Delft University of Technology

Delft Center for Systems and Control

Technical report 10-041

Predictive control for 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 control for baggage handling systems using mixed integer linear programming," *Proceedings of the 5th IFAC International Conference on Management and Control of Production Logistics (MCPL 2010)*, Coimbra, Portugal, pp. 16–21, Sept. 2010.

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_041.html

Predictive Control for Baggage Handling Systems using Mixed Integer Linear Programming

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

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 which is 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. We also assess the performance of the proposed (nonlinear and MILP) formulations of the MPC optimization problem using a benchmark case study.

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. 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 by 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.

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 by 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 that uses a receding horizon principle. As the results confirmed, centralized MPC requires a 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 that can have multiple local minima. So, in this paper we rewrite the routing problem as an MILP problem, for which efficient solvers are available. The MILP solution can then be used as an initial starting point for the original nonlinear optimization problem.

The paper is organized as follows. Section 2 briefly recapitulates the event-driven routing model developed in (Tarău et al., 2008). Next, in Section 3, we simplify the event-driven model and recast it into MILP form. In Section 4 we propose two MPC approaches where we solve the optimization problems corresponding to the nonlinear routing problem and to the MILP-based problem respectively. For a simple case study, we compare the proposed formulations in Section 5. Finally, Section 6 concludes the paper.

2. PRELIMINARIES

2.1 Model predictive control

Since later on we will use model predictive control (MPC) for determining the routes of the DCVs in the network, in this section we briefly introduce the basic MPC concepts.

MPC is an on-line model-based control design method, see e.g. (Maciejowski, 2002), that uses a receding horizon principle. In the basic MPC approach, given an horizon N, at step $k \geq 0$, where k is integer-valued, corresponding to the time instant $t_k = k\tau_\mathrm{s}$ with τ_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 $[t_k, t_k + N\tau_\mathrm{s})$ 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.

2.2 Mixed integer linear programming

Mixed integer linear programming (MILP) problems are optimization problems with a linear objective function dealing with real and integer variables, subject to linear equality and inequality constraints. The advantage is that for MILP problems efficient solvers are available (Fletcher and Leyffer, 1998) that allow us to efficiently compute the global optimal solution.

Next we present two properties that will be used 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, δ is a binary variable, y is a real-valued scalar variable, and ϵ is a small tolerance (typically the machine precision):

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

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

2.3 System description and original model

Now 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 1. The DCV-based baggage handling system operates as follows: given a demand of bags and the network of tracks, the route of each DCV (from a given loading station to the corresponding unloading station) has to be computed subject to operational and safety constraints such that the performance of the system is optimized.

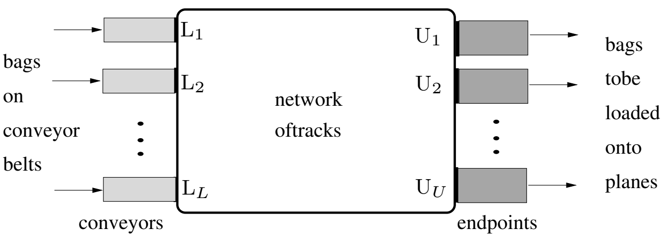


Fig. 1. Baggage handling system using DCVs.

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 onto a DCV, unloading a bag that arrives at its end point, updating the position of the switches into and out of a junction, and updating the speed 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 of Tarău et al. (2008), 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 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 $v^{\max}$, while a switch at a junction has to wait at least τ^{switch} time units between two consecutive toggles in order to avoid the quick and repeated back and forth movements of the switch which may lead to mechanical damage. We assume τ^{switch} to be an integer multiple of τ_s where τ_s is the sampling time.

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 at most 2 incoming links and at most 2 outgoing links, both indexed by $l \in \{0, 1\}$.

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).

3. SIMPLIFIED DCV ROUTING MODELS

In this section we present simplified route choice models that can be written as MILP models. We consider two cases with a gradually increasing complexity where the DCV-based baggage handling system has respectively only one unloading station and more unloading stations. We consider these two 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 that is equivalent to the simplified route choice model.

3.1 Common assumptions for both cases

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.

