \in R_\infty$.

PROOF. See Appendix.

We now prove that Algorithm 2 solve Problem 2.

The cost (57a) can be written
\[ J(x) = \|Qx(t+1)\|^2_2 + \rho_r\|u_r(t)\|^2_2 + \rho_d\|u_d(t)\|^2_2 \] (60)

where
\[ \|Qx(t+1)\|^2_2 = \|Q_Sx_S(t+1)\|^2_2 + \|Q_Tx_T(t+1)\|^2_2. \] (61)

Consider the candidate set-valued Lyapunov function $W : X \Rightarrow X$ given by
\[ W(x) = \{ z \in R_\infty(x) \mid \exists v_r \in R^n \text{ and } \exists v_d \in R^m \] such that $z = x + C^{-1}(Bv_r - v_d)$ and
\[ \|Qx\|^2_2 - \|Qz\|^2_2 \geq \epsilon_v\|v_r\|^2_2 + \epsilon_d\|v_d\|^2_2 \] (62)

where $\epsilon_v \in (0, \rho_r]$ and $\epsilon_d \in (0, \rho_d]$ are chosen such that $W(x_0) \cap R_\infty \neq \emptyset$. Such values $\epsilon_v$ and $\epsilon_d$ exists since $R_\infty \subseteq R_\infty$ is reachable and $J(x)$ is continuous.

First we prove that $W(x)$ is upper semi-continuous. Consider the graph of $W(x)$
\[ \text{graph}(W) = \{(x, z) \mid z \in W(x)\}. \] (63)

This set is closed since it is the pre-image of a continuous function. This implies $W$ is upper semi-continuous by the Closed Graph Theorem [22].

Now we show $W(x)$ satisfies condition 1 from Theorem 1. Plugging $v_r = 0$ and $v_d = 0$ into (62) we can confirm $z = x \in W(x)$. Let $x_0 \in R_\infty$ then $x = \arg\max_{x \in X} |Qx|$ since $x_i = x_j \forall i, j \in S$ minimizes $\|Q_Sx_S\|^2_2$ and $x_i = 2t$, $\forall i \in T$ minimizes $\|Q_Tx_T\|^2_2$ for $x \in X$. Thus $W(x) = \{x\}$ and $W(x)$ satisfies condition 1 in Theorem 1.

Next we show $W(x(t+1)) \subseteq W(x(t))$ using the optimality of $x(t+1)$ with respect to (57). Take $z \in W(x(t+1))$ then there exists $v_r^1$ and $v_d^1$ such that $z = x(t+1) + C^{-1}(Bv_r^1 - v_d^1)$ and $J(z) + \epsilon_v\|v_r^1\|^2_2 + \epsilon_d\|v_d^1\|^2_2 \leq J(x)$. By the definition of $x(t+1)$ there exists control inputs $u_r = u_r(t)$ and $v_d = u_d(t)$ such that $x(t+1)$ minimizes $J(x)$ subject to $v_r = v_r^1$ and $v_d = v_d^1$. Thus there exists $v_r^1$ and $v_d^1$ such that $z = x(t) + C^{-1}(Bv_r^1 - v_d^1)$ and $J(z) + \epsilon_v\|v_r^1\|^2_2 + \epsilon_d\|v_d^1\|^2_2 \leq J(x)$.

Let $x_S$ denote $x_i$ for $i \in S$ the vector of permanent storage devices and $x_T$ denote $x_i$ for $i \in T$ the vector of temporary storage devices. Define $Q_S = I_{n_S} - \frac{1}{n_S}11^T$ where $Q_T = I_{n_T}$ and let
\[ Q_{ij} = \begin{cases} Q_{Sij} & \text{for } i, j \in S \\ Q_{Tij} & \text{for } i, j \in T \\ 0 & \text{otherwise.} \end{cases} \] (59)

The cost (57a) can be written
\[ J(x) = \|Qx(t+1)\|^2_2 + \rho_r\|u_r(t)\|^2_2 + \rho_d\|u_d(t)\|^2_2 \] (60)

where
\[ \|Qx(t+1)\|^2_2 = \|Q_Sx_S(t+1)\|^2_2 + \|Q_Tx_T(t+1)\|^2_2. \] (61)

Before proving this algorithm solves the network balancing problem 2 we need the following Lemma.

Lemma 3 Let $\mathcal{G} = (V, E)$ be connected and $x(t) \in X_0$ defined in Proposition 2. Then under Assumption 1, 2, or 3, $u_r^*(t) = 0$ and $u_d^*(t) = 0$ if and only if $x(t) \in R_\infty$.

