On the optimal input allocation of discrete-event systems with dynamic input sequence

Tri Cahyono, Rully; Jayawardhana, Bayu

Published in:
58th IEEE Conference on Decision and Control

DOI:
10.1109/CDC40024.2019.9029828

On the optimal input allocation of discrete-event systems with dynamic input sequence*

R. T. Cahyono[1,2] and B. Jayawardhana[1]

***Abstract*— We study in this paper an optimal input allocation problem for a class of discrete-event systems with dynamic input sequence (DESDIS). In this case, the input space is defined by a finite sequence whose members will be removed from the sequence in the next event if they are used for the current event control input. Correspondingly, the sequence can be replenished with new members at every discrete-event time. The allocation problem for such systems describes many scheduling and allocation problems in logistics and manufacturing systems and leads to a combinatorial optimization problem. We show that for a linear DESDIS given by a Markov chain and for a particular cost function given by the sum of its state trajectories, the allocation problem is solved by re-ordering the input sequence at any given event time based on the potential contribution of the members in the current sequence to the present state of the system. In particular, the control input can be obtained by the minimization/maximization of the present input sequence only.**

I. INTRODUCTION

Discrete-event systems (DES) are a class of systems where the state variables evolve according to discrete events that take place based on interactions among different (continuous- and/or discrete-)state variables in the systems [6]. A classical example of DES is a queuing system, in which, a new discrete-event is associated to the serving of new customer after the previous one from the previous discrete-event time has been served. We refer interested readers to [5] for an extensive discussion on the modeling and analysis of DES.

For the past few decades, DES framework has been used to model and to control a large class of physical and cyber-physical systems, which includes, the control of logistics systems, internet congestion control, manufacturing systems and many others that can be described by petri nets or finite-state machine/automata. Some examples of these works are discussed in [6], [11], [13] and [14]. Fairly recent applications of DES in transportation and manufacturing systems are presented in [3] and [4] for container terminal operations and in [7] for general transportation and manufacturing systems.

When DES involve discrete-state with discrete input variables, the optimization/control of such DES leads to a combinatorial optimization problem which is NP-hard. One can resort to a standard algorithm for solving combinatorial problems in DES which is the branch and bound (BB) method. As shown in [12], the BB method can converge to the global maxima/minima for some classes of DES optimization problem. Other well-known heuristic methods for solving combinatorial optimization problems with DES are genetic algorithm and particle swarm methods.

Although the BB and other heuristic methods can be used to find a sub-optimal solution to the combinatorial problem for DES, the main drawback lies with the facts that the algorithms are limited only to the case where the problem can be recasted as a static optimization problem [8]. In this case, the static refers to constraining the dynamic problem by some terminal conditions and all possible control input are well-defined or known apriori within the given time interval (up to the terminal time).

This approach may no longer be feasible when the terminal conditions are free with infinite time horizon and when the input set changes dynamically and cannot be known apriori ahead of time. The latter case is commonly found in many DES application, such as transportation, scheduling, and logistics, where the actual incoming and outgoing goods always differ from the transmitted goods manifest and where the actual incoming and outgoing vehicles always differ from the precomputed plan. In [8], a dynamic DES model is developed for train scheduling problem where the frequent changes to the train operations (schedule, obstacle, rail availability) have limited the use of BB and similar algorithms. Instead of using BB, a greedy travel advance strategy is proposed in [8] on the basis of a dynamic DES model, which is able to find the sub-optimal control inputs of the train schedules with a framework similar to line search algorithm. Another related paper is [9] where stabilization problem for a particular DES with dynamic input set is considered. In this case, the events in DES are asynchronous where the states of each sub-system do not necessarily follow the same clock times and an LMI-based controller is proposed to solve such problem.

A similar DES with asynchronous event transition can also be found in our previous works as in [3], [4]. In these works, a model predictive allocation (MPA) method is proposed in conjunction with a pre-conditioning step. In particular, the DES model of container terminal operations is used to compute an optimal input sequence for a finite event horizon where the input sequence is heuristically pre-conditioned for accommodating the combinatorial optimization step. The proposed MPA method follows the same procedure as the model predictive control approach. The efficacy of our pro-

*This work was supported by Excellence Scholarship from Directorate General of Higher Education of Republic of Indonesia, Indonesia Port Corporation (IPC) and IPC Corporate University under project "Analysis and design of automated distributed container terminal operations systems"

[1]Discrete Technology and Production Automation group, Engineering and Technology Institute of Groningen, Faculty of Science and Engineering, Rijksuniversiteit Groningen, Nijenborgh 4 9747 AG, Groningen, the Netherlands {b.jayawardhana, r.tri.cahyono}@rug.nl

[2]Department of Industrial Engineering, Faculty of Industrial Technology, Institut Teknologi Bandung, Jalan Ganesha 10, Bandung, 40132, Indonesia rully@ti.itb.ac.id

posed method has been shown in both simulation as well as in real-life experiment. In this method, we have used the well-known first-come first-serve (FCFS) or the heavy-first light-last (HFLL) pre-conditioning step to the current input sequence and then truncate it, prior to computing the optimal solution in the model predictive step. While the re-ordering of the sequence (either using FCFS or HFLL) has played an important role in [3], [4], the mathematical analysis on the re-ordering of the input sequence in the pre-conditioning step is still missing.

