**Delft University of Technology**

**Delft Center for Systems and Control**

Technical report 10-020

# Predictive route control for automated baggage handling systems using mixed-integer linear programming*

A.N. Tarău, B. De Schutter, and J. Hellendoorn

*If you want to cite this report, please use the following reference instead:*
A.N. Tarău, B. De Schutter, and J. Hellendoorn, "Predictive route control for automated baggage handling systems using mixed-integer linear programming," *Transportation Research Part C*, vol. 19, no. 3, pp. 424–439, June 2011.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.51.19 (secretary)
fax: +31-15-278.66.79
URL: `http://www.dcsc.tudelft.nl`

*This report can also be downloaded via `http://pub.deschutter.info/abs/10_020.html`

# Predictive route control for automated baggage handling systems using mixed-integer linear programming

A.N. Tarău[a], B. De Schutter[a], J. Hellendoorn[a]

[a]*Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands {a.n.tarau,b.deschutter,j.hellendoorn}@tudelft.nl*

**Abstract**

State-of-the-art baggage handling systems transport luggage in an automated way using destination coded vehicles (DCVs). These vehicles transport the bags at high speeds on a network of tracks. In this paper we consider the problem of controlling the route of each DCV in the system. In general this results in a nonlinear, nonconvex, mixed-integer optimization problem, usually very expensive in terms of computational effort. Therefore, we present an alternative approach for reducing the complexity of the computations by simplifying and approximating the nonlinear optimization problem by a mixed-integer linear programming (MILP) problem. The advantage is that for MILP problems solvers are available that allow us to efficiently compute the global optimal solution. The solution of the MILP problem can then be used as a good initial starting point for the original nonlinear optimization problem. We use model predictive control (MPC) for solving the route choice problem. To assess the performance of the proposed (nonlinear and MILP) formulations of the MPC optimization problem, we consider a benchmark case study, the results being compared for several scenarios.

*Keywords:* Baggage handling systems, route choice control, mixed-integer linear programming.

*Preprint submitted to Transportation Research Part C*

## 1. Introduction

Modern baggage handling systems in airports transport luggage at high speeds using destination coded vehicles (DCVs) that transport the bags in an automated way on a network of tracks.

The DCV-based baggage handling system has two levels of control. The low-level controllers ensure the coordination and synchronization when loading a bag onto a DCV, in order to avoid damaging the bags or blocking the system, and when unloading it to the corresponding end point. Low-level controllers also compute the velocity of the DCVs such that collisions are avoided. These low-level controllers are typically Proportional Integral Derivative (PID) controllers and logic controllers that can stop the DCV when necessary. Higher-level controllers compute the route assignment for each bag in the network. Currently, the DCVs are routed through the system using routing schemes based on preferred routes. These routing schemes can be adapted to respond to the occurrence of predefined events. However, as argued in de Neufville (1994), the patterns of loads on the system are highly variable, depending on e.g. the season, time of the day, type of aircraft at each gate, number of passengers for each flight. Therefore, we do not consider predefined preferred routes. Instead we develop advanced control methods to determine the optimal routing in case of dynamic demand.

For applications such as automated guided vehicles route planning or traffic route guidance, the route assignment problem has been addressed in e.g. Gang et al. (1996); Kaufman et al. (1998). The optimal AGV or traffic guidance routes are the shortest-path or the shortest-time routes. Since we want the bags to arrive at their end point within given time windows, the problem of routing DCVs cannot be solved using the methods for routing of AGVs or traffic flows. The problem of routing the DCVs of a baggage handling system is presented in Fay (2005) and Hallenborg and Demazeau (2006). The solution proposed in Fay (2005) uses an analogy to data transmission through internet, but without presenting any experimental results, while in Hallenborg and Demazeau (2006) a multi-agent approach is developed. However, the later reference is not focused on control approaches for computing the optimal route of DCVs, but on designing a multi-agent hierarchy for baggage handling systems and analyzing the communication requirements. The goal of our work is to develop and compare efficient control approaches for route choice control of each individual DCV.

Theoretically, the maximum performance of such a DCV-based baggage handling system would be obtained if one computes the optimal routes using optimal control (Lewis, 1986). However, as shown in Tarău et al. (2008), this control method becomes intractable in practice due to the heavy computation burden. Therefore, in order to make a trade-off between computational effort and optimality, in Tarău et al. (2009), we have developed centralized and decentralized model predictive control (MPC). MPC is an on-line model-based predictive control design method, see e.g. Maciejowski (2002), that uses a receding horizon principle. As the results confirmed, centralized MPC requires high computation time to determine a solution. The use of decentralized control lowers the computation time, but this comes at the cost of suboptimality.

In this paper we investigate whether the computational effort required for computing the route of each DCV by using centralized MPC can be lowered by using mixed-integer linear programming (MILP). The large computation time obtained in previous work comes from solving nonlinear, nonconvex, mixed-integer optimization problems. Note that such problems may also have multiple local minima and are NP-hard, and therefore, difficult to solve. So, the main contribution of this paper is that we rewrite the route choice problem as an MILP problem, for which efficient solvers are available. The solution of this MILP will then be used as an initial starting point for the original nonlinear optimization problem.

The paper is organized as follows. Section 2 briefly introduces the concepts of MPC and MILP that will be used later on in solving the route choice problem. In Section 2 we also briefly recapitulate an event-driven route choice model that we have developed in Tarău et al. (2008). Afterwards, in Section 3 we simplify the model of Section 2 by considering streams of bags instead of individual bags, and we show that this model can be written in the mixed-integer linear form. In Section 4, we propose two formulations for the MPC optimization problem. The first formulation corresponds to the nonlinear route choice problem, while the second one corresponds to the MILP route choice problem. For a simple case study, we compare of the proposed formulations. The analysis of the simulation results is elaborated in Section 5. The results indicate that computing the DCV routing using the original nonlinear formulation for the MPC optimization problem gives better performance than using the MILP formulation, but at the cost of significantly higher computational efforts. To reduce the computation time while obtaining good results, we solve the original MPC optimization problem, but using at each step the local solution of the corresponding MILP formulation as initial guess. Finally, Section 6 are drawn and directions of future research are presented.

## 2. Background

### *2.1. Model predictive control*

In this section we briefly introduce the basic MPC concepts, which will later on be used in determining the DCV routes.

MPC is an on-line model-based control design method, see e.g. Maciejowski (2002), that uses a receding horizon principle. As illustrated in Figure 1, in the basic MPC approach, given an horizon $N$, at step $k \geq 0$, where $k$ is integer-valued, corresponding to the time instant $t_k = k\tau_s$ with $\tau_s$ the sampling time, the future control sequence $u(k), u(k+1), \ldots, u(k+N-1)$ is computed by solving a discrete-time optimization problem over the period $[t_k, t_k + N\tau_s)$ so that a performance index defined over the considered period $[t_k, t_k + N\tau_s)$ is optimized subject to the operational constraints. After computing the optimal control sequence, only the first control sample is implemented, and subsequently the horizon is shifted. Next, the new state of the system is measured or estimated, and a new optimization problem at time $t_{k+1}$ is solved using this new information. In this way, a feedback mechanism is introduced.

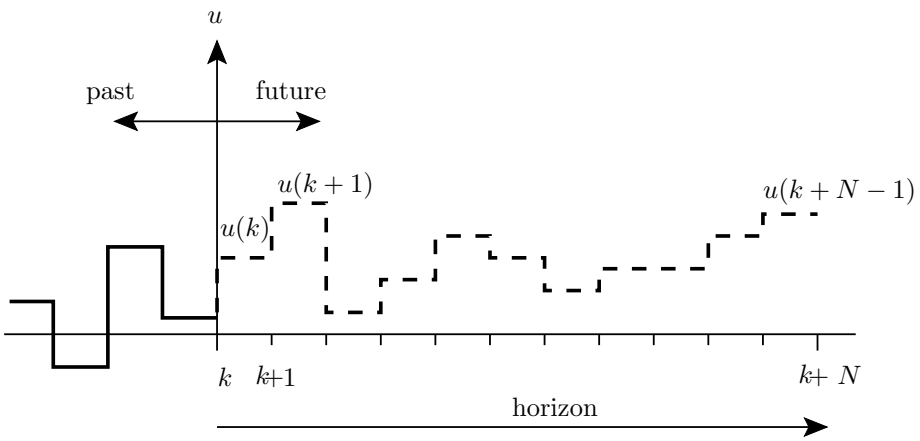



Figure 1: Basic MPC.

*2.2. Mixed-integer linear programming*

Mixed-integer linear programming (MILP) problems are optimization problems with a linear objective function, subject to linear equality and inequality constraints as presented below and where some variables are constrained to be integers. The advantage is that for MILP optimization problems efficient solvers are available (Fletcher and Leyffer, 1998) that allow us to efficiently compute the global optimal solution.

The general formulation for a mixed-integer linear programming problem is the following:

$$
\begin{aligned}
&\min_x c^{\mathrm{T}} x \\
&\text{subject to} \\
&\quad A^{\mathrm{eq}} x = b^{\mathrm{eq}} \\
&\quad Ax \leq b \\
&\quad x^{\mathrm{low}} \leq x \leq x^{\mathrm{up}}
\end{aligned}
$$

where $c$, $x$, $x^{\mathrm{low}}$, $x^{\mathrm{up}}$, $b^{\mathrm{eq}}$, and $b$ are vectors of the same length, with $x^{\mathrm{low}}$ the lower bound of $x$ and $x^{\mathrm{up}}$ its upper bound, and where $A^{\mathrm{eq}}$ and $A$ are matrices. The MILP solvers compute solutions $x$ for the problem above, where elements of $x$ are restricted to integer values.

In this section we present some properties that will be used later on in transforming the original nonlinear route choice model of a DCV-based baggage handling system into an MILP model.