PROOF. See Appendix.
where the first inequality follows from the triangle inequality, the second inequality follows from the definition of \(z \in W(x(t + 1))\), third inequality the choices of \(\epsilon_u \in (0, \rho_u]\) and \(\epsilon_d \in (0, \rho_d]\), and third inequality follows from the optimality of (57a). Thus \(z \in W(x(t))\) for each \(z \in W(x(t + 1))\) which implies \(W(x(t + 1)) \subseteq W(x(t))\). Therefore \(W(x)\) satisfies condition 2 of Theorem 1.

Let us define the function

\[
\mu(W(x)) = \sup_{z \in W(x)} \|Qz\|_2^2
\]

(68)

This function is bounded on bounded sets since \(W(x)\) is bounded and \(\|Qz\|_2^2\) is continuous. By the definition of \(W(x)\) we have \(\sup_{z \in W(x)} \|Qz\|_2^2 = \|Qx\|_2^2\). Therefore \(\mu(W(x)) = \|Qx\|_2^2\).

The optimal cost (57a) satisfies

\[
\|Qx(t + 1)\|_2^2 + \rho_r \|u_r\|_2^2 + \rho_d \|u_d\|_2^2 \leq \|Qx(t)\|_2^2
\]

(69)

where \(\|Qx(t)\|_2^2\) is the cost (57a) for \(u_r = 0\) and \(u_d = 0\). From Lemma 1 we conclude \(\|Qx(t + 1)\|_2^2 > \|Qx(t)\|_2^2\) since \(u_r \neq 0\) or \(u_d \neq 0\) for \(x \notin \mathcal{R}_\infty\). Therefore \(\mu(W(x))\) satisfies condition 3 \(\mu(W(x(t + 1))) < \mu(W(x(t)))\) for all \(x(t) \in \mathcal{R}_\infty \setminus \mathcal{R}_\infty\).

Finally we conclude by Theorem 1 that \(x(t) \to \bar{x} \in \mathcal{R}_\infty\).

\[ \square \]

**Remark 3** See Remark 1 about our choice of set-valued Lyapunov function.

### 6 Numerical Example

We now present a numerical example for Algorithms 1, 2, and 3. Figure 1 shows a Erdos-Reyni [20] random storage network generated for this example. The network contains \(n = 50\) nodes and \(m = 102\) edges. There are \(n_s = 45\) permanent storage devices and \(n_t = 5\) temporary storage devices. The initial state of the permanent storage devices is shown in Figure 2.

![Fig. 1. Storage network containing \(n = 50\) nodes and \(m = 102\) edges. The permanent storage devices \((n_s = 45)\) are marked by the blue circles and the temporary storage devices \((n_t = 5)\) are marked by the red triangles.](image)

The simulations of system (8) in closed-loop with the Algorithms 1, 2, and 3 are shown in Figures 3-6. Figure 3 shows the state \(x(t)\) of the permanent storage devices. Figures 4 and 5 show the network flow \(u_r(t)\) and dissipation \(u_d(t)\) respectively. Note in Figure 5 the total dissipation constraint is not violated. Figure 6 shows the network capacity defined in (18).

In proposition 2 we showed that algorithms for Problem 2 (Network Balancing) will also solve the Problem 1 (Capacity Maximization). This example illustrates that the converse is not true. Algorithm 1 maximized the effective capacity of the storage network. Figure 6 shows the effective capacity of the storage network converging to the maximum value \(\min_{i \in \mathcal{S}} C_i(\bar{\pi}_i - \bar{\pi}_s)\) for each Algorithm. However Figure 3(a) shows that the state of the permanent storage devices converged to an imbalanced state. Contrast this with Figures 3(b) and 3(c) which shows that Algorithms 2 and 3 balance the states of the permanent storage devices. Thus Algorithm 1 drove the system to an equilibrium point in \(\mathcal{R}_\infty^* \setminus \mathcal{R}_\infty\).

### 7 Conclusions

We considered the network balancing problem from a constrained optimal control perspective. We defined and distinguished the storage capacity maximization and network balancing problems. We showed that network
Fig. 2. Initial state of the permanent storage devices $x_0 = x(0)$.

Fig. 3. States of the storage devices $x(t)$. Notice that for Algorithms 2 and 3 the states of the storage devices converge to a common state. However for Algorithm 1 the states of the storage devices converge to an unbalanced steady-state.

