Model Predictive Control with Memory-based Discrete Search for Switched Linear Systems

Rie B. Larsen ∗ Bilge Atasoy ∗ Rudy R. Negenborn ∗

Abstract: Controlling systems with both continuous and discrete actuators using model predictive control is often impractical, since mixed-integer optimization problems are too complex to solve sufficiently fast. This paper proposes a parallelizable method to control both the continuous input and the discrete switching signal for linear switched systems. The method uses ideas from Bayesian optimization to limit the computation to a predefined number of convex optimization problems. The recursive feasibility and stability of the method is guaranteed for initially feasible solutions. Results from simulated experiments show promising performances and computation times.

Keywords: predictive control, integer control, parallel computation, computational methods, stability analysis, mixed-integer optimization, rolling horizon

1. INTRODUCTION

Many systems are controlled by a combination of discrete and continuous actuators. A class of them can be described by switched linear dynamics without dwell-time where both the continuous input and the discrete switching signals are decisions taken by the applied controller. For very small systems, model predictive control (MPC) can be applied directly to improve the performance of these systems under constraints. However, since the system must be described using both continuous and discrete variables, the MPC controller must solve a mixed-integer program every time the control sequence is updated. For larger systems, the computation of this mixed-integer program may be unacceptably long compared to the system specifications.

Switched linear systems can be categorized as either internally or externally forced. An overview of stability analysis for different classes of switched linear systems can be found in Lin and Antsaklis (2009), while Zhu and Antsaklis (2015) reviews the literature on MPC for switched systems. The literature on MPC for internally forced systems focuses mainly on the case where the switching signal is state dependent (piecewise affine systems) as in, e.g., Morari and Bari (2006), or on systems under uncontrolled switching, as in, e.g., Zhang et al. (2016). This paper considers externally forced systems, hence systems where the applied MPC must decide on both the continuous inputs and the discrete switching signals.

To improve the computation time of MPC for externally forced switched linear systems, most frameworks focus on solving each iteration faster. Often the proposed integer solvers are iteration based, e.g., Axehill et al. (2014) or Naik and Bemporad (2017), which limits how fast the computation can be and increases the computation time uncertainty. This complicates the implementation of MPC, since a typical MPC re-computes the input at predefined time intervals. One way to avoid performing iterative computations of the discrete variables is using convex relaxations of the mixed-integer problem. In Sager et al. (2012) it is shown that continuous variables can be used to approximate binary variables in the MPC optimization problem if the timesteps are sufficiently small. In Gau et al. (2017) this result is used to control a switch linear system without other input than the switching signal. In Prada et al. (2009) the switching signals are modelled as switching times resulting in a non-linear optimization problem with solely continuous variables. Mendes et al. (2017) decreased the number of iterations needed to solve the mixed-integer problem by distributing the computation between different agents coupled by auxiliary variables.

Another stream of research, namely that on suboptimal MPC, acknowledge that it is not always possible to obtain the optimal solution. This research establishes bounds and criteria on the suboptimality and the properties of the implemented control law that ensures stability and feasibility of the system. Stability and recursive feasibility of the suboptimal MPC is in, e.g., Lazar and Heemels (2009) and Pannocchia et al. (2011) established by imposing specific improvements in the objective function. In contrast, Scokaert et al. (1999) ensure stability and recursive feasibility by improving a known solution which satisfy stability and feasibility criteria.

In this paper, the proposed method finds better solutions by remembering the performance at previous timesteps,
and is thus called MPC with Memory-based Discrete search (MDS). The method is outlined in Figure 1 with notation as introduced in the remainder of this paper.

MPC with MDS solves the part of the usual MPC problem, which is continuous in the variables, for a set of switching signal sequences. It then implements the first elements of the sequence with the best performance, if this performance is better than that of a stabilizing input sequence computed based on the previous MPC prediction and a known conservative control law. If the known stabilizing input sequence performs better, its solution corresponding to the current timestep is implemented and the process starts over.

It is assumed that the performance of a switching signal sequence at a given time is comparable to the performance of related sequences at the previous timestep. This assumption is used to choose the switching signal sequences for which MPC with MDS computes the convex optimization problem. The ideas are analogous to Bayesian Optimization where the current knowledge of a function is used to assess which function evaluation to perform at the next iteration. See e.g. Jones et al. (1998) or Snoek et al. (2012) for an introduction. MPC with MDS utilizes the rolling horizon from MPC to obtain information about not only current solutions, but also future solutions. In this fashion, the optimization problems in MPC with MDS have low complexity and can be solved in parallel.