These properties are in fact equivalences, see e.g. Bemporad and Morari (1999), where $f$ is a function defined on a bounded set $X$ with upper and lower bounds $M$ and $m$ for the function values, $\delta$ is a binary variable, $y$ is a real-valued scalar variable, and $\epsilon$ is a small tolerance (typically the machine precision):

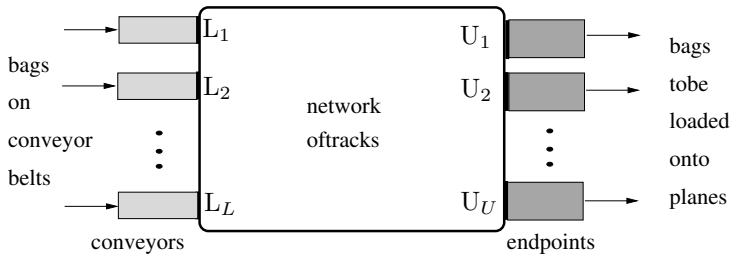



Figure 2: Baggage handling system using DCVs.

**P1**: $[f(x) \leqslant 0] \iff [\delta = 1]$ is true if and only if

$$\begin{cases} f(x) \leqslant M(1-\delta) \\ f(x) \geqslant \epsilon + (m-\epsilon)\delta \ , \end{cases}$$

**P2**: $y = \delta f(x)$ is equivalent to

$$\begin{cases} y \leqslant M\delta \\ y \geqslant m\delta \\ y \leqslant f(x) - m(1-\delta) \\ y \geqslant f(x) - M(1-\delta) \ , \end{cases}$$

The tolerance $\epsilon$ is needed to transform a constraint of the form $y > 0$ into $y \geq 0$, since the solvers available for MILP problems allow only nonstrict inequalities.

*2.3. System description and original model*

In this section we briefly recapitulate the event-driven route choice model of a baggage handling system that we have developed in Tarău et al. (2008).

Consider the general DCV-based baggage handling system with $L$ loading stations and $U$ unloading stations sketched in Figure 2. The DCV-based baggage handling system operates as follows: given a demand of bags and the network of tracks, the route of each DCV has to be computed subject to operational and safety constraints such that the performance of the system is optimized.

In this paper we focus on optimally routing the DCVs through the network. In particular we only consider routing the DCVs from the loading stations to the unloading stations. The problem of managing the empty carts will be considered in future work.

The model of the baggage handling system we have developed in Tarău et al. (2008) consists of a continuous part describing the movement of the individual vehicles transporting the bags through the network, and of the following discrete events:

- loading a new bag into the system,
- unloading a bag that arrives at its end point,

- crossing a junction,
- updating the position of the switch-in at a junction,
- updating the position of a switch-out at a junction,
- updating the velocity of a DCV.

The state of the system consists of the positions of the DCVs in the network and the positions of each switch of the network. According to the discrete-event model that we propose, as long as there are bags to be handled, the system evolves as follows: we shift the current time to the next event time, take the appropriate action, and update the state of the system.

The model of the event-based system described above can be written as

$$\mathbf{t} = \mathcal{M}^{\text{route\_ctrl}}\big(\mathcal{T}, \mathbf{x}(t_0), \mathcal{U}\big) \tag{1}$$

- $\mathbf{t}$ is a vector consisting of the time instants when the bags are loaded onto a DCV or unloaded at their end point,
- $\mathcal{T}$ is a tuple that consists of the time instants when the bags arrive at loading stations,
- $\mathbf{x}(t_0)$ is the initial state of the system with $t_0$ the initial simulation time,
- $\mathcal{U}$ is the switch control input tuple for the entire network.

The operational constraints derived from the mechanical and design limitations of the system are the following:

- the speed of each DCV is bounded between 0 and $v^{\max}$,
- a switch at a junction has to wait at least $\tau^{\text{switch}}$ time units between two consecutive switches in order to avoid the quick and repeated back and forth movements of the switch which may lead to mechanical damage. We assume $\tau^{\text{switch}}$ to be an integer multiple of $\tau_{\text{s}}$ where $\tau_{\text{s}}$ is the sampling time.

The system of inequalities describing the operational constraints of a DCV-based baggage handling system can be written as follows:

$$\mathcal{C}(\mathbf{t}) \leq 0.$$

*2.4. Network*

We represent the network of tracks that the DCVs use to transport the luggage as a directed graph. Then the nodes via which the DCVs enter the network are called loading stations, the nodes via which the DCVs unload the transported bags are called unloading stations, while all other nodes in the network are called junctions. The section of track between two nodes is called link. Note that without loss of generality we can assume that each junction has maximum 2 incoming links and maximum 2 outgoing links, both indexed by $l \in \{0, 1\}$ as sketched in Figure 3. This assumption corresponds to current practice in state-of-the-art baggage handling systems. Each junction with 2 incoming links has a

switch going into the junction (called switch-in hereafter). Each junction with 2 outgoing links has a switch going out of the junction (called switch-out hereafter). Note that a junction can have only switch-in, only switch-out, or both switch-in and switch-out.

## 3. Simplified route choice models

In this section we present simplified route choice models that can be written as MILP models. We consider 3 cases with a gradually increasing complexity where the DCV-based baggage handling system has only one unloading station, more unloading stations close together, and more unloading stations far apart as illustrated in Figure 4. We consider these cases since they grow in complexity and, for each of these cases, additional assumptions have to be made in order to write an MILP model equivalent to the simplified route choice model.

### *3.1. Common assumptions for all 3 cases*

In order to transform the route choice problem into an MILP problem, we first simplify it by assuming the following:

- The DCVs run with maximum speed along the track segment and, if necessary, they wait at the end of the link in a vertical queue. In principle, the queue lengths should be integers as their unit is "number of DCVs", but we will approximate them using reals. Also, we assume that the links are sufficiently long so that congestion is not propagated towards the upstream links, i.e. we assume there is no spillback.

- The dynamic demand $D_i$ of loading station $\mathrm{L}_i$, $i \in \{1, \ldots, L\}$, where $L$ is the number of loading stations, is approximated with a piecewise constant demand. The piecewise constant demand $D_i$ has level changes occurring only at integer multiples of $\tau_\mathrm{s}$. This is necessary in order to easily combine the time when a bag reaches a queue at a junction with the time when the demand changes. So, in the time interval $[t_k, t_{k+1})$, with $t_k = k\tau_\mathrm{s}$, the demand at loading station $\mathrm{L}_i$ is $D_i(k)$.

- For each link a free-flow travel time is assigned. This free-flow travel time represents the time period that a DCV requires to travel on a link in case of no congestion, using, hence, maximum speed. The free-flow travel time of a link is assumed to be always a multiple of $\tau_\mathrm{s}$.

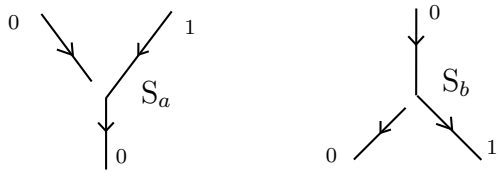



Figure 3: Incoming and outgoing links at a junction.

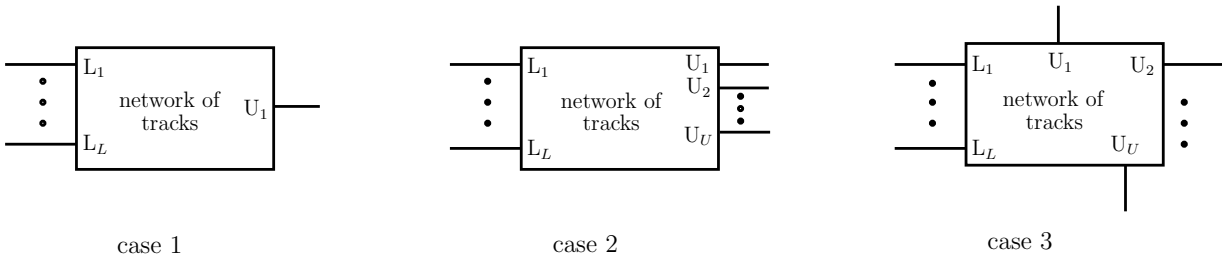



Figure 4: Cases with a gradually increasing complexity: network with one unloading station, more unloading stations close together, more unloading station far apart.

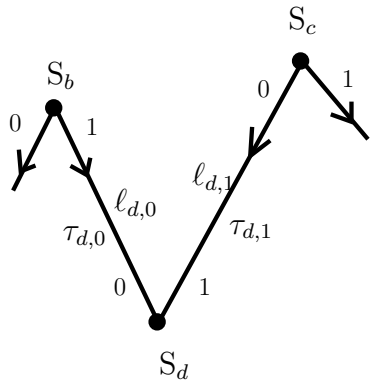



Figure 5: Network elements.

### *3.2. Case 1: one unloading station*

In this section we consider the case of a DCV-based baggage handling system with only one unloading station.

#### *3.2.1. Model*

The control time step for each junction in the network is $\tau_s$. So, at each step $k \geq 0$, for each junction that has two incoming links, we compute a control action that determines the position of the switch *into* a junction for the time period $[t_k, t_{k+1})$. Let $S_a$ be such a junction as sketched in Figure 3. Then the control action that we determine is denoted by $u_a^{\text{sw\_in}}(k)$. If we consider the position of the switch into the junction of Figure 3 to be at step $k$ then $u_a^{\text{sw\_in}}(k) = 1$. At each step $k$, we also compute, for the time period $[t_k, t_{k+1})$, a control action that determines the position of the switch *out of* a junction that has two outgoing links. Let $S_b$ be such a junction. Then the control action that we determine is denoted by $u_b^{\text{sw\_out}}(k)$. For Figure 3, we have $u_b^{\text{sw\_out}}(k) = 1$.