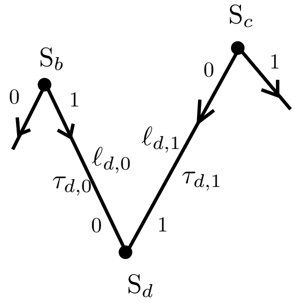


Fig. 2. Network elements.

- The dynamic demand D_i of loading station L_i, $i \in \{1, \ldots, L\}$, is approximated with a piecewise constant demand; in the time interval $[t_k, t_{k+1})$, with $t_k = k\tau_\mathrm{s}$, the demand at loading station 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 always assumed to be a multiple of τ_s.

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.

Model To illustrate the derivation of the route model we now consider the most complex cell a network can contain, as depicted in Figure 2 where junction S_d has 2 neighboring junctions S_b and S_c connected to it via its incoming links.

The control time step for each junction in the network is τ_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_d be such a junction as sketched in Figure 2. Then the control action that we determine is denoted by $u_d^{\text{sw\_in}}(k)$ and expresses the index of the incoming link that the switch is positioned on. At each step k, we also compute a control action that determines the position of the switch *out of* a junction during $[t_k, t_{k+1})$. Let S_b be such a junction. Then the control action that we determine is denoted by $u_b^{\text{sw\_out}}(k)$.

Next we present how the evolution of the queue length at the end of each incoming link of S_d is determined. At step $k \geq 0$, we compute $u_b^{\text{sw\_out}}(k)$, $u_c^{\text{sw\_out}}(k)$, and $u_d^{\text{sw\_in}}(k)$. Let $\ell_{j,l}$ denote the link between a junction S_j and its neighbor connected via the incoming link l of S_j as illustrated in Figure 2. 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 $\tau_{d,0}$ and $\tau_{d,1}$ denote the free-flow travel time of link $\ell_{d,0}$ and $\ell_{d,1}$ respectively. 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 $\dfrac{\tau_{d,0}}{\tau_\mathrm{s}}$ and respectively $\dfrac{\tau_{d,1}}{\tau_\mathrm{s}}$ time steps.

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

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

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 of link $\ell_{d,l}$ during the period $[t_k, t_{k+1})$, and $O_{d,l}^{\max}(k)$ is the maximum number of DCVs per time unit that cross 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 S_d during $[t_k, t_{k+1})$ is defined as follows:

$$O_{d,0}^{\max}(k) = (1 - u_d^{\text{sw\_in}}(k))O^{\max} \tag{2}$$

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

where $O^{\max}$ is the maximum outflow of a junction. Note that we have used the operator max in (1) 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:

$$I_{d,0}(k) = u_b^{\text{sw\_out}}(k)O_b(k) \tag{4}$$

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

with $O_b(k)$ and $O_c(k)$ respectively the outflow of junction S_b and S_c during the time interval $[t_k, t_{k+1})$.

For $k \geq 0$ the outflow $O_d(k)$ of a junction S_d with two incoming links is defined as:

$$O_d(k) = \min\Bigg((1 - u_d^{\text{sw\_in}}(k))\Big(\frac{q_{d,0}(k)}{\tau_\mathrm{s}} + I_{d,0}(k - \frac{\tau_{d,0}}{\tau_\mathrm{s}})\Big) + u_d^{\text{sw\_in}}(k)\Big(\frac{q_{d,1}(k)}{\tau_\mathrm{s}} + I_{d,1}(k - \frac{\tau_{d,1}}{\tau_\mathrm{s}})\Big), O^{\max}\Bigg) \tag{6}$$

The unloading station is modeled as follows. Let S^{exit} denote the junction connected to the unloading station. Then let $O^{\text{exit}}(k)$ denote the outflow at S^{exit} during the period $[t_k, t_{k+1})$. The outflow $O^{\text{exit}}(k)$ can be deduced in a similar way as explained above. Furthermore, let $U(k)$ denote the outflow at the unloading station during $[t_k, t_{k+1})$. We assume that the unloading station is always link 0 out of S^{exit}. Then

$$U(k) = \Big(1 - u_{\text{sw\_out}}^{\text{exit}}(k - \frac{\tau}{\tau_\mathrm{s}})\Big)O^{\text{exit}}(k - \frac{\tau}{\tau_\mathrm{s}})$$

where $u_{\text{sw\_out}}^{\text{exit}}(k)$ expresses the position of the switch out of S^{exit} during $[t_k, t_{k+1})$ and τ is the free-flow travel time between $S^{\text{exit}}(k)$ and the unloading station.