The remainder of the paper is organized as follows: in Section 2, the considered system and problem is specified. In Section 3, MPC with MDS is presented together with an analysis of stability and recursive feasibility. In Section 4, MPC with MDS is assessed using a simulation and compared with three other controllers. Finally, Section 5 draws the main conclusions and outlines future research.

2. SYSTEM AND PROBLEM DEFINITION

We consider a discrete time, switched, linear system without dwell-time with the following dynamics:

\[
x_{t+1} = A_\sigma x_t + B_\sigma u_t,
\]

where \( x_t \in \mathbb{R}^n \) and \( u_t \in \mathbb{R}^{m_\sigma} \) are the state of and the input applied to the system at discrete time \( t \). The switching signal \( \sigma_t \in S = I_{[1,b]} \) is the input at time \( t \), that determines from finite sets, the dynamic matrices \( A_\sigma \in \mathcal{A} = \{ A_1, A_2, \ldots, A_l \} \) and \( B_\sigma \in \mathcal{B} = \{ B_1, B_2, \ldots, B_l \} \). \( I_{[a,b]} \) denotes the set of integers \( \{ x \in \mathbb{I} | a \leq x \leq b \} \). The system is controllable for one or more switching signals.

**Definition 1.** The basic switching signal is denoted by \( \gamma \). It is one switching signal \( \gamma \in S \) for which system (1) is controllable. The corresponding dynamics \( x_{t+1} = A_\gamma x_t + B_\gamma u_t \) will be referred to as the basic dynamics.

The system’s state and inputs are constrained by

\[
x_t \in X \subseteq \mathbb{R}^n \quad \forall t
\]

\[
u_t \in U_\sigma \subseteq \mathbb{R}^{m_\sigma} \forall t.
\]

The set \( X \) is convex, closed and contains the origin in its interior, while each \( U_\sigma \) is convex and compact. \( U_\gamma \) contains the origin.

It is assumed that more information about the basic dynamics is known.

**Assumption 1.** A control law \( u = \kappa(x) \), called the safe control law, and a set \( X_f \) are known for which the following is true:

1. The basic dynamics under the safe control law, i.e. \( x_{t+1} = A_\gamma x_t + B_\gamma \kappa(x_t) \) is asymptotically stable.
2. \( X_f \subseteq X \), \( 0 \in X_f \) and \( X_f \) is closed.
3. \( \kappa(x) \in U_\gamma \) \( \forall x \in X_f \).
4. \( A_\gamma x + B_\gamma \kappa(x) \in X_f \) \( \forall x \in X_f \).

2.1 Problem definition

This paper considers stability to the equilibrium at the origin, but the results can be generalized to set point stability.

When system (1) is controlled by an MPC controller, the controller solves at timestep \( t \) the following mixed-integer program:

\[
\min_{u_0, \ldots, u_{N-1}} J(\varsigma, u, x)
\]

\[
s.t. \quad x_0 = x_t
\]

\[
x_{k+1} = A_{\varsigma_k} x_k + B_{\varsigma_k} u_k \quad \forall k \in [0, N-1]
\]

\[
x_k \in X_k, u_k \in U_{\varsigma_k}, \quad \forall k \in [0, N-1]
\]

\[
x_N \in X_f,
\]

where variables \( x_k \in \mathbb{R}^n \), \( \varsigma_k \in I_{[1,l]} \) and \( u_k \in \mathbb{R}^{m_{\varsigma_k}} \) represent the controller’s model of the state, switching signal and input to the system at prediction time \( t + k \), respectively. \( N \) is the prediction horizon. The cost matrices \( R \) and \( Q \) are symmetric and positive semi-definite, while \( Q \) is symmetric and at least positive semi-definite. Bold symbols denote ordered time sequences, e.g., \( \varsigma_{[0,N]} = \{ \varsigma_0, \varsigma_1, \ldots, \varsigma_N \} \), where each element \( \varsigma_k \) is a vector related to timestep \( k \). When appropriate, the interval subscript is omitted for a simpler notation.