In order to illustrate the derivation of the route choice model let us now consider the most complex cell a network can contain, as depicted in Figure 5 where junction $S_d$ has two neighboring junctions $S_b$ and $S_c$ connected to it via its incoming links.

Next we present how the evolution of the queue length at the end of each incoming link of $S_d$ is determined. At step $k \geq 0$, $u_b^{\text{sw\_out}}(k)$ and $u_c^{\text{sw\_out}}(k)$ are computed for junctions $S_b$ and $S_c$, and $u_d^{\text{sw\_in}}(k)$ for junction $S_d$. Let $\ell_{j,l}$ denote the link between a junction $S_j$ and its

neighbor connected via the incoming link $l$ of $\mathrm{S}_j$ as illustrated in Figure 5. Also, let $q_{j,l}(k)$ denote the length of the queue at the end of link $\ell_{j,l}$ at time instant $t_k$. Recall that each link in the network has been assigned a given free-flow travel time. Then, let $\tau_{d,0}$ denote the free-flow travel time of link $\ell_{d,0}$ and $\tau_{d,1}$ the free-flow travel time of link $\ell_{d,1}$. Hence, the control signals $u_b^{\text{sw\_out}}(k)$ and $u_c^{\text{sw\_out}}(k)$ influence $q_{d,0}$ and $q_{d,1}$ after $\frac{\tau_{d,0}}{\tau_{\mathrm{s}}}$ and respectively $\frac{\tau_{d,1}}{\tau_{\mathrm{s}}}$ time steps[1].

The evolution of the length of the queue at the end of link $\ell_{d,l}$, is given by:

$$q_{d,l}(k+1) = \max\left(0, q_{d,l}(k) + \Big(I_{d,l}\big(k - \frac{\tau_{d,l}}{\tau_{\mathrm{s}}}\big) - O_{d,l}^{\max}(k)\Big)\tau_{\mathrm{s}}\right) \tag{2}$$

where

- $q_{d,l}(k+1)$ is the length of the queue at the end of link $\ell_{d,l}$ at time instant $t_{k+1}$.
- $I_{d,l}(k)$ represents the inflow[2] of link $\ell_{d,l}$ during the period $[t_k, t_{k+1})$. By definition $I_{d,l}(k) = 0$ for $k < 0$.
- $O_{d,l}^{\max}(k)$ is the maximum number of DCVs per time unit that cross $\mathrm{S}_d$ during $[t_k, t_{k+1})$ via link $\ell_{d,l}$.

The maximum number of DCVs per time unit that wait in the queue or arrive at the end of link $\ell_{d,l}$, and that cross $\mathrm{S}_d$ during $[t_k, t_{k+1})$ is defined as follows:

$$O_{d,0}^{\max}(k) = (1 - u_d^{\text{sw\_in}}(k))O^{\max} \tag{3}$$
$$O_{d,1}^{\max}(k) = u_d^{\text{sw\_in}}(k)O^{\max} \tag{4}$$

where $O^{\max}$ is the maximum outflow[3] of a junction. Note that we have used the operator max in (2) since the length of the queue is always larger than or equal to 0.

The inflows $I_{d,0}(k)$ and $I_{d,1}(k)$ are defined as:

$$I_{d,0}(k) = u_b^{\text{sw\_out}}(k)O_b(k) \tag{5}$$
$$I_{d,1}(k) = (1 - u_c^{\text{sw\_out}}(k))O_c(k) \tag{6}$$

with $O_b(k)$ and $O_c(k)$ respectively the outflow of junction $\mathrm{S}_b$ and $\mathrm{S}_c$ during the time interval $[t_k, t_{k+1})$. If $k < 0$, the outflows $O_b(k)$ and $O_c(k)$ are equal to 0 by definition. For $k \geq 0$

[1]Recall that $\tau_{j,l}$ is an integer multiple of $\tau_{\mathrm{s}}$.

[2]The inflow of a link equals the number of DCVs that entered that link per time unit.

[3]The outflow of a junction is defined as the number of DCVs that cross that junction per time unit.

the outflow $O_j(k)$ of a junction $\mathrm{S}_j$ with two incoming links is defined as:

$$O_j(k) = \min\Bigg(O^{\max}, \big(1 - u_j^{\text{sw\_in}}(k)\big)\Big(\frac{q_{j,0}(k)}{\tau_{\mathrm{s}}} + I_{j,0}\big(k - \frac{\tau_{j,0}}{\tau_{\mathrm{s}}}\big)\Big) + u_j^{\text{sw\_in}}(k)\Big(\frac{q_{j,1}(k)}{\tau_{\mathrm{s}}} + I_{j,1}\big(k - \frac{\tau_{j,1}}{\tau_{\mathrm{s}}}\big)\Big)\Bigg) \tag{7}$$

If a junction $\mathrm{S}_j$ has only one incoming link ($l = 0$), then for $k \geq 0$ the outflow $O_j(k)$ is defined as:

$$O_j(k) = \min\Bigg(O^{\max}, \Big(\frac{q_{j,0}(k)}{\tau_{\mathrm{s}}} + I_{j,0}\big(k - \frac{\tau_{j,0}}{\tau_{\mathrm{s}}}\big)\Big)\Bigg) \tag{8}$$

Let $S^{\text{exit}}$ denote the junction connected to the unloading station. Then let $O^{\text{exit}}(k)$ denote the outflow at $S^{\text{exit}}$ during the period $[t_k, t_{k+1})$. The outflow $O^{\text{exit}}(k)$ can be deducted as previously explained. Furthermore, let $U(k)$ denote the outflow at the unloading station during the time interval $[t_k, t_{k+1})$. Then if $S^{\text{exit}}$ has only one outgoing link, $U(k) = O^{\text{exit}}\big(k - \frac{\tau}{\tau_{\mathrm{s}}}\big)$ where $\tau$ is the free-flow travel time between $S^{\text{exit}}(k)$ and the unloading station. If $S^{\text{exit}}$ has 2 outgoing links we assume that the unloading station is link 0 out of $S^{\text{exit}}$. Then $U(k) = \Big(1 - u_{\text{exit}}^{\text{sw\_out}}\big(k - \frac{\tau}{\tau_{\mathrm{s}}}\big)\Big)O^{\text{exit}}\big(k - \frac{\tau}{\tau_{\mathrm{s}}}\big)$ where $u_{\text{exit}}^{\text{sw\_out}}(k)$ expresses the position of the switch out of $S^{\text{exit}}$ during the time interval $[t_k, t_{k+1})$.

*3.2.2. MILP model*

In this section we use the MILP properties presented in Section 2.2 in order to obtain an MILP model for the route choice model given by equations (2)-(7).

We start by transforming (7) using Property **P1**. Let the real-valued variable $f_j^{\text{out}}(k)$ be equal to

$$f_j^{\text{out}}(k) = \big(1 - u_j^{\text{sw\_in}}(k)\big)\Big(\frac{q_{j,0}(k)}{\tau_{\mathrm{s}}} + I_{j,0}\big(k - \frac{\tau_{j,0}}{\tau_{\mathrm{s}}}\big)\Big) + u_j^{\text{sw\_in}}(k)\Big(\frac{q_{j,1}(k)}{\tau_{\mathrm{s}}} + I_{j,1}\big(k - \frac{\tau_{j,1}}{\tau_{\mathrm{s}}}\big)\Big). \tag{9}$$

So, we introduce the binary variable $\delta_{d,1}^{\text{out}}(k)$ which equals 1 if and only if $O^{\max} \leq f_j^{\text{out}}(k)$. Then we rewrite (7) as follows:

$$O_j(k) = \delta_j^{\text{out}}(k)O^{\max} + (1 - \delta_j^{\text{out}}(k))f_j^{\text{out}}(k) \tag{10}$$

where the condition $\delta_j^{\text{out}} = 1$ if and only if $O^{\max} - f_j^{\text{out}}(k) \leq 0$ is equivalent to (cf. Property **P1**):

$$\begin{cases} O^{\max} - f_j^{\text{out}}(k) \leq M(1 - \delta_j^{\text{out}}(k)) \\ O^{\max} - f_j^{\text{out}}(k) \geq \epsilon + (m - \epsilon)\delta_j^{\text{out}}(k) \end{cases}$$

with $M = O^{\max}$ and $m = -\frac{1}{\tau_{\mathrm{s}}}q^{\max}$ where $q^{\max}$ is the maximum possible length of the queue at the end of a link.

But (10) is not yet linear, so, we use Property **P2** and introduce the real-valued scalar variables $y_j^{\text{out}}(k)$ such that:

$$y_j^{\text{out}}(k) = \delta_j^{\text{out}}(k) f_j^{\text{out}}(k)$$

or equivalently:

$$\begin{cases} y_j^{\text{out}}(k) \leqslant M\delta_j^{\text{out}}(k) \\ y_j^{\text{out}}(k) \geqslant 0 \\ y_j^{\text{out}}(k) \leqslant f_j^{\text{out}}(k) \\ y_j^{\text{out}}(k) \geqslant f_j^{\text{out}}(k) - M(1 - \delta_j^{\text{out}}(k)) \ . \end{cases}$$

Hence, one obtains:

$$O_j(k) = O^{\max}\delta_j^{\text{out}}(k) + f_j^{\text{out}}(k) - y_j^{\text{out}}(k)$$

which is linear. Note that (9) can be written as a linear expression by introducing the additional variables $y_{\text{q},j,l}^{\text{in}}(k) = u_j^{\text{sw\_in}}(k) q_{j,l}(k)$ and $y_{\text{I},j,l}^{\text{in}}(k) = u_j^{\text{sw\_in}}(k) I_{j,l}\big(k - \frac{\tau_{j,l}}{\tau_{\text{s}}}\big)$ and the corresponding set of linear inequalities of Property **P2** for $f(x) = q_{j,l}(k)$ with $M = q^{\max}$, and $m = 0$, and $f(x) = I_{j,l}\big(k - \frac{\tau_{j,l}}{\tau_{\text{s}}}\big)$ with $M = O^{\max}$, and $m = 0$ respectively.