Fig. 4. Redistributive control inputs $u_r(t)$ for each Algorithm. The figures show the intensity of flow $u_r(t)$. Red indicates an edge is saturated and blue indicates the edge is inactive. The edges have been sorted by activity to provide an indication of the edge usage by each algorithm.

Balancing solves the capacity maximization problem. We presented an example that demonstrates the existence of unbalanced capacity maximizing states. Three algorithms were presented; one for solving the capacity maximization problem and two for solving the network balancing problem. We have shown the persistent feasibility and the asymptotic stability of the algorithm. In particular we have highlighted the link between the MPC cost, the network topology and the stability of the closed-loop system.

8 Appendix

Proof of Proposition 1
Fig. 5. Total dissipative control inputs \( u_d(t) \) for each Algorithm. The figures show the total resource dissipation \( \mathbf{1}^T u_d(t) \). This provides an indication of the total dissipation by each algorithm.

**PROOF.**

1. From the definition of reachable set and under Assumption 1 we have

\[
\begin{align*}
\left[ B \circ \left( \bigoplus_{t=0}^{\infty} U_t \right) \right] \times \bigoplus_{t=0}^{\infty} U_t & \subseteq C \circ (\mathcal{R}_\infty - x_0) \\
& \subseteq (B \circ \mathbb{R}^n) \oplus (-\mathbb{R}^n_{\geq 0}).
\end{align*}
\]

(71)

By Assumption 1 the set \( \bigoplus_{k=0}^{\infty} U_d = \mathbb{R}^m \) spans the flow space \( \mathbb{R}^m \) since \( U_d \) is full-dimensional and contains the origin in its interior. Since \( B \) is the incidence matrix of a connected network \( \mathcal{G} = (V, \mathcal{E}) \) it satisfies \( \text{range}(B) = B \circ \mathbb{R}^m = \text{null}(\mathbf{1}^T) \) [26]. Therefore equation (71) can be rewritten as

\[
\text{null}(\mathbf{1}^T) \times \bigoplus_{t=0}^{\infty} U_d \subseteq C \circ (\mathcal{R}_\infty - x_0) \subseteq \text{null}(\mathbf{1}^T) \oplus (-\mathbb{R}^n_{\geq 0}).
\]

(72)

The Minkowski sum \( \text{null}(\mathbf{1}^T) \oplus (-\mathbb{R}^n_{\geq 0}) \) is equal to the set \( \mathcal{P} = \{ y \in \mathbb{R}^n \mid \mathbf{1}^T y \leq 0 \} \). By Assumption 1 the set \( \bigoplus_{k=0}^{\infty} U_d \) is an unbounded subset of \( \mathbb{R}^n_{\geq 0} \) containing the origin. This implies that the Minkowski sum of \( \text{null}(\mathbf{1}^T) \) and \( \bigoplus_{k=0}^{\infty} U_d \) is also equal to the set \( \mathcal{P} \). Therefore we have

\[
\mathcal{P} \subseteq C \circ (\mathcal{R}_\infty - x_0) \subseteq \mathcal{P}.
\]

(73)

Therefore \( x_\infty \in \mathcal{R}_\infty \) if and only if \( \mathbf{1}^T (C x_\infty - C x_0) \leq 0 \).
2. By using the same arguments of the previous point, under Assumption 2 we have
\[ C \circ (R_\infty - x_0) = B \circ \mathbb{R}^m. \] (74)
Thus \( x_\infty \in X \) is reachable from \( x_0 \) if and only if
\[ 1^T(Cx_\infty - Cx_0) = 0. \]

3. Under assumption 3 we have
\[ C \circ (R_\infty - x_0) = -\mathbb{R}_{\geq 0}. \] (75)
Thus \( x_\infty \in X \) is reachable from \( x_0 \) if and only if \( Cx_\infty - Cx_0 \leq 0. \)

**Proof of Lemma 1**

**PROOF.** From the definition of \( R^*_\infty(x_0) \) in (20) it is clear \( u^*_t(t) = 0 \) and \( u^*_d(t) = 0 \) if \( x(t) \in R^*_\infty(x_0) \). It remains to prove that \( u^*_t(t) \neq 0 \) or \( u^*_d(t) \neq 0 \) for \( x(t) \not\in R^*_\infty \).