In this paper, we present an analysis to the re-ordering of the input sequence to the combinatorial optimization problem. The systems' descriptions of our discrete-event time systems with dynamic input sequence (DESDIS) and the associated combinatorial optimization problem are presented in Section II. In Section III, we present the problem of combinatorial optimization where the input sequence changes dynamically with possible new additional sequence and we propose a simple control law that solves the optimization problem. In particular, we show that the resulting control law is based on a particular re-ordering of the input sequence. Finally, conclusions and future works are presented in Section IV.

II. Preliminaries and Optimal Input Allocation Problem

Notations. We denote the vector of all ones by $\mathbb{1}$. A matrix $A \in \mathbb{R}^{n\times n}$ is called a stochastic matrix if $A_{ij} \geq 0$ for all i, j and $\sum_j A_{ij} = 1$ for all i where A_{ij} is the (i, j) element of A. Let us consider an undirected graph $\mathcal{G}$ given by the (V, E) where V is the set of vertices and $E \subset V \times V$ is the set of edges. Such graph $\mathcal{G}$ can be represented by the stochastic matrix A where the element A_{ij} shows the communication weight from the j-th vertex to the i-th vertex. The graph $\mathcal{G}$ is called connected if for every pair vertices there is a path on the graph that connects these vertices. Equivalently, it is connected if the kernel of A has rank one and is given by $\mathbb{1}$.

Let us consider the following generic model of discrete event system Σ whose input is taken from a time-varying sequence:

$$\Sigma : x(k + 1) = Ax(k) + Bf(x(k), u(k)), \tag{1}$$

where $x(k) \in \mathbb{R}^n_+$ denotes the state variables (such as, the berth starting time, berth operations time, berth finishing time, etc.), k is the discrete-event time variable, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ is assumed to be *non-negative matrix*, i.e., $a_{ij} \geq 0$ and $b_{ij} \geq 0$. This system description is relevant for describing the dynamic behaviour of planning and scheduling in logistics systems, particulary in the container terminal operations, which will be shown later in our example.

We remark that if we consider the case where $f = 0$ then our assumption on A implies that (1) describes a positive discrete-event system. In this case, $x(0) > 0$ (where the inequality is interpreted element-wise) implies that $x(k) > 0$ for all $k > 0$. Each element in the function $f : \mathbb{R}^n_+ \times \mathbb{R}^m_+ \to \mathbb{R}^m_+$ is assumed to be positive definite function, i.e., $f_i(x, u) > 0$ for all $(x, u) \neq (0, 0)$ and for all i.

The input variable $u(k) \in \mathbb{R}^m$ is the decision/input variable that is taken from a (possibly, infinite) sequence $\mathcal{U}_k = \{\upsilon_i\}_{i\in\{1,2,...,N\}}$. We further assume that the evolution of $u(k)$, $x(k)$ and $\mathcal{U}_k$ follow the following rule.

Discrete-Event Systems with Dynamic Input Sequence (DESDIS):

- **(A1).** (*Initialization.*) Let the initial sequence $\mathcal{U}_0$ be given by $\mathcal{U}_0 := \{\upsilon_i\}_{i\in\{1,2,...N\}}$ with $\upsilon_i \in \mathbb{R}^m_+$ for all $i = 1, \dots N$ with N be the dimension of initial sequence and let $k = 0$.
- **(A2).** (*Decision making step and state evolution.*) If $\mathcal{U}_k \neq \emptyset$ then let the input $u(k)$ be given by an element from $\mathcal{U}_k$, i.e., $u(k) \in \mathcal{U}_k$ (which can be based on a particular input allocation/control law which will be discussed later) and the state is updated according to (1). Otherwise, $u(k) = 0$.
- **(A3).** (*Update of the input sequence.*) If $\mathcal{U}_k \neq \emptyset$ then the decision sequence is updated according to $\mathcal{U}_{k+1} = \mathcal{U}_k \backslash u(k)^\frown V_k$ where $\backslash$ is the element removal operation from the sequence, $^\frown$ is a concatenation operator of two sequences and V_k is the possible new additional sequence at step k. In other words
$$\mathcal{U}_{k+1} = \{\mathcal{U}_k \backslash u(k), V_k\}.$$
Otherwise $\mathcal{U}_{k+1} = V_k$.
- **(A4).** (*Update of the event time.*) Let $k = k+1$ and return to **(A2)**.