Similarly we write the MILP equivalent for (8). Finally, we transform (2) into its MILP equivalent. Let the real-valued variable $f_{d,l}(k)$ be equal to $q_{d,l}(k) + \Big(I_{d,l}\big(k - \frac{\tau_{d,l}}{\tau_{\text{s}}}\big) - O_{d,l}^{\max}(k)\Big)\tau_{\text{s}}$. Additionally we also introduce the binary variable $\delta_{d,l}(k)$ which equals 1 if and only if $f_{d,l}(k) \leq 0$ and we rewrite (2) as:

$$q_{d,l}(k+1) = \big(1 - \delta_{d,l}(k)\big) f_{d,l}(k)) \tag{11}$$

together with the set of linear inequalities of Property **P1** with $M = q^{\max} + O^{\max}\tau_{\text{s}}$ and $m = -O^{\max}\tau_{\text{s}}$.

However (11) is not yet linear. Therefore, we introduce the variable $y_{d,l}(k) = \delta_{d,l}(k) f_{d,l}(k)$ and the set of linear inequalities of Property **P2** for $f(x) = f_{d,l}(k)$, with $M$ and $m$ as defined above, and we obtain:

$$q_{d,l}(k+1) = f_{d,l}(k) - y_{d,l}(k)$$

which is linear.

Next we collect all the variables for the route choice model (i.e. inputs, control variables, and extra variables introduced by the MILP transformations) in vector $\mathbf{v}(k)$ and all the partial queue lengths $q_{j,l}(k)$ in vector $\mathbf{q}(k+1)$. Then the expressions derived above allow us to express $\mathbf{q}(k+1)$ as an affine function of $\mathbf{v}(k)$:

$$\mathbf{q}(k+1) = \Lambda \mathbf{v}(k) + \boldsymbol{\gamma}$$

with a properly defined matrix $\Lambda$ and vector $\boldsymbol{\gamma}$, where $\mathbf{v}(k)$ satisfies a system of linear

equations and inequalities

$$C\mathbf{v}(k) = \mathbf{e}$$
$$F\mathbf{v}(k) \leq \mathbf{g},$$

which corresponds to the linear equations and constraints introduced above by the MILP transformations.

*3.3. Case 2: more unloading stations close together*

In this section we determine the route choice model for a network of tracks with more unloading stations close together as illustrated in Figure 6 where, without loss of generality, we consider that a junction can directly serve all unloading stations (this can be done by lumping together a sequence of junctions that are located closely together and connected to unloading stations). Let $\mathrm{S}^{\mathrm{exit}}$ denote this junction. Also, let $U$ denote the number of unloading stations in the system. Then the free-flow travel time from $\mathrm{S}^{\mathrm{exit}}$ to unloading station $\mathrm{U}_\upsilon$ with $\upsilon \in \{1, \ldots, U\}$, is expressed by an integer $\tau_\upsilon$.

*3.3.1. Assumptions*

In this case we assume that out of the total demand of bags, a certain fraction $\rho_\upsilon$ of bags have to be transported to unloading station $\mathrm{U}_\upsilon$ for $\upsilon = 1, \ldots, U$ such that $\sum_{\upsilon=1}^{U} \rho_\upsilon = 1$. At $\mathrm{S}^{\mathrm{exit}}$ the stream of bags is split into $m$ substreams according to the fractions $\rho_\upsilon$.

*3.3.2. Model*

Note that one can virtually expand junction $\mathrm{S}^{\mathrm{exit}}$ to two junctions $\mathrm{S}^{\mathrm{prev\_exit}}$ and $\mathrm{S}^{\mathrm{exit}}$ connected via a link of length 0. Now $\mathrm{S}^{\mathrm{exit}}$ has only one incoming link. Then the flow model for all junctions in the network except $\mathrm{S}^{\mathrm{exit}}$ can be derived as in the previous case. Next we will determine the flow model corresponding to junction $\mathrm{S}^{\mathrm{exit}}$.

The stream of DCVs waiting at the end of link going into $\mathrm{S}^{\mathrm{exit}}$ can be now divided into substreams (each substream corresponding to an unloading station). Let $q_\upsilon^{\mathrm{exit}}(k)$ denote the queue length of the substream corresponding to unloading station $\mathrm{U}_\upsilon$ for $\upsilon = 1, \ldots, U$

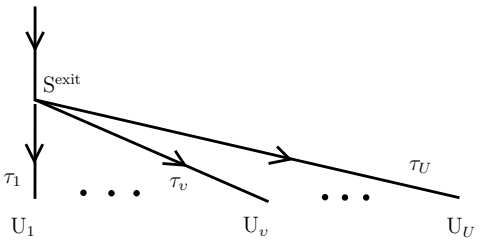



Figure 6: Unloading stations close together.

at time instant $t_k$. The evolution of $q_\upsilon^{\text{exit}}(k)$ is then defined as follows:

$$q_\upsilon^{\text{exit}}(k+1) = q_\upsilon^{\text{exit}}(k) + \big(\rho_\upsilon O^{\text{prev\_exit}}(k) - U_\upsilon(k+\frac{\tau_\upsilon}{\tau_\text{s}})\big)\tau_\text{s}$$

with $O^{\text{prev\_exit}}(k)$ the outflow of $\text{S}^{\text{prev\_exit}}$ and $U_\upsilon(k)$ the outflow of unloading station $\text{U}_\upsilon$ during $[t_k, t_{k+1})$.

We consider two patterns that the low-level switch-out controller could follow:

*Pattern* 1: During the time interval $[t_k, t_{k+1})$ the low-level switch-out controller at $\text{S}^{\text{exit}}$ serves only one unloading station. To determine which unloading station to serve, we introduce the integer control variable $u^{\text{exit}}(k)$ that indicates the index of the unloading station to be served during the time interval $[t_k, t_{k+1})$.

*Pattern* 2: During the time interval $[t_k, t_{k+1})$ all unloading stations are served (we consider fast switching). Then each partial queue is emptied according to the fractions $\rho_\upsilon$ for $\upsilon = 1, \ldots, U$.

Recall that the DCVs that cross $\text{S}^{\text{exit}}$ traveling towards unloading station $\text{U}_\upsilon$ reach the end point with a delay of $\tau_\upsilon$ time units. Then the outflow of unloading station $\text{U}_\upsilon$ with $\upsilon = 1, \ldots, U$ is given by:

*Pattern* 1:

$$U_\upsilon(k) = \begin{cases} \min\big(\frac{q_\upsilon^{\text{exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}})}{\tau_\text{s}} + \rho_\upsilon O^{\text{prev\_exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}}), O^{\max}\big) \text{ if } \upsilon = u^{\text{exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}}), \\ 0 \qquad \text{otherwise.} \end{cases} \tag{12}$$

*Pattern* 2:

$$U_\upsilon(k) = \min\Big(\frac{q_\upsilon^{\text{exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}})}{\tau_\text{s}} + \rho_\upsilon O^{\text{prev\_exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}}), O^{\max}\Big).$$

*3.3.3. MILP model*

The MILP equivalents for the additional equations describing the outflow of an unloading station except (12) can be derived using a reasoning similar to that in Section 3.2.2.

In this section we briefly explain how we write the MILP equivalents for (12). One can introduce $U$ binary variables $\delta_1^{\text{exit}}(k), \ldots, \delta_U^{\text{exit}}(k)$ where $\delta_\upsilon^{\text{exit}}(k) = 1$ means that unloading station $\text{U}_\upsilon$ is served during the time interval $[t_k, t_{k+1})$. Additionally, we introduce the constraint that:

$$\sum_{\upsilon=1}^{U} \delta_\upsilon^{\text{exit}}(k) = 1$$

which means that there can only be one unloading station served at the time. Then, for $\upsilon = 1, \ldots, U$, we have:

$$U_\upsilon(k) = \delta_\upsilon^{\text{exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}}) \min\Big(\frac{q_\upsilon^{\text{exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}})}{\tau_\text{s}} + \rho_\upsilon O_\upsilon^{\text{prev\_exit}}(k-\frac{\tau_\upsilon}{\tau_\text{s}}), O^{\max}\Big)$$

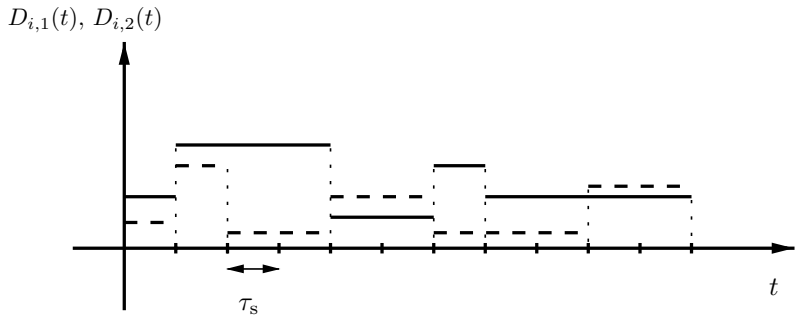



Figure 7: Demand profile at loading station $L_i$ for a network with 2 unloading stations. The continuous line corresponds to unloading station $U_1$ and the dashed line demand corresponds to $U_2$.

### *3.4. Case 3: more unloading stations far apart*

In this section we analyze the case where the track network has more unloading stations far apart.

#### *3.4.1. Assumptions*

For this case we define partial demand patterns at loading stations. So, each loading station has a demand pattern corresponding to each end point. As example we illustrate in Figure 7 the dynamic demand pattern at loading station $L_i$, with $i \in \{1, 2, \ldots, L\}$ for a network with 2 unloading stations. In this figure the piecewise constant demand represented as a continuous line corresponds to unloading station $U_1$, being denoted by $D_{i,1}(t)$, and the dashed piecewise constant demand corresponds to $U_2$, denoted by $D_{i,2}(t)$. Then for a network with $U$ unloading stations, the total demand of $L_i$ during the time interval $[t_k, t_{k+1})$ is given by $D_i(k) = \sum_{v=1}^{U} D_{i,v}(k)$.