MILP model Now 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 (1)-(6).

We start by transforming (6) using Property **P1**. Let the real-valued variable $f_j^{\text{out}}(k)$ be equal to

$$f_d^{\text{out}}(k) = \Big(\frac{q_{d,0}(k)}{\tau_\mathrm{s}} + I_{d,0}(k - \frac{\tau_{d,0}}{\tau_\mathrm{s}})\Big)(1 - u_d^{\text{sw\_in}}(k)) + \Big(\frac{q_{d,1}(k)}{\tau_\mathrm{s}} + I_{d,1}(k - \frac{\tau_{d,1}}{\tau_\mathrm{s}})\Big)u_d^{\text{sw\_in}}(k). \tag{7}$$

So, we introduce the binary variable $\delta_{d,1}^{\text{out}}(k)$ which equals 1 if and only if $O^{\max} \leq f_d^{\text{out}}(k)$. Then we rewrite (6) as follows:

$$O_d(k) = \delta_d^{\text{out}}(k)O^{\max} + (1 - \delta_d^{\text{out}}(k))f_d^{\text{out}}(k) \tag{8}$$

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

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

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

But (8) is not yet linear, so, we use Property **P2** and introduce the real-valued scalar variables $y_d^{\text{out}}(k)$ such that:

$$y_d^{\text{out}}(k) = \delta_d^{\text{out}}(k) f_d^{\text{out}}(k).$$

Hence, one obtains:

$$O_d(k) = O^{\max} \delta_d^{\text{out}}(k) + f_d^{\text{out}}(k) - y_d^{\text{out}}(k)$$

which is linear. Note that (7) can be written as a linear expression by introducing the additional variables $y_{\text{q},d,l}^{\text{in}}(k) = u_d^{\text{sw\_in}}(k) q_{d,l}(k)$ and $y_{\text{I},d,l}^{\text{in}}(k) = u_d^{\text{sw\_in}}(k) I_{d,l}(k - \frac{\tau_{d,l}}{\tau_{\text{s}}})$ and the corresponding set of linear inequalities of Property **P2** for $f(x) = q_{d,l}(k)$ with $M = q^{\max}$, and $m = 0$, and $f(x) = I_{d,l}(k - \frac{\tau_{d,l}}{\tau_{\text{s}}})$ with $M = O^{\max}$, and $m = 0$ respectively.

Finally, we transform (1) into its MILP equivalent. Let the real-valued variable $f_{d,l}(k)$ be equal to $q_{d,l}(k) + \Big(I_{d,l}(k - \frac{\tau_{d,l}}{\tau_{\text{s}}}) - O_{d,l}^{\max}(k)\Big)\tau_{\text{s}}$. 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 (1) as:

$$q_{d,l}(k+1) = (1 - \delta_{d,l}(k)) f_{d,l}(k)) \tag{9}$$

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 (9) 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 a vector $\mathbf{v}(k)$ and all the partial queue lengths in a 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 Λ 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

Now we analyze the case where the track network has several unloading stations.

Assumptions For this case we define partial demand patterns at loading stations. So, each loading station L_i has a demand pattern $D_{i,\upsilon}(\cdot)$ corresponding to each end point U_υ with $\upsilon \in \{1, \ldots, U\}$. 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 $D_i(k) = \sum_{\upsilon=1}^{U} D_{i,\upsilon}(k)$.

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.

Model We now derive the route choice model by referring again to the network cell illustrated in Figure 2. We consider partial queues at the end of each link and corresponding to each unloading station U_υ. Then the evolution of the length of the partial queue $q_{d,l,\upsilon}$ is given by:

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

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 U_υ.

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 S_b has 2 incoming links $I_{d,0,\upsilon}(k) = (1 - u_b^{\text{sw\_out}}(k))O_{b,0,\upsilon}(k)$ if S_b has only one incoming link. Similarly, one can define $I_{d,1,\upsilon}(k)$.