Note that for the particular case of $V_k = u(k)$ (i.e., $\mathcal{U}_{k+1} = \mathcal{U}_k$ or the element that is taken from $\mathcal{U}(k)$ is directly replenished by an identical element $\{u(k)\}$), we have the usual description of linear discrete-time systems with atomic input set $\mathcal{U}$. For the control of such systems with a fixed atomic input set $\mathcal{U}$, we can refer interested readers to the literature on discrete-event control systems, on hybrid control systems and on finite-state automata.

When A is a stochastic matrix and $f = 0$, it is well-known that the state trajectories of the autonomous system will reach consensus (see, for instance, [2], [10]).

Lemma 1: Consider the system (1) with $f = 0$ and A be stochastic matrix. Then for every $x(0) \in \mathbb{R}^n_+$, there exists $x^* \in \mathbb{R}^n_+$ such that $\lim_{k\to\infty} x(k) = x^*$, i.e., the state trajectory x converges to a positive constant vector. Moreover if the graph associated to A is strongly connected then $x^* = a\mathbb{1}$ for some constant $a > 0$.

Example 1: As a simple example to the system described in (1), we can consider the following system

$$x(k + 1) = x(k) + F(u(k)) \tag{2}$$

where $x(k), u(k)$ are scalar and the function F is a positive definite function. This simple dynamics may represent many logistics systems where the state x represents a particular operations time. For example, a simple berthing process [3] can be described by (2) where $x(k)$ defines the berthing time for the k-th event and F defines the operational time of loading/unloading the ship whose size is associated to $u(k)$. In this case, each element in the sequence $\mathcal{U}_k$ represents the set of ship's size that are waiting to be berthed. When a ship $u(k)$ has been assigned for berthing, then this ship will be no longer in the set $\mathcal{U}_k$ and at the same event, a new set of arriving ships V_k can call to the port for berthing. After the berthing process of the ship $u(k)$ is completed, the ship will continue its journey to its next destination and therefore it will also not appear in the next input sequence $\mathcal{U}_{k+1}$ and V_k will be added to this new sequence.

Other example that can be described by (2) is the discrete production process where $x(k)$ describes the finishing production time for the k-th event, $u(k)$ represents the set of number of products to be produced in the line and F is the operational production time which is a (nonlinear) function of the product order.

For a higher-order example, we can consider Markov chain describing Markov processes where A is a stochastic matrix. $\triangle$

We remark that the assumption on A and the positive definiteness of f ensures that $\|x(\ell)\| > \|x(m)\|$ for all $\ell > m$ when $u = 0$. This property is shown in the following lemma.

Lemma 2: Consider the system (1) as above with $u = 0$ and positive definite function f satisfying $\|f(\xi, 0)\|^2 \geq L\|\xi\|^2$ where $L > 0$ satisfies

$$L > \frac{1 - \lambda_{\min}(A^T A)}{\lambda_{\min}(B^T B)} \tag{3}$$

with $\lambda_{\min}$ denotes the minimum eigenvalue. Then $\|x(\ell)\| > \|x(m)\|$ for all $\ell > m$.

Proof: By taking $\ell = m + 1$, we have that

$$\begin{aligned}
&\|x(\ell)\|^2 = \|Ax(m) + Bf(x(m), 0)\|^2 \\
&= x^T(m)A^T Ax(m) + 2x^T(m)A^T Bf(x(m), 0) \\
&\quad + f^T(x(m), 0)B^T Bf(x(m), 0) \\
&\geq \lambda_{\min}(A^T A)\|x(m)\|^2 + \lambda_{\min}(B^T B)\|f(x(m), 0)\|^2 \\
&> \|x(m)\|^2
\end{aligned}$$

where the first inequality is due to the non-negativity of $x(m)$, A and B, and last inequality is due to the bound on $\|f(\xi, 0)\|^2$ in (3). ■

The above results show that under the hypothesis of Lemma 2, we have the following ordering of the norm of state trajectories.

$$\|x(0)\| < \|x(1)\| < \ldots < \|x(k)\| < \|x(k+1)\| < \ldots$$

Furthermore, if we restrict to the classes of system where the diagonal elements of A are greater than or equal to 1 then we have ordering of every element of the state trajectories.