Next, since we deal with partial demands at each loading station, we assume that the DCVs wait before the junctions in partial vertical queues according to the unloading station towards which the DCVs travel.

#### *3.4.2. Model*

The control time step for each junction in the network is $\tau_s$. So, at each step $k \geq 0$, for each junction $S_a$ with 2 incoming links, we compute $u_a^{\text{sw\_in}}(k)$, while the switch out of a junction is controlled by a low-level controller as will be presented next.

We consider two patterns that the low-level switch-out controller at $S_j$ with $j \in \{1, \ldots, S\}$ could follow:

*Pattern* 1: During the time interval $[t_k, t_{k+1})$ the low-level switch-out controller serves only one outgoing link of $S_j$. To determine which outgoing link to serve, we compute $u_j^{\text{sw\_out}}(k)$.

*Pattern* 2: During the time interval $[t_k, t_{k+1})$ the low-level controller serves both outgoing links.

According to these patterns, we derive the route choice model by referring again to the network cell illustrated in Figure 5. Without loss of generality we assume that for any junction $\mathrm{S}_z$ directly connected to $\mathrm{U}_\upsilon$, the unloading station is link 0 out of $\mathrm{S}_z$.

*Pattern* 1: We consider partial queues at *the end of each link* and corresponding to each unloading station $\mathrm{U}_\upsilon$ with $\upsilon \in \{1, \ldots, U\}$. Then the evolution of the length of the partial queue $q_{d,l,\upsilon}$ is given by:

$$q_{d,l,\upsilon}(k+1) = q_{d,l,\upsilon}(k) + \big(I_{d,l,\upsilon}(k - \frac{\tau_{d,l}}{\tau_\mathrm{s}}) - O_{d,l,\upsilon}(k)\big)\tau_\mathrm{s}$$

where $I_{d,l,\upsilon}(k)$ is the partial inflow at link $\ell_{d,l}$ and $O_{d,l,\upsilon}(k)$ is the partial outflow of link $\ell_{d,l}$ during the time interval $[t_k, t_{k+1})$ corresponding to $\mathrm{U}_\upsilon$.

The inflow $I_{d,0,\upsilon}(k)$ is defined as $I_{d,0,\upsilon}(k) = u_b^{\text{sw\_out}}(k)\Big((1 - u_b^{\text{sw\_in}}(k))O_{b,0,\upsilon}(k) + u_b^{\text{sw\_in}}(k)O_{b,1,\upsilon}(k)\Big)$ if $\mathrm{S}_b$ has 2 incoming links $I_{d,0,\upsilon}(k) = (1 - u_b^{\text{sw\_out}}(k))O_{b,0,\upsilon}(k)$ if $\mathrm{S}_b$ has only one incoming link. Similarly, one can define $I_{d,1,\upsilon}(k)$.

The partial outflows $O_{j,l,\upsilon}(k)$ at the end of link $\ell_{j,l}$ ($l = u_j^{\text{sw\_in}}(k)$) are determined such that we have maximal exhaustion of the available capacity as described in **Algorithm 1** for $O_{j,l,\upsilon} = O_\upsilon^{\text{alg}}$ and $q_{j,l,\upsilon} = q_\upsilon^{\text{alg}}$. Note that if junction $\mathrm{S}_j$ has 2 incoming links, then $O_{j,1-l,\upsilon}(k) = 0$.

The outflow of unloading station $\mathrm{U}_\upsilon$ during the period $[t_k, t_{k+1})$ is given by:

$$U_\upsilon(k) = \min\left(\Big(1 - u_z^{\text{sw\_out}}(k - \frac{\tau_\upsilon}{\tau_\mathrm{s}})\Big)O_{z,0,\upsilon}(k - \frac{\tau_\upsilon}{\tau_\mathrm{s}}), O^{\max}\right).$$

*Pattern* 2: We consider partial queues at *each junction* $\mathrm{S}_j$ with $j \in \{1, \ldots, S\}$ corresponding to each unloading station $\mathrm{U}_\upsilon$. Then we determine the partial outflows $O_{j,\upsilon}(k)$ for a junction $\mathrm{S}_j$ such that $\sum_{\upsilon=1}^{U} O_{j,\upsilon}(k) \leq O^{\max}$. We consider again a fair distribution over all flows as described in **Algorithm 1** for $O_{j,\upsilon} = O_\upsilon^{\text{alg}}$ and $q_{j,\upsilon} = q_\upsilon^{\text{alg}}$.

Based on historical data, for each junction $\mathrm{S}_j$ we can determine $U$ fixed turning rates $\eta_{j,\upsilon}$ with $\upsilon = 1, \ldots, U$. These fixed turning rates represent the fraction of the partial queue $q_{j,\upsilon}(k)$ that will be sent to link 0 out of $\mathrm{S}_j$ during $[t_k, t_{k+1})$. Then $\tau_\mathrm{s} \sum_{\upsilon=1}^{U} \eta_{j,\upsilon} O_{j,\upsilon}(k)$ DCVs will be sent towards the outgoing link 0 of $\mathrm{S}_j$, and $\tau_\mathrm{s} \sum_{\upsilon=1}^{U} (1 - \eta_{j,\upsilon}) O_{j,\upsilon}(k)$ DCVs will be sent towards its outgoing link 1.

Then the evolution of the length of the partial queue $q_{d,\upsilon}$ is given by:

$$q_{d,\upsilon}(k+1) = q_{d,\upsilon}(k) + \big(\xi_{d,\upsilon}(k) - O_{d,\upsilon}(k)\big)\tau_\mathrm{s}$$

where $\xi_{d,\upsilon}(k)$ expresses the number of DCVs going towards unloading station $\mathrm{U}_\upsilon$ that enter the partial queue at junction $\mathrm{S}_d$ during the time interval $[t_k, t_{k+1})$, $\xi_{d,\upsilon}(k) = (1 - u_d^{\text{sw\_in}}(k))(1 - \eta_{b,\upsilon})O_{b,\upsilon}(k - \frac{\tau_{d,0}}{\tau_\mathrm{s}}) + u_d^{\text{sw\_in}}(k)\eta_{c,\upsilon}O_{c,\upsilon}(k - \frac{\tau_{d,1}}{\tau_\mathrm{s}})$.

Accordingly, $U_\upsilon(k) = \min\left(\left(1 - u_z^{\text{sw\_out}}(k - \frac{\tau_\upsilon}{\tau_\text{s}})\right)\eta_{z,\upsilon}O_{z,\upsilon}(k - \frac{\tau_\upsilon}{\tau_\text{s}}), O^{\max}\right)$.

**Algorithm 1. Outflow distribution at the end of link $\ell_{j,l}$**

$\Omega = \{1, 2, \ldots, U\}$
**while** $\Omega \neq \emptyset$ **do**
  $\Lambda = \arg\min_{\upsilon \in \Omega} \left(q_\upsilon^{\text{alg}}(k) + \tau_\text{s} I_{j,l,\upsilon}(k)\right)$
  **for** all $\upsilon \in \Lambda$ **do**
    $O_\upsilon^{\text{alg}}(k) = \min\left(\frac{q_\upsilon^{\text{alg}}(k)}{\tau_\text{s}} + I_{j,l,\upsilon}(k), \frac{O^{\max}}{|\Omega|}\right)$
    $O^{\max} \leftarrow O^{\max} - O_\upsilon^{\text{alg}}(k)$
  **end for**
  $\Omega \leftarrow \Omega \setminus \Lambda$
**end while**

Let us now consider *Pattern* 1 and derive (as example) the output of Algorithm 1 for link $\ell_{d,l}$ of the cell illustrated in Figure 5 and for $U = 2$. According to **Algorithm 1**, if $q_{d,l,1}(k) + I_{d,l,1}(k)\tau_\text{s} \geq q_{d,l,2}(k) + I_{d,l,2}(k)\tau_\text{s}$ then the outflow $O_{d,l,\upsilon}(k)$, for $\upsilon = 1, 2$ is given by:

$$O_{d,l,1}(k) = \min\left(\frac{q_{d,l,1}(k)}{\tau_\text{s}} + I_{d,l,1}(k), \frac{O^{\max}}{2}\right) \tag{13}$$

$$O_{d,l,2}(k) = \min\left(\frac{q_{d,l,2}(k)}{\tau_\text{s}} + I_{d,l,2}(k), O^{\max} - \frac{q_{d,l,1}(k)}{\tau_\text{s}} - I_{d,l,1}(k)\right) \tag{14}$$

else

$$O_{d,l,1}(k) = \min\left(\frac{q_{d,l,1}(k)}{\tau_\text{s}} + I_{d,l,1}(k), O^{\max} - \frac{q_{d,l,2}(k)}{\tau_\text{s}} - I_{d,l,2}(k)\right) \tag{15}$$

$$O_{d,l,2}(k) = \min\left(\frac{q_{d,l,2}(k)}{\tau_\text{s}} + I_{d,l,1}(k), \frac{O^{\max}}{2}\right) \tag{16}$$

*3.4.3. MILP model*

In this section we will transform (13)-(16) into their MILP equivalents. The rest of the MILP route choice model for the case with more unloading stations far apart, can be derived using a reasoning similar to that in Section 3.2.2.