For notational simplicity let \( x = x(t) \), \( u_r = u^*_r(t) \), \( u_d = u^*_d(t) \), \( x^+ = x(t + 1) \), \( \mathcal{U} = \mathcal{U}(t) \), \( R_\infty = R_\infty(x_0) \), and \( R^*_\infty = R^*_\infty(x_0) \). With abuse of notation we write \( J(x, u_r, u_d) = J_{\text{cap}}(x + C^{-1}Bu_r - C^{-1}u_d) \). Note that \( J(x, u_r, u_d) \) is a concave function of \( u_r \) and \( u_d \). This follows from the facts that the minimum of a set of linear functions is concave (i.e., \( \min_{x \in \mathcal{S}} \{ C_1(x^1 - x_2) \} \) is concave), the sum of concave functions is concave, and composition of a concave function with a linear function is concave [14].

1. Let assumption 1 from Section 2.1 hold. We will prove that \( u_d = 0 \) implies that \( u_r \neq 0 \). Conversely by the contra-positive of this statement, this implies \( u_d \neq 0 \) if \( u_r = 0 \). Therefore \( u_r \neq 0 \) or \( u_d \neq 0 \) for \( x \not\in R^*_\infty \).

Suppose \( x \not\in R^*_\infty \) and \( u_d = 0 \). Since \( x \in R_\infty \) and \( R^*_\infty \subseteq R_\infty \) are reachable there exists \( \bar{u}_r \) such that
\[ x^+ = x + C^{-1}B\bar{u}_r \in R^*_\infty. \] (76)
This \( \bar{u}_r \) is not necessarily feasible. However there exists \( \delta_1 > 0 \) such that \( \delta_1 \bar{u}_r \) is feasible since \( U_r \subseteq \text{Proj}_{u_r}(\mathcal{U}) \) is full-dimensional, convex, and contains the origin in its interior. There exists \( \delta_2 > 0 \) such that
\[ \delta_2 [J(x, \bar{u}_r, 0) - J(x, 0, 0)] > \rho_r \delta_2 \| \bar{u}_r \|_2. \] (77)
for all \( x \in R_\infty \setminus R^*_\infty \). Let \( \epsilon = \min \{ \delta_1, \delta_2, 1 \} \). Then \( \epsilon \bar{u}_r \) is feasible and by the the concavity of \( J(x, u_r, u_d) \) it satisfies
\[ J(x, \epsilon\bar{u}_r, 0) - J(x, 0, 0) \geq \epsilon [J(x, \bar{u}_r, 0) - J(x, 0, 0)] > \rho_r \epsilon^2 \| \bar{u}_r \|_2. \] (78)
This is a direct result of the definition of concavity \( J(x, \bar{u}_r, 0) \geq \epsilon J(x, \bar{u}_r, 0) + (1 - \epsilon) J(x, 0, 0) \). Rearranging the terms in (78) we obtain
\[ J(x, \epsilon\bar{u}_r, 0) - \rho_r \| \epsilon\bar{u}_r \|_2 > J(x, 0, 0). \] (79)
The left side of this expression is the cost (37a) of a feasible input \( \epsilon\bar{u}_r \) and the right side is the cost of \( u_r = 0 \). Thus \( u_r = 0 \) is sub-optimal. Therefore \( u_r \neq 0 \) whenever \( u_d = 0 \) and \( x \not\in R^*_\infty \).

2. Under Assumption 2 from Section 2.1 the proof is identical to the previous case since the conditions on \( U_r \subseteq \text{Proj}_{u_r}(\mathcal{U}) \) are identical.

3. Let assumption 3 from Section 2.1 hold. Then \( u_r = 0 \). Since \( x \in R_\infty \) and \( R^*_\infty \subseteq R_\infty \) are reachable there exists \( \bar{u}_d \) such that
\[ x^+ = x + C^{-1}\bar{u}_d \in R^*_\infty. \] (80)
There exist \( \delta_1 > 0 \) such that \( \delta_1 \bar{u}_d \) is feasible since \( U_d \subseteq \text{Proj}_{u_d}(\mathcal{U}) \) is full-dimensional and contains the origin for systems with only dissipative balancing. Using a similar argument as above we can show
\[ J(x, 0, \epsilon\bar{u}_d) - \rho_r \| \epsilon\bar{u}_d \|_2 > J(x, 0, 0) \] (81)
where \( \epsilon = \min \{ \delta_1, \delta \} \) with \( \delta \) chosen as before. Thus \( u_d = 0 \) is sub-optimal. Therefore \( u_d \neq 0 \) whenever \( x \not\in R^*_\infty \).