Lemma 3: Suppose that (3) holds and $a_{ii} \geq 1$ for all i. Then we have

$$x_i(0) < x_i(1) < \ldots < x_i(k) < x_i(k+1) < \ldots$$

for all $i = 1, \ldots n$.

Proof: The proof of the lemma follows immediately from the dynamics of each state variable x_i that is given by

$$x_i(k+1) = a_{ii}x_i(k) + \sum_{j \neq i} a_{ij}x_j(k) + b_i f(x(k), u(k)).$$

Since $x_j(k) > 0$ for all $j \neq i$ and f is a positive definite function, the claim follows trivially. ■

In the above results, we have shown nice ordering properties in terms of the state trajectories of (1). In particular, when the system is driven by integrators (such as the one in Example 1) each state trajectory is a monotonically increasing signal.

Let us now introduce how the decision process in the input allocation can influence the state dynamical behaviour by using the simple example as in Example 1.

Example 2: Consider the simple system as in (2) with $F(s) = s$. For simplicity, let us consider $\mathcal{U}_0 = \{1, 5, 4\}$ and the state is initialized at the origin, i.e., $x(0) = 0$. Since the cardinality of $\mathcal{U}_0$ is 3 ($\dim(\mathcal{U}_0) = 3$) and we do not consider replenishment to the input sequence (e.g., $V_k = \emptyset$ for all k), then the state evolution following the **DESDIS** rule reaches steady state in a finite-time.

First-Come First-Serve (FCFS) rule. If the input is assigned according to the first-come first-serve rule (where the order in the sequence $\mathcal{U}_0$ determines the assignment timing, i.e., the first element is used first and the last element will be applied lastly) then the input is given by $u(0) = 1, u(1) = 5, u(2) = 4$. In this case, the trajectory of x is given by

$$\begin{aligned}
x(0) &= 0 \\
x(1) &= x(0) + u(0) = 1 \\
x(2) &= x(1) + u(1) = 6 \\
x(3) &= x(2) + u(2) = 10 \\
x(4) &= x(3) \\
&\vdots \\
x(k+1) &= x(k)
\end{aligned}$$

and the evolution of the input sequence is

$$\begin{aligned}
\mathcal{U}_1 &= \mathcal{U}_0 \backslash u(0)^\frown \emptyset = \{5, 4\} \\
\mathcal{U}_2 &= \mathcal{U}_1 \backslash u(1)^\frown \emptyset = \{4\} \\
\mathcal{U}_3 &= \mathcal{U}_2 \backslash u(2)^\frown \emptyset = \emptyset \\
\mathcal{U}_k &= \emptyset \qquad\qquad \forall k > 3.
\end{aligned}$$

Reordering the input sequence. Instead of assigning the input using FCFS rule as above, we can also set the input $u(k)$ according to a certain control law/allocation mechanism based on the available elements in the current input sequence $\mathcal{U}_k$. One particular instance for this is by firstly rearranging the input sequence $\mathcal{U}_0$ in the *ascending order* and then apply the FCFS rule. Note that this control law is equivalent to taking the minimum over $\mathcal{U}_k$, i.e., $u(k) = \min\{\mathcal{U}_k\}$. For the above example, we can rearrange $\mathcal{U}_0$ into $\{1, 4, 5\}$ and then we apply the FCFS rule. This gives us $u(0) = 1, u(1) = 4, u(2) = 5, u(k) = 0$ for all $k > 2$ and subsequently, $x(0) = 0, x(1) = 1, x(2) = 5, x(3) = 10 = x(k)$ for all $k > 3$. The input sequence evolution is given by $\mathcal{U}_1 = \{4, 5\}, \mathcal{U}_2 = \{5\}, \mathcal{U}_3 = \emptyset = \mathcal{U}_k$ for all $k > 3$. We remark that since the cardinality of $\mathcal{U}_0$ is 3 then there are 6 combinations for the reordering of the input sequence $\mathcal{U}_0$. In the rest of the paper, we will discuss such allocation/control problem for finding an optimal reordering of the input sequence that minimizes a given cost function. △

In Example 2, two possible state trajectories have been shown based on two different ways the input is allocated from the initial input sequence. As remarked in Example 2, there are in total six possible state trajectories where all of them will reach the same steady-state value of 10. In fact, the property of steady state value that is invariant to all possible combination of input allocation can be extended to the system (1), where f is only a function of u, as given in the following proposition.

Proposition 1: Consider the system (1) with $f : (x, u) \mapsto F(u)$ where F is a function of u and A is a doubly stochastic matrix. If $V_k = \emptyset$ for all k then for all initial input sequence $\mathcal{U}_0$ with $\text{card}\{U_0\} =: N_0$, the systems with dynamic input sequence described by **(A1)-(A4)** satisfies

$$\mathbb{1}^T x(k) = \sum_{n=0}^{N_0-1} \mathbb{1}^T BF(u(n)) \qquad \forall k \geq N_0,$$

i.e., the sum of the state $x(k)$ is constant for all $k \geq N_0$.