To transform (13)-(16), we introduce the binary variables $\delta_{d,1}(k) = 1$ if and only if $q_{d,l,1}(k) \geq q_{d,l,2}(k)$, $\delta_{d,2}(k) = 1$ if and only if $\frac{q_{d,l,1}(k)}{\tau_\text{s}} \leq \frac{O^{\max}}{2}$, $\delta_{d,3}(k) = 1$ if and only if $\frac{q_{d,l,2}(k)}{\tau_\text{s}} \leq \frac{O^{\max}}{2}$, and $\delta_{d,4}(k) = 1$ if and only if $\frac{q_{d,l,2}(k)}{\tau_\text{s}} \leq \frac{O^{\max}}{2} - \frac{q_{d,l,1}(k)}{\tau_\text{s}}$ together with the system of linear inequalities of Property **P1**. Then the outflows $O_{d,l,1}(k)$ and $O_{d,l,2}(k)$

can be written as follows:

$$O_{d,l,1}(k) = \delta_{d,1}(k)\left(\delta_{d,2}(k)\frac{q_{d,l,1}(k)}{\tau_{\mathrm{s}}} + \big(1-\delta_{d,2}(k)\big)\frac{O^{\max}}{2}\right) + \big(1-\delta_{d,1}(k)\big)\left(\delta_{d,4}(k)\frac{q_{r,1}(k)}{\tau_{\mathrm{s}}}\big(1-\delta_{d,4}(k)\big)\big(O^{\max}-\frac{q_{d,l,2}(k)}{\tau_{\mathrm{s}}}\big)\right) \quad (17)$$

$$O_{d,l,2}(k) = \delta_{d,1}(k)\left(\delta_{d,4}(k)\frac{q_{d,l,2}(k)}{\tau_{\mathrm{s}}} + \big(1-\delta_{d,4}(k)\big)\big(O^{\max}-\frac{q_{d,l,1}(k)}{\tau_{\mathrm{s}}}\big)\right) + \big(1-\delta_{d,1}(k)\big)\left(\delta_{d,3}(k)\frac{q_{d,l,2}(k)}{\tau_{\mathrm{s}}} + \big(1-\delta_{d,3}(k)\big)\frac{O^{\max}}{2}\right) \quad (18)$$

To transform (17)-(18) into MILP equations one has to further introduce real-valued scalar variables and the corresponding sets of linear inequalities of Property **P2** using a reasoning similar to that in Section 3.2.2.

## 4. Model predictive route choice control

In this section we define the MPC optimization problem for both the nonlinear and the MILP case.

Recall that we want to assess the performance of MPC when using the original nonlinear model (1) and when using the approximated[4] MILP model. Therefore, the performance index should be linear or piecewise affine.

### *4.1. MPC objective function*

The first objective of a baggage handling system is to transport all the checked-in or transfer bags to the corresponding end points before the planes have to be loaded. However, due to the airports' logistics, an end point is allocated to a plane with a given time period before the departure of the plane. Hence, the baggage handling system performs optimally if each of the bags to be handled arrives at its given end point in a specific time window. In previous work we have considered an objective function that penalizes both the overdue time and the additional storage time as follows:

$$J^{\mathrm{tot}}(\mathbf{t}) = \sum_{i=1}^{N^{\mathrm{bags}}} \big(\max(0, t_{z,\upsilon}^{\mathrm{unload}} - t_{z,\upsilon}^{\mathrm{load\_plane}}) + \lambda \max(0, t_{z,\upsilon}^{\mathrm{load\_plane}} - \tau_{z,\upsilon}^{\mathrm{open}} - t_{z,\upsilon}^{\mathrm{unload}})\big)$$

where $N^{\mathrm{bags}}$ is the number of bags to be handled, $t_{z,\upsilon}^{\mathrm{unload}}$ is the time instant when the bag with index $z$ is unloaded at its endpoint $\mathrm{U}_\upsilon$ (the index $\upsilon$ is determined by index $z$ since each bag has to be unloaded at a specific end point), $t_{z,\upsilon}^{\mathrm{load\_plane}}$ is the time instant when the end

[4]The MILP model is an approximation of the nonlinear model due to the assumptions that we have made to simplify the model (in particular due to the approximation of the dynamic demand of bags with a piecewise constant demand).

point $\mathrm{U}_\upsilon$ closes, and $\tau_{z,\upsilon}^{\text{open}}$ is the maximum possible length of the time window for which the end point $\mathrm{U}_\upsilon$ is open for bag index $z$. The weighting parameter $\lambda > 0$ expresses the penalty for the additionally stored bags. This objective function is nonlinear and, hence, cannot be used to recast the nonlinear routing problem into an MILP one. Therefore, in this paper we transform the nonlinear objective above into a piecewise constant one that approximates $J^{\text{tot}}(\mathbf{t})$. We now consider the objective of reaching a desired outflow for each unloading station.

Without loss of generality, in this paper we consider that each destination has assigned only one flight. But this can be easily extended to the general case.

Let $[t_\upsilon^{\text{load\_plane}} - \tau_\upsilon^{\text{open}}, t_\upsilon^{\text{load\_plane}})$ be the time window when the endpoint $\mathrm{U}_\upsilon$ is open where $t_\upsilon^{\text{load\_plane}}$ is the time instant when the end point $\mathrm{U}_\upsilon$ closes and the last bags are loaded onto the plane and $\tau_\upsilon^{\text{open}}$ is the time period for which the end point $\mathrm{U}_\upsilon$ stays open. For the simplicity of the explanation we assume that $t_\upsilon^{\text{load\_plane}}$ and $\tau_\upsilon^{\text{open}}$ are integers multiple of $\tau_{\text{s}}$.

Note that the desired outflow at each unloading station is in general a dynamic signal. But this can always be approximated with a piecewise constant one. Since the objective is to have each bag arriving at its end point within a given time interval, we can define the desired outflow at unloading station $\mathrm{U}_\upsilon$ with $\upsilon \in \{1, \ldots, U\}$ as follows:

$$U_\upsilon^{\text{desired}}(k) = \begin{cases} \dfrac{N_\upsilon^{\text{bags}}}{\tau_\upsilon^{\text{open}}} & \text{if } \dfrac{t_\upsilon^{\text{load\_plane}} - \tau_\upsilon^{\text{open}}}{\tau_{\text{s}}} \leq k \leq \dfrac{t_\upsilon^{\text{load\_plane}}}{\tau_{\text{s}}} \\ 0 & \text{otherwise} \end{cases}$$

where $N_\upsilon^{\text{bags}}$ is the total number of bags to be sent to unloading station $\mathrm{U}_\upsilon$ during the simulation period.

However, to add some additional gradient to this objective function and make sure that all the bags will be handled, we add the weighted length of queues at each junction in the network, but only for time steps bigger than $k_\upsilon^{\text{stop}}$ with $\upsilon \in \{1, \ldots, U\}$, where $k_\upsilon^{\text{stop}} = \dfrac{t_\upsilon^{\text{load\_plane}}}{\tau_{\text{s}}}$.

Let $U_\upsilon(k)$ denote the actual outflow of unloading station $\mathrm{U}_\upsilon$ during the period $[t_k, t_{k+1})$. Then, such a performance index at step $k$, for a prediction horizon $N$, can be written as follows:

$$J_{k,N} = \sum_{\upsilon=1}^{U} \left( w_\upsilon \sum_{i=k}^{k+N-1} \left( |U_\upsilon(i) - U_\upsilon^{\text{desired}}(i)| + \alpha_{i,\upsilon} \sum_{j=1}^{S} \lambda_{j,\upsilon} q_j(i) \right) \right) \tag{19}$$

where

- $\alpha_{i,\upsilon}$ is a binary variable equal to 1 if $i > k_\upsilon^{\text{stop}}$ and 0 otherwise;
- $q_j(k)$ denotes the sum of the partial queue lengths at junction $\mathrm{S}_j$ at time instant $t_k$;
- $w_\upsilon > 0$ is a penalty that expresses the importance of the flight;

- $\lambda_{j,\upsilon} > 0$ is a weighting parameter that expresses the penalty[5] on junction $\mathrm{S}_j$.

Now let us consider the case where $k + N - 1 \leq k_\upsilon^{\text{stop}}$. Since we want to write the problem $\min \sum_{\upsilon=1}^{U} w_\upsilon \sum_{i=k}^{k+N-1} |U_\upsilon(i) - U_\upsilon^{\text{desired}}(i)|$ as a linear programming problem, the MPC optimization problem can be rewritten as follows:

$$\begin{aligned}
&\min \sum_{\upsilon=1}^{U} w_\upsilon \sum_{i=k}^{k+N-1} U_\upsilon^{\text{diff}}(i) \\
&\text{subject to} \\
&\qquad \mathbf{q}(i+1) = \Lambda \mathbf{v}(i) + \boldsymbol{\gamma} \\
&\qquad U_\upsilon^{\text{diff}}(i) \geqslant U_\upsilon(i) - U_\upsilon^{\text{desired}}(i) \\
&\qquad U_\upsilon^{\text{diff}}(i) \geqslant -U_\upsilon(i) + U_\upsilon^{\text{desired}}(i) \\
&\qquad\qquad \text{for } i = k, \ldots, k+N-1
\end{aligned}$$

where, for appropriately defined $\Lambda$ and $\boldsymbol{\gamma}$, $\mathbf{v}(i)$ consists of all the variables for the MILP route choice model at step $i$ and $\mathbf{q}(i+1)$ consists of all the partial queue lengths such that $\mathbf{v}(i)$ satisfies a system of linear equations and inequalities

$$\begin{aligned}
C_i \mathbf{v}(i) &= \mathbf{e_i} \\
F_i \mathbf{v}(i) &\leq \mathbf{g_i}.
\end{aligned}$$

The MPC optimization problem above is a linear programming problem that has as optimal solution

$$U_\upsilon^{\text{diff},*}(i) = \max\left(U_\upsilon^*(i) - U_\upsilon^{\text{desired}}(i), -U_\upsilon^*(i) + U_\upsilon^{\text{desired}}(i)\right) = |U_\upsilon^*(i) - U_\upsilon^{\text{desired}}(i)|.$$

For the case where $k+N-1 > k_\upsilon^{\text{stop}}$ we will still obtain an MILP optimization problem by applying a similar procedure.