**Assumption 2.** The stage cost for the basic dynamics \( l(x, u, \gamma) = x_k^T Q x_k + u_k^T R u_k \) and the
Algorithm 1 MPC with MDS

1: Let $\hat{\varsigma}(t), \hat{u}(t)$ and $\hat{x}(t)$ be a feasible solution to (4)-(9)
2: while system is to be controlled do
3:   implement $\sigma_{t} = \hat{\varsigma}_{0}, u_{t} = u_{0}$
4:   $t = t+1$
5:   measure $x_{t}$
6:   Compute $S(t)$ using Algorithm 2
7:   for all $\varsigma \in S(t)$ do
8:     if (10)-(14) is feasible do
9:       $F_{t}(\varsigma) = F_{t}(\varsigma)$
10:      else
11:       $F_{t}(\varsigma) = M$
12:      end if
13:      $Y_{t}(\varsigma) = \beta J(\hat{\varsigma}^{+}(t-1), \hat{u}^{+}(t-1), x^{+}(t-1))$
14:      end for
15:     Find $F_{t}^{*}(\varsigma^{*}) = \min_{\varsigma \in S(t)} F_{t}(\varsigma)$
16:     if $F_{t}^{*}(\varsigma^{*}) \leq J(\hat{\varsigma}^{+}(t-1), \hat{u}^{+}(t-1), x^{+}(t-1))$ do
17:       $\hat{\varsigma}(t) = \varsigma^{*}, \hat{u}(t) = u^{*}, \hat{x}(t) = x^{*}$
18:     else
19:       $\hat{\varsigma}(t) = \hat{\varsigma}^{+}(t-1), \hat{u}(t) = \hat{u}^{+}(t-1), \hat{x}(t) = \hat{x}^{+}(t-1)$
20:     end if
21: end while

final cost $V_{f}(x) = x^{T}Q_{N}x$ are such that $V_{f}(A_{x}x + B_{x}\kappa(x)) - V_{f}(x) + l(x, \kappa(x), \gamma) \leq 0 \forall x \in X_{f}$.

Notice that Assumption 1 and 2 can be satisfied by the definition of $\kappa(x), X_{f}$ and $V_{f}(x)$.

The time it takes to solve (4)-(9) often limits which switched linear systems can realistically be controlled with MPC. Most solvers are iteration based, and thus not suitable for parallel implementations. Furthermore, current efforts typically do not take the recursive nature of MPC into account.

3. MEMORY-BASED DISCRETE SEARCH

Instead of solving a slow mixed-integer problem at each timestep, we suggest to search over the possible sequences of switching signals based on their previous performance and only solve the MPC problem for the chosen sequences. We call the method MPC with Memory-based Discrete Search (MDS). For given switching signal sequences, the optimization problem corresponding to each sequence is continuous in the variables, and hence solvable by fast, convex optimization solvers. When the set of switching signals, for which the MPC problem should be solved, depends on the method’s memory of previous performances, the optimizations can be performed in parallel to decrease the computation time further.

MPC with MDS searches for good switching signal sequences based on previous performance. Since the controlled system’s state changes dynamically, the expected performance of a given sequence will depend on the previous performance of sequences that are time-shifted and similar, not identical to that sequence. MPC with MDS is thus very suitable for systems that require long prediction horizons, as more information about a sequence’s potential performance can be gathered before the dynamic information becomes obsolete.

To choose for which set of switching signal sequences to solve the MPC problem, MPC with MDS uses ideas from Bayesian optimization to ensure the set contains both sequences that are expected to perform well, and sequences that will provide new information. To measure the performance and information value of a switching signal sequence, the sequence’s performance $F_{t}(\varsigma_{[0,N-1]})$, expected fitness $F_{t}(\varsigma_{[0,N-1]})$ and uncertainty value $Y_{t}(\varsigma_{[0,N-1]})$ are introduced.

Definition 2. The fitness of switching signal sequence $\varsigma_{[0,N-1]} = \{\varsigma_{0},...,\varsigma_{N-1}\}$ is

$$F_{t}(\varsigma_{[0,N-1]}) = \min_{u_{[1,\ldots,N-1]}} J(\varsigma_{[0,N-1]}, u_{[1,\ldots,N-1]}) \quad (10)$$

s.t.

$$x_{0} = x_{t} \quad (11)$$

$$x_{k+1} = A_{x}x_{k} + B_{x}u_{k} \quad \forall k \in [0,N-1] \quad (12)$$

$$x_{k} \in X_{x}, u_{k} \in U_{u}, \quad \forall k \in [0,N-1] \quad (13)$$

$$x_{N} \in X_{f} \quad (14)$$

where the notation corresponds to that of (4)-(9).

Definition 3. The expected fitness for a sequence $\varsigma$ at time $t$ is denoted by $F_{t}(\varsigma)$ and its uncertainty value is denoted by $Y_{t}(\varsigma)$.

Figure 1 presents a schematics of MPC with MDS. The detailed notation will be provided in the remainder of this section. At each timestep $t$, fitness estimates of all switching signal sequences are used to determine a set of promising sequences. For each promising sequence, the fitness is evaluated for the current system state. The best performance of the promising sequences is then compared to the performance of a known controller. This controller knows a switching signal sequence and a corresponding continuous input sequence which satisfy all constraints and stabilizes the system. The first switching signal and continuous input corresponding to the sequence that performs the best are implemented on the system and the process starts over at the next timestep.