The partial outflows $O_{d,l,\upsilon}(k)$ at the end of link $\ell_{d,l}$ ($l = u_d^{\text{sw\_in}}(k)$) are determined such that we have maximal exhaustion of the available capacity as described in **Algorithm 1**. Note that if junction S_d has 2 incoming links, then $O_{d,1-l,\upsilon}(k) = 0$ since only the partial queues at the end of the incoming link indexed by $l = u_d^{\text{sw\_in}}(k)$ are emptied during $[t_k, t_{k+1})$.

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

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

Without loss of generality we assume that for any junction S_z directly connected to U_υ, the unloading station is link 0 out of S_z. Then the outflow of unloading station U_υ during the period $[t_k, t_{k+1})$ is given by:

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

MILP model The MILP routing model for this case can be derived using a reasoning that is similar to that in Section 3.2 (for more details see (Tarău, 2010)).

[1] In **Algorithm 1**, $|\Omega|$ represents the cardinality of the set Ω.

4. MODEL PREDICTIVE ROUTE CHOICE CONTROL

Next we derive the MPC optimization problems that we will later on solve to determine the optimal routing. We consider both the nonlinear and the MILP case.

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 only 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 $[t_v^{\text{load\_plane}} - \tau_v^{\text{open}}, t_v^{\text{load\_plane}})$ where $t_v^{\text{load\_plane}}$ is the time instant when the end point U_v closes and the last bags are loaded onto the plane, and τ_v^{open} is the time period for which the end point U_v stays open for a specific flight. We have assumed $t_v^{\text{load\_plane}}$ and τ_v^{open} to be integer multiple of τ_{s}. As a consequence, in this paper we consider the objective of reaching a desired outflow for each unloading station. In this paper we consider that each destination has only one flight assigned to it. However, this can be easily extended to the general case.

Hence, one objective is to achieve a desired outflow at destination U_v during the prediction period. 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 U_v with $v \in \{1, \ldots, U\}$ as follows: $U_v^{\text{desired}}(k) = \dfrac{N_v^{\text{bags}}}{\tau_v^{\text{open}}}$ if $k \geq \dfrac{t_v^{\text{load\_plane}} - \tau_v^{\text{open}}}{\tau_{\text{s}}}$ and $k \leq \dfrac{t_v^{\text{load\_plane}}}{\tau_{\text{s}}}$ with N_v^{bags} the total number of bags to be sent to unloading station U_v during the simulation period, and $U_v^{\text{desired}}(k) = 0$ otherwise.

However, to add some additional gradient to the objective function and to 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_v^{stop} with $v \in \{1, \ldots, U\}$, where $k_v^{\text{stop}} = \dfrac{t_v^{\text{load\_plane}}}{\tau_{\text{s}}}$.

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

$$J_{k,N} = \sum_{v=1}^{U} \left(w_v \sum_{i=k}^{k+N-1} \Big(|U_v(i) - U_v^{\text{desired}}(i)| + \alpha_{i,v} \sum_{j=1}^{S} \lambda_{j,v} q_j(i) \Big) \right)$$

where $\alpha_{i,v}$ is a binary variable equal to 1 if $i > k_v^{\text{stop}}$ and 0 otherwise, $q_j(k)$ denotes the sum of the partial queue lengths at junction S_j at time instant t_k, $w_v > 0$ is a penalty that expresses the importance of the flight, $\lambda_{j,v} > 0$ is a weighting parameter that expresses the penalty on junction S_j.

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

$$\begin{aligned}
&\min \sum_{v=1}^{U} w_v \sum_{i=k}^{k+N-1} U_v^{\text{diff}}(i) \\
&\text{subject to} \\
&\qquad \text{system's dynamics} \\
&\qquad \text{operational constraints} \\
&\qquad U_v^{\text{diff}}(i) \geqslant U_v(i) - U_v^{\text{desired}}(i) \\
&\qquad U_v^{\text{diff}}(i) \geqslant -U_v(i) + U_v^{\text{desired}}(i) \\
&\qquad\qquad \text{for } i = k, \ldots, k+N-1.
\end{aligned}$$

Since U_v^{diff} only appears on the left-hand side of the last inequalities, it is easy to verify that this problem has as optimal solution

$$\begin{aligned}
U_v^{\text{diff},*}(i) &= \max\left(U_v^*(i) - U_v^{\text{desired}}(i), -U_v^*(i) + U_v^{\text{desired}}(i)\right) \\
&= |U_v^*(i) - U_v^{\text{desired}}(i)|.
\end{aligned}$$

For the case where $k+N-1 > k_v^{\text{stop}}$ we can apply a similar procedure.