Proof: As the input sequence is not replenished for every input allocation, it follows from the **DESDIS** rule **(A3)** that $\mathcal{U}_k = \emptyset$ for all $k \geq N_0$. This means that for all $k \geq N_0$, $u(k) = 0$ and therefore,

$$\begin{aligned}
\mathbb{1}^T x(k+1) &= \mathbb{1}^T Ax(k) = \mathbb{1}^T x(k) \qquad \forall k \geq N_0 \\
&= \mathbb{1}^T Ax(N_0 - 1) + \mathbb{1}^T BF(u(N_0 - 1)) \\
&= \mathbb{1}^T x(N_0 - 1) + \mathbb{1}^T BF(u(N_0 - 1)) \\
&= \mathbb{1}^T Ax(N_0 - 2) + \sum_{n=N_0-1}^{N_0-2} \mathbb{1}^T BF(u(n)) \\
&\ \ \vdots \\
&= \sum_{n=0}^{N_0-1} \mathbb{1}^T BF(u(n)).
\end{aligned}$$

This equality holds for arbitrary choice of input allocation $u(k), k = 0, \ldots, N_0 - 1$ from $\mathcal{U}_0$. ■

When we furnish the current input sequence $\mathcal{U}_k$ with an additional new sequence V_k at every time step k (as in the step **(A3)** of the **DESDIS** update rule), the number of possible state trajectories can increase dramatically.

For the system in (1), suppose that we can define a cost function $J(x, u)$ that must be minimized by optimally allocating the input $u(k)$ from the available finite sequence $\mathcal{U}_k$. Using such J, we can define our optimal allocation/control problem as follows.

Optimal input allocation problem: For a given discrete-event system (1) with dynamic input sequence satisfying **(A1)-(A4)**, with given expansion input sequences V_k, $k \in \mathbb{N}$ and cost function $J(x, u) : l^\infty(\mathbb{N}, \mathbb{R}^n) \times l^\infty(\mathbb{N}, \mathbb{R}^m) \to \mathbb{R}_+$, determine the optimal input allocation $u^*(k) \in \mathcal{U}_k$ for all k such that

$$u^* = \operatorname*{argmin}_u J(x, u)$$

where the state trajectory x satisfies (1) and the **DESDIS** rule in **(A1)-(A4)**.

In the following proposition, it is shown that for the particular case when $V_k = \emptyset$ for all k, the solution to the above optimal input allocation problem corresponds to a particular ordering of $\mathcal{U}_0$ that depends on B and the nonlinear function F.

Proposition 2: Consider the system as in Proposition 1 and suppose that the cost function is given by

$$J(k) = \sum_{n=0}^{k} \mathbb{1}^T x(n). \tag{4}$$

Then the optimal input allocation u^* satisfies

$$\mathbb{1}^T BF(u^*(0)) \leq \mathbb{1}^T BF(u^*(1)) \leq \cdots \leq \mathbb{1}^T BF(u^*(N_0)) \tag{5}$$

where $N_0 := \text{card}(\mathcal{U}_0)$ and $u^*(k) = 0$ elsewhere.

Proof: Following the computation in the proof of Proposition 1, we can expand (4) as follows.

$$\begin{aligned}
J(k) &= \mathbb{1}^T x(k) + \mathbb{1}^T x(k-1) + \cdots + \mathbb{1}^T x(0) \\
&= \sum_{n=0}^{k-1} \mathbb{1}^T BF(u(n)) + \sum_{n=0}^{k-2} \mathbb{1}^T BF(u(n)) + \cdots \\
&\qquad + \mathbb{1}^T BF(u(0)) + k\mathbb{1}^T x(0) \\
&= k\mathbb{1}^T BF(u(0)) + (k-1)\mathbb{1}^T BF(u(1)) + \cdots \\
&\qquad + \mathbb{1}^T BF(u(k)) + k\mathbb{1}^T x(0).
\end{aligned}$$

It follows immediately from this equality and since $\mathcal{U}_0$ is finite that the minimum of J is reached if and only if (5) holds which is independent of the initial condition $x(0)$. ■

We want to recall the readers that as the **DESDIS** fulfills **(A1) - (A4)**, the optimal input sequence satisfies

$$(u^*(0), u^*(1), \ldots, u^*(N_0)) \in \mathrm{P}_{N_0}(\mathcal{U}_0)$$

where $\mathrm{P}_{N_0}(\mathcal{U}_0)$ denotes the set of N_0-permutations of $\mathcal{U}_0$.