How the promising sequences are chosen does not affect the stability of a system controlled by MPC with MDS. Therefore, the stability and recursive feasibility of the proposed method are proven in Section 3.1 before the selection method is presented in Section 3.2.

3.1 Stability and feasibility

To guarantee stability and recursive feasibility of the MPC problem, MPC with MDS compares the performance of the potential sequences with the performance of a control law, which is known to satisfy all constraints. This section outlines how this is done, and proves that the implemented inputs will stabilize the system.

The algorithm knows a feasible solution at time $t$, but needs to access the performance of the potential sequences at time $t+1$. It is thus necessary to define how the solution at time $t$ is shifted in time and to prove that the shifted solution is indeed feasible.

Definition 4. The sequences $\varsigma_{[0,N-1]}^{+} = \{\varsigma_{[1,N-1]}, \gamma\}, u_{[0,N-1]}^{+} = \{u_{[1,N-1]}, \kappa(x_{N})\}$ and $x_{[0,N]}^{+} = \{x_{[1,N]}, A_{x}x_{N} + B_{x}\kappa(x_{N})\}$ are said to be the shifted sequences of $\varsigma_{[0,N-1]}, u_{[0,N-1]}$ and $x_{[0,N]}$. 
The outer framework of MPC with MDS can be seen in Algorithm 1. The method is initialized with a feasible solution to the mixed-integer problem (4)-(9) for the initial state. In the algorithm the best solution that is known to the controller at a given time is marked with a hat. The first elements of the best known switching signal sequence and continuous input sequence are implemented as is usually done with MPC. Hereafter MPC with MDS decides on the set of potential sequences. Before

The outer framework of MPC with MDS can be seen in Algorithm 1. The method is initialized with a feasible solution to the mixed-integer problem (4)-(9) for the initial state. In the algorithm the best solution that is known to the controller at a given time is marked with a hat. The first elements of the best known switching signal sequence and continuous input sequence are implemented as is usually done with MPC. Hereafter MPC with MDS decides on the set of potential sequences. Before

Algorithm 2 Deciding \( S(t) \)

1: for all \( \varsigma \in \Omega \) do
2: \( \tilde{F}_t(\varsigma) = \alpha \tilde{F}_{t-1}(\varsigma) + \frac{1-\alpha}{\varsigma} \sum_{\psi \in N_\varsigma} \tilde{F}_{t-1}(\psi) \)
3: \( \tilde{Y}_t(\varsigma) = \alpha \tilde{Y}_{t-1}(\varsigma) + \frac{1-\alpha}{\varsigma} \sum_{\psi \in N_\varsigma} \tilde{Y}_{t-1}(\psi) \)
4: end for
5: \( S(t) = \emptyset \)
6: while \( S(t) \) is not full do
7: \( S(t) = S(t) \cup \arg \min_{\varsigma \in \Omega \setminus S(t)} \tilde{F}_t(\varsigma) \)
8: \( S(t) = S(t) \cup \arg \min_{\varsigma \in \Omega \setminus S(t)} \tilde{Y}_t(\varsigma) \)
9: \( S(t) = S(t) \cup \arg \max_{\varsigma \in \Omega \setminus S(t)} \tilde{Y}_t(\varsigma) \)
10: \( S(t) = S(t) \cup \{ \psi \} \) where \( \psi \in \Omega \setminus S(t) \)
11: end while
12: return \( S(t) \)

3.2 Sequence selection

Several methods can be used to select the set of potential sequences. MPC with MDS estimates, like in Bayesian optimization, whether a sequence is likely to be either the best solution (exploitation) or bring new information (exploration). The set of potential sequences is assembled to reflect a trade-off between exploitation and exploration. In the following it is assumed that \( N \) and size \( l \) of \( S \) are sufficiently small, that saving and sorting information on each \( l \) switching signal is realistic. If \( l \) and \( N \) are large, sampling methods and parallelization can be used to represent the switching signals.

When the fitness and uncertainty value of a sequence is estimated, only information from the previous timestep is used. It is thus necessary to establish which switching signal sequences from the previous time \( t - 1 \) will have influence at a given sequence at time \( t \). For this, we define neighbour sequences and implemented ancestors.