**Proof of Lemma 2**

**PROOF.** From the definition of \( R_\infty(x_0) \) in (21) it is clear \( u^*_t(t) = 0 \) and \( u^*_d(t) = 0 \) for \( x(t) \in R_\infty \). It remains to prove that \( u^*_t(t) \neq 0 \) or \( u^*_d(t) \neq 0 \) for \( x(t) \not\in R_\infty \).

For notational simplicity let \( x = x(t) \), \( u_r = u^*_r(t) \), \( u_d = u^*_d(t) \), \( x^+ = x(t + 1) \), \( \mathcal{U} = \mathcal{U}(t) \), \( R_\infty = R_\infty(x_0) \), and \( R^*_\infty = R^*_\infty(x_0) \).

1. Let Assumption 1 hold. For \( x \not\in R_\infty \) at least one of the following conditions must occur; some of the temporary storage devices contain more than the minimum resource \( x_i \neq x_j \) for some \( t \in \mathcal{T} \) or some of the permanent storage devices are unbalanced \( x_i \neq x_j \) for some \( i, j \in \mathcal{S} \). We will show that either of these conditions implies \( u_r \neq 0 \) or \( u_d \neq 0 \).

Let us consider the case \( x_i \neq x_j \) for some \( t \in \mathcal{T} \) first. We will show there exists a feasible flow that transfers excess resource from the temporary storage to another device where is can be dissipated. Furthermore this flow and dissipation has a lower cost (47a) than the cost achieved for \( u_r = 0 \) and \( u_d = 0 \).
Let \( \delta_1 = C_t(x_t - x_t) > 0 \) be the excess resource in the temporary storage device \( t \in \mathcal{T} \). Let \( \mathcal{V} \subseteq \mathcal{V} \) denote the set of storage devices that allow non-zero dissipation and \( u_d \) denote the vector of dissipations for these devices. Then the set \( \text{Proj}_{\{u_r, u_d\}}(\mathcal{U}) \) is full-dimensional and contains the origin. Therefore a combination of flow \( u_r \) and dissipation \( u_d \geq 0 \) such that \( \max\{\|u_r\|_\infty, \|u_d\|_\infty\} \leq \delta_2 \) is feasible for some \( \delta_2 > 0 \).

Define \( \epsilon = \min\{\delta_1, \delta_2\} \). Let \( s \in \mathcal{V} \) be the storage device closest to \( t \) that allows dissipation. Consider the flow that moves \( \epsilon \) resource from \( t \) to \( s \) along the shortest path and dissipates it. This flow and dissipation are feasible due to our choice of \( \epsilon \leq \delta_2 \). The state of \( s \in \mathcal{V} \) is unchanged and \( x_t \) is decreased by \( \epsilon/C_t \). Therefore the cost (47a) is decreased by

\[
\frac{\epsilon}{C_t} \left(1 - \frac{\rho_r}{2} C_t \Delta(t, s) - \frac{\rho_d}{2} C_t \right) > 0. \tag{82}
\]

The quantity in (82) is positive due to the assumptions on \( \rho_r \) and \( \rho_d \) since

\[
\frac{\rho_r}{2} C_t \Delta(t, s) \leq \max_{j \in \mathcal{V}} \sum_{i \in \mathcal{V}} \frac{\rho_r}{2} C_i \Delta(i, j) = \frac{\rho_r}{2} \|C \Delta\|_1 < \frac{1}{2}, \tag{83}
\]
and

\[
\frac{\rho_d}{2} C_t \leq \sum_{i \in \mathcal{V}} \frac{\rho_d}{2} C_i = \frac{\rho_d}{2} \text{tr} \! C < \frac{1}{2}. \tag{84}
\]

Thus we have found a flow and dissipation with lower cost (47a) than \( u_r = 0 \) and \( u_d = 0 \). We conclude Algorithm 2 will not produce the sub-optimal input \( u_r = 0 \) and \( u_d = 0 \) when \( x_t \neq x_t \) for some \( t \in \mathcal{T} \).

Now consider the case \( x_t = x_t \) for some \( i, j \notin \mathcal{S} \). We will show there exists a feasible flow that decreases the maximum state of the permanent storage devices by removing resource from the highest storage devices. Furthermore this flow has a lower cost (47a) than the cost for \( u_r = 0 \) and \( u_d = 0 \).