*4.2. Optimization problems*

Next we formulate the optimization problem for both the nonlinear and the MILP model formulations.

The nonlinear MPC optimization problem is defined as:

$$\begin{aligned}
&\min_{\mathcal{U}_{k,N}} J_{k,N}(\mathcal{U}_{k,N}) \\
&\text{subject to} \\
&\qquad \mathbf{t}(k) = \mathcal{M}^{\text{route_ctrl}}\big(\mathcal{T}, \mathbf{x}(t_k), \mathcal{U}_{k,N}\big) \\
&\qquad \mathcal{C}(\mathbf{t}(k)) \leq 0
\end{aligned}$$

[5]Since a baggage handling system has to transport all the checked-in or transfer bags to the corresponding end points before the planes have to be loaded, the closer junction $\mathrm{S}_j$ with $j \in \{1, \ldots, S\}$ is to the end point $\mathrm{U}_\upsilon$ with $\upsilon \in \{1, \ldots, U\}$, the smaller is the weighting parameter $\lambda_{j,\upsilon}$.

where $\mathcal{U}_{k,N} = (\mathbf{u}(k), \mathbf{u}(k+1), \ldots, \mathbf{u}(k+N-1))$ with $\mathbf{u}(k) = [u_1^{\text{sw\_in}}(k)\, u_1^{\text{sw\_out}}(k)\, \ldots\, u_S^{\text{sw\_in}}(k)\, u_S^{\text{sw\_out}}(k)]^{\text{T}}$, while the outflows of the unloading stations are determined via simulation.

In order to solve this mixed-integer nonlinear optimization problem one could use e.g. *mixed-integer nonlinear programming* solvers such as `bqpd`, `miqpBB`, `minlpBB` of the Tomlab/MINLP optimization toolbox of Matlab, *genetic* algorithms, *simulated annealing* of the Matlab optimization toolbox *Genetic Algorithm and Direct Search*, or *tabu search* see e.g. Dowsland (1993); Floudas (1995); Glover and Laguna (1997); Reeves and Rowe (2002).

Similarly, the linear (MILP) MPC optimization problem is defined as:

$$\begin{aligned}
&\min_{\mathcal{V}_{k,N}} J_{k,N}(\mathcal{V}_{k,N}) \\
&\text{subject to} \\
&\qquad \mathbf{A}^{\text{eq}}_{k,N}\mathcal{V}_{k,N} = \mathbf{b}^{\text{eq}}_{k,N} \\
&\qquad \mathbf{A}_{k,N}\mathcal{V}_{k,N} \leq \mathbf{b}_{k,N}
\end{aligned}$$

where $\mathcal{V}_{k,N} = (\mathbf{v}(k), \mathbf{v}(k+1), \ldots, \mathbf{v}(k+N-1))$ with $\mathbf{v}(k)$ the vector of MILP variables at step $k$ for the corresponding case study.

To solve the MILP optimization problem one could use solvers such as CPLEX, Xpress-MP, GLPK, see e.g. Atamtürk and Savelsbergh (2005).

Recall from Section 3 that in order to be able to write the DCV routing problem as an MILP optimization problem we have simplified the original problem by making some approximations. Therefore, computing the route for each DCV in the network by solving nonlinear MPC optimization problems will result in a better performance than by solving the MILP optimization problems. However, this happens at the cost of higher computational efforts. So, one could use MILP to compute a good initial point for the nonlinear optimization problem and this would reduce the computation time. One could also use directly the MILP solution, but at the cost of suboptimality.

## 5. Case study

We are interested in analyzing the trade-off between performance and computation time when using the two formulations of the MPC optimization problems. To this aim we consider as benchmark case study the network depicted in Figure 8. This network consists of four loading stations and three unloading stations close together connected via single-direction track segments, where the free-flow travel time is indicated for each link. Note that we have chosen this case study since the network contains all the important elements of a real network, but it allows us to faster assess the efficiency of the proposed routing approach.

The evolution of each queue at the end of a link in the network above, $q_{j,l}$ for $j = 1, 2, 3, 4$ and $l = 0, 1$ is given by:

$$q_{j,l}(k+1) = \max\big(0, f_{j,l}(k)\big)$$

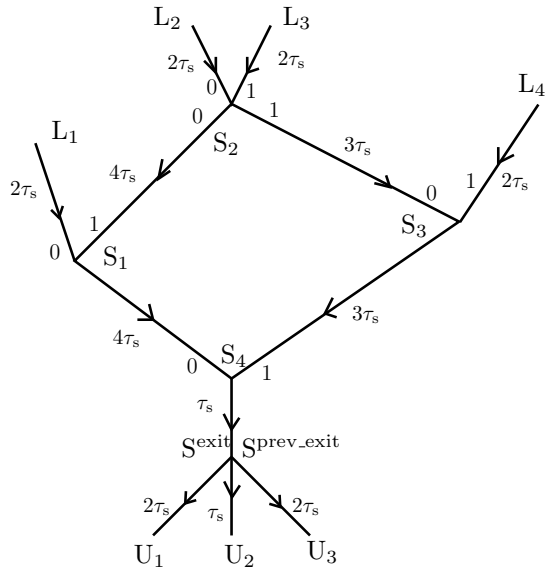



Figure 8: Case study for a DCV-based baggage handling system.

with $f_{j,l}(k)$ defined as follows:

$$
\begin{aligned}
f_{1,0}(k) =& q_{10}(k) + \big(D_1(k-2) - (1 - u_1^{\text{sw\_in}}(k))O^{\max}\big)\tau_{\text{s}} \\
f_{1,1}(k) =& q_{11}(k) + \big((1 - u_2^{\text{sw\_out}}(k-4))O_2(k-4) - u_1^{\text{sw\_in}}(k)O^{\max}\big)\tau_{\text{s}} \\
f_{2,0}(k) =& q_{20}(k) + \big(D_2(k-2) - (1 - u_2^{\text{sw\_in}}(k))O^{\max}\big)\tau_{\text{s}} \\
f_{2,1}(k) =& q_{21}(k) + \big(D_3(k-2) - u_2^{\text{sw\_in}}(k)O^{\max}\big)\tau_{\text{s}} \\
f_{3,0}(k) =& q_{30}(k) + \big(u_2^{\text{sw\_out}}(k-3)O_2(k-3) - (1 - u_3^{\text{sw\_in}}(k))O^{\max}\big)\tau_{\text{s}} \\
f_{3,1}(k) =& q_{31}(k) + \big(D_4(k-2) - (u_3^{\text{sw\_in}}(k)O^{\max}\big)\tau_{\text{s}} \\
f_{4,0}(k) =& q_{40}(k) + \big(O_1(k-4) - (1 - u_4^{\text{sw\_in}}(k))O^{\max}\big)\tau_{\text{s}}
\end{aligned}
$$

$$
f_{4,1}(k) = q_{41}(k) + \big(O_3(k-3) - u_4^{\text{sw\_in}}(k)O^{\max}\big)\tau_{\text{s}}
$$

where $O_j(k)$ with $j \in \{1, 2, 3\}$ is given by (7).

The evolution of the partial queues corresponding to unloading station $\text{U}_\upsilon$ for $\upsilon = 1, \ldots, U$ at the end of the link leading to $S^{\text{exit}}$ is given by:

$$
q_\upsilon^{\text{exit}}(k+1) = q_\upsilon^{\text{exit}}(k) + \big(\rho_\upsilon O^{\text{prev\_exit}}(k) - U_\upsilon(k + \frac{\tau_\upsilon}{\tau_{\text{s}}})\big)\tau_{\text{s}}
$$

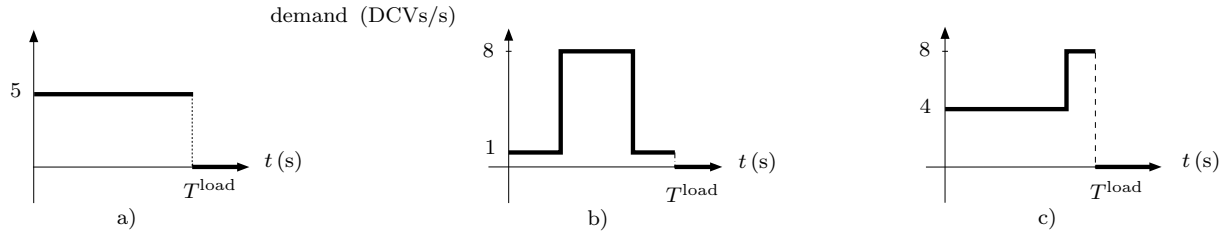



Figure 9: Demand profile.

with $U_\upsilon(k)$ given by (12), and

$$O^{\text{prev\_exit}}(k) = \min\Bigg(O^{\max}, \Big(\frac{q_{40}(k-1)}{\tau_{\text{s}}} + O_1(k-5)\Big)\big(1 - u_4^{\text{sw\_in}}(k-1)\big) + \Big(\frac{q_{41}(k-1)}{\tau_{\text{s}}} + O_3(k-4)\Big)u_4^{\text{sw\_in}}(k-1)\Bigg).$$

We assume that the velocity of each DCV varies between 0 m/s and 10 m/s. In order to faster assess the efficiency of our control method we assume that we do not start with an empty network but with a network already populated by DCVs transporting bags. We consider two different initial states of the network:

- at $t_0$ 20 DCVs are waiting in a queue on incoming link 0 of junction $\text{S}_1$ and 40 DCVs are waiting in a queue on incoming link 1 of $\text{S}_1$.
- at $t_0$ 20 DCVs are waiting in a queue on incoming link 0 of junction $\text{S}_1$ and 40 DCVs on incoming link 1 of $\text{S}_1$. Additionally, 40 DCVs are waiting in a queue on incoming link 0 of junction $\text{S}_3$ and 20 DCVs are waiting in a queue on incoming link 1 of $\text{S}_3$.