As defined in (4), the cost function in the above proposition is given by the sum of all state values at all time. Thus the minimization of the cost function means that at any given time the state values must be made as small as possible by allocating proper input sequence. If we refer to Example 1, this cost function can be interpreted as the sum of all berthing time and the minimization of this function implies that the optimal input allocation will ensure that the berthing time is kept as small as possible. Consequently the berth position will be made vacant as earlier as possible.

We have shown in Proposition 2 that in the absence of input sequence expansion V_k, the solution to the optimal input allocation problem is given by ordering the sequence $(u^*(0), u^*(1), \ldots, u^*(N_0))$ such that (5) holds. In particular, if $x(k)$ is a scalar, i.e., $B = 1$, and F is positive definite and non-decreasing then (5) becomes

$$u^*(0) \leq u^*(1) \leq \cdots \leq u^*(N_0),$$

i.e., the scalar input $u^*(k)$ is ordered from the lowest to the largest. On the other hand, if F is non-increasing then we have that $u^*(0) \geq u^*(1) \geq \cdots \geq u^*(N_0)$. Such an ordering property in the optimal input sequence is closely related to the reordering input sequence approach as discussed in Example 2. In this example, instead of allocating the input according to the sequence $\mathcal{U}_0$ (i.e., $u(0) = 1, u(1) = 5$ and $u(2) = 4$), we can firstly reorder $\mathcal{U}_0$ and then assign the input according to the re-ordered sequence.

In general, such ordering property in the optimal input sequence is not true for arbitrary cost functions J. This can again be exemplified by Example 2. In this example, if the cost function is given by $J(3) = \sum_{n=0}^{3} x(n)$ then the optimal control input is $u(0) = 1, u(1) = 4, u(2) = 5, u(k) = 0$ for all $k > 2$. However, when the cost function is modified into $J(3) = \sum_{n=0}^{3} x(n) + \sum_{n=0}^{3} 2u(n)n$, where we penalize the allocation of high control input at a later time, the optimal control input becomes $u(0) = 5, u(1) = 4, u(2) = 1, u(k) = 0$ for all $k > 2$. In this case the order of optimal input allocation is reversed despite the fact that F is an identity (contrary to what has been described before on non-decreasing F).

In the following section, we consider the aforementioned optimal input allocation problem when $V_k \neq 0$.

III. Optimal Input Allocation with $V_k \neq 0$

Previously, we have seen that when the input sequence is not dynamically expanded by V_k as defined in step **(A3)**, the optimal input allocation is based on a particular order of the initial input sequence $\mathcal{U}_0$. In fact, it can be described by

$$u(k) = \operatorname*{argmin}_{\xi \in \mathcal{U}_k} \mathbb{1}^T BF(\xi)$$

where $\mathcal{U}_k$ is always updated according to **(A3)**. We will now study whether the above control law solves also the optimal input allocation problem for the case of $V_k \neq 0$.

Proposition 3: Consider the system (1) with $f : (x, u) \mapsto F(u)$ where F is a function of u, A is a doubly stochastic matrix and the system satisfies **(A1)-(A4)**. Let V_k, $k \in \mathbb{N}$, be the expansion sequences for the input sequence $\mathcal{U}_k$ that is updated according to **(A3)**. Suppose that the cost function is given by (4). Then for all initial input sequence $\mathcal{U}_0$, the control law defined by

$$u(k) = \operatorname*{argmin}_{\xi \in \mathcal{U}_k} \mathbb{1}^T BF(\xi) \qquad \forall \text{ event time } k \in \mathbb{N} \tag{6}$$

solves the optimal input allocation problem.

Proof: We prove the proposition by induction. When we want to minimize $J(1)$, it is straightforward to see that the optimal input $u^*(0)$ is given by (6) with $k = 0$. When $k = 2$, we need to show that the optimal cost function $J^*(2)$ is obtained by taking

$$u(0) = \operatorname*{argmin}_{\xi \in \mathcal{U}_0} \mathbb{1}^T BF(\xi)$$

and

$$u(1) = \operatorname*{argmin}_{\xi \in \mathcal{U}_0 \setminus u(0)^\frown V_0} \mathbb{1}^T BF(\xi),$$

where we have applied **(A3)** to get $\mathcal{U}_1 = \mathcal{U}_0 \setminus u(0)^\frown V_0$. Indeed, by direct computation, we have that

$$\begin{aligned} J^*(2) &= \min_u J(k) \\ &= \min_{\substack{u(0) \in \mathcal{U}_0 \\ u(1) \in \mathcal{U}_1}} \left(2\mathbb{1}^T BF(u(0)) + \mathbb{1}^T BF(u(1))\right) + 2\mathbb{1}^T x(0). \end{aligned} \tag{7}$$