Define the subset of permanent storage devices that have the maximum state

\[
\hat{\mathcal{S}} = \left\{ j \in \mathcal{S} \mid x_j = \max_{i \in \mathcal{S}} x_i \right\}. \tag{85}
\]

and let \( \hat{x} = \max_{i \in \mathcal{S}} x_i \) be the common state of these storage devices. Define \( \hat{\mathcal{G}} \) as the vertex contraction of \( \mathcal{G} \) where vertices \( \mathcal{S} \setminus \hat{\mathcal{S}} \) are identified. We denote this new vertex by \( s \). We assign to vertex \( s \) the maximum state \( x_s = \max_{j \in \mathcal{S} \setminus \hat{\mathcal{S}}} x_j \) and minimum capacity \( C_s = \min_{j \in \mathcal{S} \setminus \hat{\mathcal{S}}} C_j \) of all vertices in \( \mathcal{S} \setminus \hat{\mathcal{S}} \).

Define \( \delta_1 = C_s(\hat{x} - x_s) \) as the amount of resource node \( s \) can accept without joining \( \hat{\mathcal{S}} \). Choose \( \delta_2 \) such that a flow satisfying \( \|u_r\|_\infty \leq \delta_2 \) is feasible. Define

\[
\epsilon = \min \left\{ \frac{\delta_1}{C_s + \sum_{i \in \mathcal{S}} C_i}, \frac{\delta_2}{\sum_{i \in \mathcal{S}} C_i} \right\}. \tag{86}
\]

Consider the flow that transfers \( \epsilon C_i \) resource from each storage device \( i \notin \hat{\mathcal{S}} \) into some device \( j \in \mathcal{S} \setminus \hat{\mathcal{S}} \) along the shortest path. By our choice of \( \epsilon \leq \delta_2/\sum_{i \in \mathcal{S}} C_i \) this flow is feasible since

\[
\|u_r\|_\infty \leq \sum_{j \in \mathcal{S}} \epsilon C_j \leq \delta_2. \tag{87}
\]

This follows from the fact that the shortest path from a vertex \( i \in \hat{\mathcal{S}} \) to some vertex \( j \in \mathcal{S} \setminus \hat{\mathcal{S}} \) will pass through each edge at most once. This flow has cost

\[
\|u_r\|_1 = \sum_{i \in \mathcal{S}} \epsilon C_i \hat{\Delta}(i, s) \tag{88}
\]
where \( \hat{\Delta} \) is the distance matrix of the contracted network \( \hat{\mathcal{G}} \).

This flow will decrease the state of each permanent storage device \( i \in \hat{\mathcal{S}} \) by \( \epsilon \). The state of storage devices \( j \in \mathcal{S} \setminus \hat{\mathcal{S}} \) will only increase. The increase in the state for \( j \in \mathcal{S} \setminus \hat{\mathcal{S}} \) can be upper-bounded

\[
x_j(t + 1) \leq x_j(t) + C_j^{-1} \sum_{i \in \mathcal{S}} \epsilon C_i \leq x_j(t) + C_j^{-1} C_s(\hat{x} - x_s) - \epsilon C_j^{-1} C_s \leq x_j(t) + \hat{x} - \epsilon \leq \hat{x} - \epsilon. \tag{91}
\]

The first inequality is due to the worst-case scenario where all the storage devices \( i \in \mathcal{S} \) transfer resource to a single storage device \( j \in \mathcal{S} \setminus \hat{\mathcal{S}} \). The second inequality is due to our choice of \( \epsilon \) i.e.

\[
\sum_{i \in \mathcal{S}} \epsilon C_i \leq C_s(\hat{x} - x_s) - C_s \delta_1 = C_s(\hat{x} - x_s) - C_s. \tag{90}
\]

The third and fourth inequalities follow from our choice of \( C_s \leq C_j \) and \( x_s \geq x_j \).

Therefore our shortest-path flow will decrease \( \max_{i \in \mathcal{S}} x_i \) by \( \epsilon \). This flow reduces the cost (47a) by

\[
\epsilon \left(1 - \frac{\rho_r}{2} \sum_{i \in \hat{\mathcal{S}}} C_i \hat{\Delta}(i, s) \right) \tag{93}
\]
where the first term is due to \( \max_{i \in \mathcal{S}} x_i \) decreasing by \( \epsilon \) and the second terms is the penalty on the flow. This quantity is positive due to the assumptions on
Let Assumption 2 from Section 2.1 hold. Then

\[ \Delta(i, s) \leq \max_{j \in \mathcal{S}} \Delta(i, j) \]

Note that \( \Delta(i, s) \leq \min_{j \in \mathcal{S}, i} \Delta(i, j) \) since vertex contractions shorten the distances in a network.