Hence, when using the original model we have to solve mixed integer nonlinear optimization problems, while when using the MILP model we solve MILP problems.

4.2 Optimization algorithms

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

In general, computing the route for each DCV in the network when solving nonlinear MPC optimization problems will give better performance than when solving the MILP optimization problems (due to the simplified assumptions used to write the MILP model), but 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

Let us now analyze 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 3. This network consists of 4 loading stations, 5 junctions, and 2 unloading stations where the free-flow travel time is indicated for each link.

We assume that the velocity of each DCV varies between 0 m/s and 20 m/s. In order to faster assess the efficiency

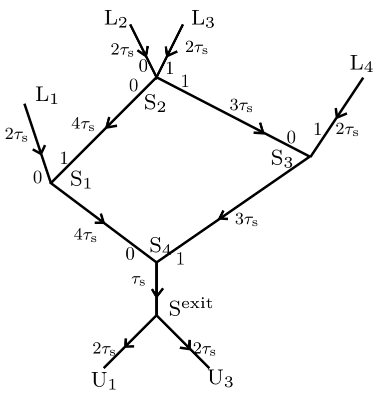


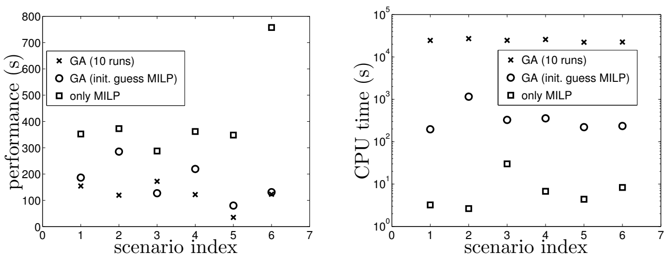
Fig. 3. Case study.

of our control method we do not start with an empty network but with a network already populated by DCVs transporting bags.

To compare the results we have considered 6 scenarios where 800 bags have to be handled for different initial states of the system, queues at different junctions, and different demand profiles at each loading station. We simulate a period of 600 s, for a network where the capacity of each junction is 5 DCVs/s. The simulation time step τ_s is set to 20 s.

In order to solve the MILP optimization problem we have used the CPLEX solver implemented through the `cplex` interface function of the Matlab *Tomlab* toolbox, while to solve the original mixed integer nonlinear MPC optimization problem we have chosen the genetic algorithm of the Matlab optimization toolbox *Genetic Algorithm and Direct Search*. This function allows the user to set the initial search point. Then we can apply directly the results of the MILP optimization to the original nonlinear route choice problem, we can solve the nonlinear optimization problem starting from random initial points only, or we can use the solution of the MILP optimization problem as a good initial guess when solving the nonlinear optimization. As prediction horizon we have considered $N = 8$ for all MPC optimization problems.

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 4. These results confirm that computing the route choice using the original nonlinear formulation for the MPC optimization problem gives better performance than using only the MILP formulation. However, this happens at the cost of a much higher computational effort. Finally, we also compute the DCV routing using as initial feasible solution for the original nonlinear MPC problem the control sequence determined by solving the MILP optimization problem. As illustrated in Figure 4, the results indicate that this last method offers a good trade-off between performance and computational effort.



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

Fig. 4. Comparison of the obtained results.

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 network of tracks. In general, 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 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.

In future work we will apply this method to more complex case studies. We will also consider reducing the computation time by developing hierarchical route choice control.

ACKNOWLEDGEMENTS

Research supported by the STW-VIDI project "Multi-Agent Control of Large-Scale Hybrid Systems", the BSIK project "Next Generation Infrastructures", 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 of Large Scale Systems".

REFERENCES

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

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

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

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

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

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

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

Tarău, A. (2010). *Model-Based Control for Postal Automation and Baggage Handling*. Ph.D. thesis, Delft University of Technology.

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