To compare the results we have considered 6 scenarios where 200 bags are loaded at each loading station, for the two different initial states of the system and where $\rho_\upsilon = 25\,\%$ for $\upsilon = 1, 2$, and $\rho_3 = 50\,\%$. For this particular case study we consider $w_\upsilon = 1$ for $\upsilon = 1, 2, 3$ since the bags are not assigned to a specific destination from the beginning of the simulation. We simulate a period of 600 s, for a network where the capacity of each junction is 5 DCVs/s. The simulation time step $\tau_{\text{s}}$ is set to 20 s. We have considered the bag arrival pattern for each loading station according to the three different classes of demand profiles sketched in Figure 9, where $T^{\text{load}} = 100\,\text{s}$ is the total loading time. The demand of each loading station equals 0 for $t > T^{\text{load}}$. These scenarios will involve very tight transportation since the time window for each unloading station is $[150\,\text{s}, 300\,\text{s})$ (the last bag that enters the system can only arrive in time at the corresponding endpoint if the DCV travels the shortest route with maximum speed).

Let us now compare the results obtained when using the proposed predictive control method with different formulations of the optimization problem.

To solve the original mixed-integer nonlinear MPC optimization problem we have chosen a *simulated annealing* algorithm of the Matlab optimization toolbox *Genetic Algorithm and Direct Search* implemented via the function `simulannealbnd`, using multiple initial

points, and adapted to obtain integer variables. Based on experiments (Tarău et al., 2008), we have noticed that both the genetic algorithm of the Matlab optimization toolbox *Genetic Algorithm and Direct Search* implemented via the function `ga` and the simulated annealing algorithm offer a good trade-off between performance and computational effort. However, the `ga` solver is a multi-run approach, the starting point being selected always randomly by the algorithm. Recall that we could use the MILP solution as an initial feasible guess when solving the nonlinear optimization problem. Consequently, we have chosen the simulated annealing algorithm since the `simulannealbnd` solver computes a local solution starting from a user-given initial feasible solution. For solving the MILP optimization we have used the CPLEX solver implemented through the `cplex` interface function of the Matlab *Tomlab* toolbox. As prediction horizon we have considered $N = 8$ for all MPC optimization problems.

In order to have faster computation of the routing solution, in this paper we apply all the $N$ control samples that have been computed by the MPC method. Therefore, to have real time computation, each time we compute the future control sequence, the CPU time has to be smaller than $\tau_{\mathrm{s}}N$ (for our case study this means that the CPU time has to be smaller than 160 s at each MPC step). Note that the total computation time required to determine the complete routing solution depends then also on how many MPC steps we need to perform.

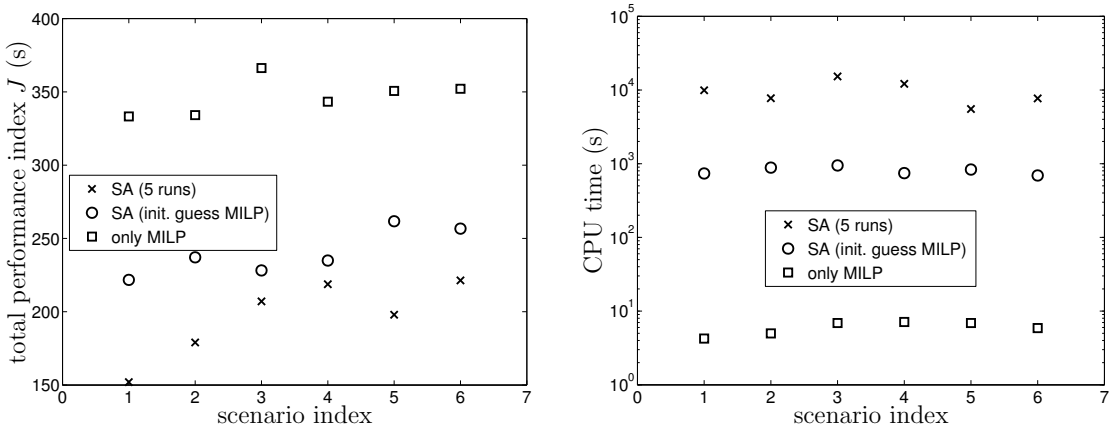
Based on simulations we now compare, for the given scenarios, the results obtained for the proposed formulations of the optimization problem. The results of the simulations are reported in Figure 10 where the total performance of the system is defined as:

$$J = \sum_{v=1}^{U} \left( w_v \sum_{k=0}^{K^{\mathrm{sim}}-1} \left( |U_v(k) - U_v^{\mathrm{desired}}(k)| + \alpha_k \sum_{j=1}^{S} \lambda_{j,v} q_j(k) \right) \right)$$

with $K^{\mathrm{sim}}\tau_{\mathrm{s}}$ the real time at which the last DCV transporting a bag through the network arrives at its end point. These results confirm that computing the route choice using the original nonlinear formulation for the MPC optimization problem gives better performance than using the MILP formulation, but at the cost of higher computational effort. Finally, we have computed the DCV route choice with the *simulated annealing* algorithm by using as initial feasible solution for the original nonlinear MPC problem the control sequence computed by solving the MILP optimization problem. As illustrated in Figure 10, the results indicate that this last method offers a good trade-off between performance and computational effort.

## 6. Conclusions

We have considered the problem of efficiently computing (sub)optimal routes for destination coded vehicle (DCV) that transport bags in an airport on a "mini" railway network. This results in a nonlinear, nonconvex, mixed-integer optimization problem that is very expensive to solve in terms of computational effort. Therefore, we have proposed an alternative approach for reducing the complexity of the computations by approximating the




(a) closed-loop performance (b) computation time

Figure 10: Comparison of the results obtained using the proposed MPC formulations. At each MPC step we solve the MILP optimization only, we use the MILP solution as feasible initial guess to solve the original MPC optimization problem, we solve the MPC original optimization using solvers for nonlinear mixed-integer optimization problems.

nonlinear optimization problem by a mixed-integer linear programming (MILP) problem. The advantage is that for MILP problems solvers are available that allow us to efficiently compute the global optimal solution. These two formulations of the optimization problem have been used to compute the route of DCVs using model predictive control (MPC) for a benchmark case study.

Simulation results confirm that computing the route choice using the original nonlinear formulation for the MPC optimization problem gives usually better performance than using the MILP formulation, but at the cost of significantly higher computational efforts. To reduce the computation time while obtaining good results, one can solve the original MPC optimization problem, but using at each step the local solution of the corresponding MILP formulation as initial guess.

In future work we will apply this method to more complex case studies and scenarios. We will perform sensitivity analysis on the deviation with respect to the desired outflow and the queues length. Furthermore, we will also consider reducing the computation time by developing hierarchical route choice control.

## Acknowledgments

This research is supported by the VIDI project "Multi-Agent Control of Large-Scale Hybrid Systems" (DWV.6188) of the Dutch Technology Foundation STW, Applied Science division of NWO and the Technology Programme of the Dutch Ministry of Economic Affairs, by the BSIK project "Next Generation Infrastructures (NGI)", by the Transport Research Centre Delft, by the Delft Research Centre Next Generation Infrastructures, and by the European 7th framework STREP project "Hierarchical and Distributed Model Predictive Control (HD-MPC)" (contract number INFSO-ICT-223854).

## References


A. Atamtürk and M. Savelsbergh. Integer-programming software systems. *Annals of Operations Research*, 140(1):67–124, November 2005.

A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.

R. de Neufville. The baggage system at Denver: Prospects and lessons. *Journal of Air Transport Management*, 1(4):229–236, December 1994.

K.A. Dowsland. Simulated annealing. In C.R. Reeves, editor, *Modern Heuristic Techniques for Combinatorial Problems*, chapter 2, pages 20–69. John Wiley & Sons, Inc., New York, New York, USA, 1993.

A. Fay. Decentralized control strategies for transportation systems. In *Proceedings of the 2005 IEEE International Conference on Control and Automation*, pages 898–903, Budapest, Hungary, June 2005.

R. Fletcher and S. Leyffer. Numerical experience with lower bounds for MIQP branch-and-bound. *SIAM Journal on Optimization*, 8(2):604–616, May 1998.

C.A. Floudas. *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press, New York, USA, 1995.

H. Gang, J.S. Shang, and L.G. Vargas. A neural network model for the free-ranging AGV route-planning problem. *Journal of Intelligent Manufacturing*, 7(3):217–227, 1996.

F. Glover and F. Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, Massachusetts, USA, 1997.

K. Hallenborg and Y. Demazeau. Dynamical control in large-scale material handling systems through agent technology. In *Proceedings of the 2006 IEEE /WIC/ACM International Conference on Intelligent Agent Technology*, pages 637–645, Hong Kong, China, December 2006.

D.E. Kaufman, J. Nonis, and R.L. Smith. A mixed integer linear programming model for dynamic route guidance. *Transportation Research Part B: Methodological*, 32(6): 431–440, 1998.

F.L. Lewis. *Optimal Control*. John Wiley & Sons, New York, New York, USA, 1986.

J.M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, Harlow, UK, 2002.

C.R. Reeves and J.E. Rowe. *Genetic Algorithms – Principles and Perspectives: A Guide to GA Theory*. Kluwer Academic Publishers, Norwell, Massachusetts, USA, 2002.

A. Tarău, B. De Schutter, and J. Hellendoorn. Travel time control of destination coded vehicles in baggage handling systems. In *Proceedings of the 17th IEEE International Conference on Control Applications*, pages 293–298, San Antonio, Texas, USA, September 2008.

A.N. Tarău, B. De Schutter, and J. Hellendoorn. Route choice control of automated baggage handling systems. *Transportation Research Record*, (2106):76–82, 2009.