Thus we have constructed a feasible flow with a lower cost (47a) than \( u_r = 0 \) and \( u_d = 0 \).

We can conclude \( u_r \neq 0 \) or \( u_d \neq 0 \) whenever \( x \notin R_\infty \).

2. Let Assumption 2 from Section 2.1 hold. Then \( u_d = 0 \).

There are again two cases for \( x \notin R_\infty \) to consider; \( x_i \neq x_j \) for some \( t \in \mathcal{T} \) or \( x_i \neq x_t \) for some \( i, j \in \mathcal{S} \).

In case where \( x_i \neq x_j \) or \( i, j \in \mathcal{S} \) we can use the same argument as above to show \( u_r \neq 0 \). It remains to show that \( u_r \neq 0 \) when \( x_i \neq x_j \) for some \( t \in \mathcal{T} \).

We will show there exists a feasible flow that removes excessive resource from the temporary storage and distributes it evenly among the permanent storage devices. Furthermore this flow has a lower cost (47a) than the flow \( u_r = 0 \).

Let \( \delta_1 = C_t(x_i - x_j) > 0 \) be the excess resource in the temporary storage device \( t \in \mathcal{T} \). There exists \( \delta_2 > 0 \) such that flows satisfying \( \|u_r\|_\infty \leq \delta_2 \) are feasible since \( \text{Proj}_{u_r}(U) \) is full-dimensional, convex, and contains the origin in its interior. Define

\[ \epsilon = \frac{\min\{\delta_1, \delta_2\}}{\sum_{j \in \mathcal{S}} C_j}. \]  \hspace{1cm} (95)

Consider the flow that transfers \( \epsilon C_j \) resource from the temporary storage to each of the permanent storage devices \( j \in \mathcal{S} \) along the shortest path. The permanent storage devices can accommodate this additional resource since \( x_0 \in \mathcal{X}_0 \). By our choice of \( \epsilon \leq \delta_2/\sum_{j \in \mathcal{S}} C_j \) this flow is feasible since

\[ \|u_r\|_\infty \leq \sum_{j \in \mathcal{S}} \epsilon C_j \leq \delta_2. \]  \hspace{1cm} (96)

This follows from the fact that the shortest path from temporary storage device \( t \in \mathcal{T} \) to permanent storage device \( j \in \mathcal{S} \) will pass through each edge at most once. This flow has cost

\[ \|u_r\|_1 = \sum_{j \in \mathcal{S}} \epsilon C_j \Delta(t, j) \]  \hspace{1cm} (97)

where \( \Delta(\cdot, \cdot) \) is the distance matrix of \( G = (\mathcal{V}, \mathcal{E}) \).

This flow will not increase the imbalance since the state-of-charge of each permanent storage device is increased by the same amount \( \epsilon \). Thus this flow decreases the cost (47a) by

\[ \frac{\epsilon}{C_t} \sum_{j \in \mathcal{S}} C_j \left( 1 - \frac{\rho_r}{2} C_t \Delta(t, j) \right). \]  \hspace{1cm} (98)

This quantity is positive due to our choice of \( \rho_r \) since

\[ \frac{\rho_r}{2} \sum_{j \in \mathcal{S}} C_j \Delta(t, j) \leq \frac{\rho_r}{2} \max_{i \in \mathcal{V}} \sum_{i \in \mathcal{V}} C_i \Delta(i, j) \]

\[ = \frac{\rho_r}{2} \|C\Delta\|_1 < \frac{1}{2}. \]  \hspace{1cm} (99)

Thus \( u_r = 0 \) is sub-optimal. We conclude that \( u_r \neq 0 \) whenever \( x \notin R_\infty \).