It remains to show that the above equality can be written as

$$\begin{aligned} &= \min_{u(0) \in \mathcal{U}_0} 2\mathbb{1}^T BF(u(0)) + \min_{u(1) \in \mathcal{U}_1} \mathbb{1}^T BF(u(1)) \\ &\quad + 2\mathbb{1}^T x(0), \end{aligned}$$

in which case, the claim of the proposition for the minimization of $J^*(2)$ holds. We prove it by contradiction. Suppose that there exists $u(0) \in \mathcal{U}_0$ and $u(1) \in \mathcal{U}_1$ such that (7) holds while the above equality is not satisfied. In this case, we have two cases: (i). both $u(0), u(1) \in \mathcal{U}_0$; or (ii). $u(0) \in \mathcal{U}_0$ and $u(1) \in V_0$. For the first case, we arrive at a similar situation as in Proposition 2 which also implies that the above equality holds, a contradiction. On the other hand, for the second case, it follows trivially that the above equality holds; again a contradiction.

For the last part of proof by induction, we will now show that given the optimal input allocation $U_k^* := (u^*(0), u^*(1), \ldots, u^*(k))$, which is calculated recursively as in (6) and minimizes $J(k + 1)$, the minimizer of $J(k + 2)$ is given by $(U_k^*, u^*(k + 1))$ where $u^*(k + 1)$ is computed as in (6). Similar as before, we have that

$$\begin{aligned} &J^*(k + 2) \\ &= \min_{\substack{u(0) \in \mathcal{U}_0 \\ \vdots \\ u(k+1) \in \mathcal{U}_{k+1}}} \Big((k + 1)\mathbb{1}^T BF(u(0)) + k\mathbb{1}^T BF(u(1)) + \cdots \\ &\qquad \cdots + 2\mathbb{1}^T BF(u(k)) + \mathbb{1}^T BF(u(k + 1))\Big) \\ &\qquad + (k + 1)\mathbb{1}^T x(0). \end{aligned}$$

Using the same arguments as before, we can prove by contradiction that the above equality is equivalent to

$$
\begin{aligned}
&J^*(k+2) \\
&= \min_{\substack{u(0)\in\mathcal{U}_0 \\ \vdots \\ u(k)\in\mathcal{U}_k}} \Big((k+1)\mathbb{1}^T BF(u(0)) + k\mathbb{1}^T BF(u(1)) + \cdots \\
&\qquad \cdots + 3\mathbb{1}^T BF(u(k-1)) + 2\mathbb{1}^T BF(u(k)) \Big) \\
&\qquad + \min_{u(k+1)\in\mathcal{U}_{k+1}} \mathbb{1}^T BF(u(k+1)) + (k+1)\mathbb{1}^T x(0).
\end{aligned}
$$

The solution to the first term of the above equality is the same as the minimization of $J(k+1)$, which is U_k^*. For the second term, it is the solution $u^*(k+1)$ to (6) for $k+1$. Therefore, we have that $U_{k+1}^* = \big(U_k^*, u^*(k+1)\big)$ as claimed. ∎

In this proposition, we have shown that the optimal input allocation can be computed recursively. In particular, at each event time k, $u^*(k)$ is obtained based only on the current input sequence $\mathcal{U}_k$ and is independent on the possible expansion sequence in any future event time. In this regards, the optimization of $J(k+1)$ requires only $\text{card}(\mathcal{U}_k) =: N_k$ operations instead of $N_k!$ (or even larger when we take into account all possible permutation with the inclusion of future V_n, $n > k$).

Example 3: Let us consider again the system as in Example 2. Suppose that $V_0 = \{3, 2\}$, $V_1 = \{1, 3, 5\}$ and $V_k = \emptyset$ for all $k > 2$. Using the input allocation law as in Proposition 3, we can recursively compute the optimal input allocation $u(k)$ as follows:

$$
\begin{aligned}
u(0) &= \operatorname*{argmin}_{\xi\in\{1,5,4\}} \xi = 1, & \mathcal{U}_1 &= \{5,4\} \cup \{3,2\} \\
u(1) &= \operatorname*{argmin}_{\xi\in\{5,4,3,2\}} \xi = 2, & \mathcal{U}_2 &= \{5,4,3\} \cup \{1,3,5\} \\
u(2) &= \operatorname*{argmin}_{\xi\in\{5,4,3,1,3,5\}} \xi = 1, & \mathcal{U}_3 &= \{5,4,3,3,5\} \\
u(3) &= \operatorname*{argmin}_{\xi\in\{5,4,3,3,5\}} \xi = 3, & \mathcal{U}_4 &= \{5,4,3,5\} \\
u(4) &= \operatorname*{argmin}_{\xi\in\{5,4,3,5\}} \xi = 3, & \mathcal{U}_5 &= \{5,4,5\} \\
u(5) &= \operatorname*{argmin}_{\xi\in\{5,4,5\}} \xi = 4, & \mathcal{U}_6 &= \{5,5\} \\
u(6) &= \operatorname*{argmin}_{\xi\in\{5,5\}} \xi = 5, & \mathcal{U}_7 &= \{5\} \\
u(7) &= 5 \qquad \text{and} \qquad u(k) = 0 & \forall k &> 7.
\end{aligned}
$$

Thus, the maximum number of operations is on the computation of $u(2)$. We can compare this with the exhaustive search of $(u^*(0), u^*(1), u^*(2), \ldots, u^*(7))$ where we evaluate all permutation of the combined input sequence $(\mathcal{U}_0, V_0, V_1, V_2) = \{1, 5, 4, 3, 2, 1, 3, 5\}$ and evaluate the state evolution of **DESDIS** for each of possible permutation of input sequence according to **(A1)–(A4)**. △

IV. Conclusion

We have shown that for a particular case of allocation problem in DESDIS, re-ordering of the input sequence at every discrete-event time is needed for determining the optimal sequence. The analysis provides a good basis for the development of model predictive allocation methods for DESDIS as pursued, for instance, in [3], [4]. Further works are needed on the re-ordering of input sequence $\mathcal{U}_k$ (and the subsequent expansion sequences V_k for a finite event horizon) when a general cost function is considered.

References

[1] A. Alessandri, M. Gaggero, and F. Tonelli, "Min-max and predictive control for the management of distribution in supply chains," IEEE Trans. Contr. Syst. Techn., vol. 19, no. 54, pp. 1075–1089, 2011.

[2] V.D. Blondel, J.M. Hendrickx, A. Olshevsky, J.N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," IEEE Proc. 44th Conf. Dec. Contr. & Eur. Contr. Conf., Sevilla, 2005.

[3] R.T. Cahyono, E.J. Flonk, B. Jayawardhana, "Dynamic berth and quay crane allocation for multiple berth positions and quay cranes," Proc. 2015 European Control Conference, Linz, Austria, 2015.

[4] R.T. Cahyono, E.J. Flonk, B. Jayawardhana, "Discrete-event systems modeling and model predictive allocation algorithm for integrated berth and quay crane allocation," submitted to IEEE Trans. Intell. Trans. Syst., 2019.

[5] C.G. Cassandras, S. Lafortune, *Introduction to Discrete Event Systems,* 2nd edition, Springer, New York, 2008.

[6] G. Cohen, D. Dubois, J.P. Quadrat, M. Viot, "A linear-system-theoretic view of discrete-event processes and its use for performance evaluation in manufacturing," IEEE Trans. Aut. Contr., vol. AC-30, no. 3, pp. 210-220, 1985.

[7] B. De Schutter, T. v.d. Boom, "MPC for discrete-event systems with soft and hard synchronization constraints," Int. J. Con., vol. 76, no. 1, pp. 82-94, 2007.

[8] M.D. Dorfman, J. Medanic, "Scheduling trains on a railway network using a discrete event model of railway traffic," Trans. Res. Part B, vol. 38, pp. 81-98, 2004.

[9] A. Hassibi, S.P. Boyd, J.P. How, "Control of asynchronous dynamical system with rate constraints on events," Proc. of the 38th Conf. on Dec. and Contr., pp. 1345-1351, 1999.

[10] J.M. Hendrickx, G. Shi, K.H. Johansson, "Finite-time consensus using stochastic matrices with positive diagonals," IEEE Trans. Aut. Contr., vol. 60, no. 4, pp. 1070-1073, 2015.

[11] G.C. Goodwin, P.J.G. Ramadge, P.E. Caines, "Discrete-time multivariable adaptive control," IEEE Trans. Aut. Contr., vol. AC-25, no. 3, pp. 449-456, 1980.

[12] V.I. Norkin, G.C. Pflug, A. Ruszczyński, "Discrete-time multivariable adaptive control," IEEE Trans. Aut. Contr., vol. AC-25, no. 3, pp. 449-456, 1980.

[13] K.M. Passino, P.J. Antsaklis, "On the optimal control of discrete event system," Proc. of the 29th Conf. on Dec. and Contr., pp. 2713-2718, 1989.

[14] P.J.G. Ramadge, W.M. Wonham, "The control of discrete event system," Proc. of the IEEE, vol. 77, no. 1, pp. 81-98, 1989.