3. Let Assumption 3 from Section 2.1 hold. Then \( u_r = 0 \) and there are no temporary storage devices. Therefore \( x \notin R_\infty \) implies the storage devices are unbalanced \( x_i \neq x_j \) for some \( i, j \in \mathcal{S} \). Define the subset of permanent storage devices that have the maximum state

\[ \mathcal{S} = \left\{ j \in \mathcal{S} \mid x_j = \max_{i \in \mathcal{S}} x_i \right\}. \]  \hspace{1cm} (100)

Let \( \delta_1 = \max_{i \in \mathcal{S}} x_i - \max_{i \in \mathcal{S}, j \in \mathcal{S}} x_i > 0 \) be the different between the state of the fullest storage devices and the second fullest storage devices. Since \( \text{Proj}_{u_r}(U) \) is full-dimensional, convex, and contains the origin there exists \( \delta_2 > 0 \) such that \( |C^{-1}d|_\infty \leq \delta_2 \) implies \( u_d \geq 0 \) is feasible. Let \( \epsilon = \min\{\delta_1, \delta_2\} \) and define the dissipation \( u_d \)

\[ d_i = \begin{cases} \epsilon C_i & \text{for } i \in \mathcal{S} \\ 0 & \text{for } i \notin \mathcal{S}. \end{cases} \]  \hspace{1cm} (101)

This dissipation is feasible by our choice of \( \epsilon \leq \delta_2 \). It decreases \( \max_{i \in \mathcal{S}} x_i \) by \( \epsilon \) since \( \epsilon \leq \delta_1 \). Therefore the cost (47a) is decreased by

\[ \epsilon \left( 1 - \frac{\rho_d}{2} \sum_{i \in \mathcal{S}} C_i \right) \]  \hspace{1cm} (102)

where the first term is due to \( \max_{i \in \mathcal{S}} x_i \) decreasing by \( \epsilon \) and the second term is the penalty on the dissipation. This quantity is positive due to our choice of \( \rho_d \) since

\[ \frac{\rho_d}{2} \sum_{i \in \mathcal{S}} C_i \leq \frac{\rho_d}{2} \sum_{i \in \mathcal{S}} C_i = \frac{\rho_d}{2} \text{tr}(C) < \frac{1}{2}. \]  \hspace{1cm} (103)

Therefore there exists \( u_d \neq 0 \) that improves the cost (47a). We conclude Algorithm 2 will produce a control input \( u_d \neq 0 \) for \( x \notin R_\infty \).

\[ \blacksquare \]

Proof of Lemma 3
**PROOF.** From the definition of $\tilde{R}_\infty(x_0)$ in (21) it is clear $u_r^*(t) = 0$ and $u_d^*(t) = 0$ for $x(t) \in \mathcal{R}_\infty$. It remains to show $u_r^*(t) \neq 0$ or $u_d^*(t) \neq 0$ for $x(t) \notin \mathcal{R}_\infty$. Since (57a) is differentiable we will show there exists a feasible descent direction that improves the cost. The gradient $g$ of (57a) with respect to $u_r$ and $u_d$ evaluated at $u_r = 0$ and $u_d = 0$ is

$$g = \begin{bmatrix} x(t) - \bar{x}(t) \\ x(t) - \bar{x}(t) \end{bmatrix}.$$  \hfill (104)

Let $\tilde{g} = g / \|g\|_\infty$ be the normalized gradient. Note $\tilde{g} \neq 0$ for $x \notin \mathcal{R}_\infty$.

1. Let Assumption 1 from Section 2.1 hold. Then there exists $\delta > 0$ such that $\|u_r\|_\infty \leq \delta$ is feasible since $\mathcal{U}_r \subseteq \text{Proj}_{u_r}(\mathcal{U})$ is full-dimensional, convex, and contains the origin in its interior. Define the flow $\tilde{u}_r = -\delta \tilde{g}$. This flow is feasible by our choice of $\delta$ and $\tilde{g}$. It has lower cost than $u_r = 0$ since $\tilde{u}_r^T g < 0$. Therefore $u_r = 0$ is sub-optimal.

2. Under Assumption 2 from Section 2.1 the proof is the same as above.

3. Let Assumption 3 from Section 2.1 hold. Then there exists $\delta > 0$ such that $\|u_d\|_\infty \leq \delta$ implies $u_d \geq 0$ is feasible since $\mathcal{U}_d \subseteq \text{Proj}_{u_d}(\mathcal{U})$ is full-dimensional. Define the flow $\tilde{u}_d = -\max\{\delta \tilde{g}, 0\}$ where the maximum is component-wise. This dissipation must have at least one non-zero term since $x \notin \mathcal{R}_\infty$ and therefore $\bar{x}$ is the average of non-identical terms. This dissipation is feasible by our choice of $\delta$ and $\tilde{g}$. It has lower cost than $u_d = 0$ since $\tilde{u}_d^T g < 0$. Therefore $u_d = 0$ is sub-optimal.

\hfill ■
References