Definition 5. Two switching signal sequences \( \varsigma^a \) and \( \varsigma^b \) are neighbours, if they differ at only one timestep except for the last timestep, that is \( \varsigma^a \in \{ \varsigma^i : \varsigma^i_{j} \neq \varsigma^b_{j} \ \forall \ i \in (0,N-2) \setminus \{ j \} \}, \) \( j \in (0,N-2) \). Switching signal sequence \( \varsigma^a \) has \( \varsigma^a_{0,}\varsigma^a_{N-1} \) neighbours.

Definition 6. A switching sequence \( \varsigma^a \)’s implemented ancestor \( a(\varsigma) \) at time \( t \) is a switching sequence whose first element is applied to the system at time \( t - 1 \), and whose remaining elements are identical with the first \( N - 2 \) elements in the sequence in question, i.e. switching signal sequence \( \varsigma^a \) has implemented ancestor \( a(\varsigma) = \{ \varsigma_{\sigma_{t-1} = \psi_0} \cup \varsigma_{[1,N-1]} = \varsigma_{0,N-2} \} \).

A switching sequence’s neighbours can be precomputed and does not vary over time, while the implemented ancestor will vary depending on the switching signal implemented at the previous timestep. However, a set of potential ancestors of a switching signal sequence can be precomputed and the correct implemented ancestor can be found online. This reduces computation time.

Algorithm 2 shows how MPC with MDS computes expected fitness and uncertainty value for the switching signal sequences and uses simple optimization over these values to decide the set of potential sequences. Before
Algorithm 1 is started, $\tilde{F}_t(\varsigma)$ and $\tilde{Y}_t(\varsigma)$ must be given initial values. We recommend a positive factor times the optimal function value of the initially known solution, to ensure correct scaling.

The expected fitness $\tilde{F}_t(\varsigma)$ and uncertainty value $\tilde{Y}_t(\varsigma)$ are updated for all sequences in $\Omega$ at all times $t$. $\Omega$ contains all possible switching signal sequences, even those for which (10)-(14) may be infeasible at time $t$. When the values are updated, the impact of the implemented ancestor is weighted to the impact of the average of the neighbour sequences by $0 < \alpha < 1$. Notice that the expected fitness and uncertainty value could be easily computed in parallel. The selection of the set of potential sequences could also be parallelized, if a centralized step ensures the uniqueness of each potential sequence in the set by replacing duplicates with random sequences. This will lead to increased exploration.

4. IMPLEMENTATION EXAMPLE

To illustrate the performance of the proposed method, simulation experiments are conducted where MPC with MBS has been implemented on a small switched linear system and the performance of MPC with MBS is compared to three benchmark controllers. All controllers are implemented in serial, since the authors do not have access to parallel computing yet. MPC with MBS and the Random controller are however prepared for parallel implementation as mentioned in the end of Section 3. The experiments are implemented in Matlab using Yalmip, Löfberg (2004), and Gurobi.

MPC with MDS: follows Algorithm 1 and 2 with the selection of the potential sequences prepared for parallelization as mentioned in the end of Section 3.2. (10)-(14) is solved using Gurobi’s barrier method limited to 150 iterations. The experiment is repeated 10 times to illustrate the effect of the random selections used in Algorithm 2.

Optimal controller: solves the optimal mixed-integer MPC problem (4)-(9) at each time $t$. To obtain a convex continuous approximation of the mixed-integer program, the switching signals are represented by binary variables.

Safe controller: solves (4)-(9) for the system’s initial state and uses the solution as switching signal and continuous input the first 9 timesteps. Hereafter the basic switching signal and the Safe control law is implemented, i.e. $\sigma_t = \gamma$ and $u_t = \kappa(x_t) \forall t \geq 10$.

Random controller: selects the set of potential sequences randomly instead of using Algorithm 2 but follows otherwise Algorithm 1. (10)-(14) is solved using Gurobi’s barrier method with a limitation of 150 iterations. The experiment is repeated 10 times.

4.2 System

The system used to illustrate the performance has initial state $x_0 = [-7 -2]_T$ and dynamics

$$x_{t+1} = Ax_t + Bu_t \quad \text{where } \sigma \in [0,1,2,3],$$

$$A_0 = A_2 = \begin{bmatrix} 0.9 & 0.1 \\ 0 & 1.1 \end{bmatrix}, \quad B_0 = B_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 0.1 \\ 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 1 & 0.3 \\ 0 & 1 \end{bmatrix} \text{ and } B_3 = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix}.$$  

It is known that the unconstrained system is stable when $\sigma_t = 0$ and $u_t = [-0.1 -0.2] x_t$ for all $t$.

The system is subject to state constraint $H_s x_t \leq h_s \forall t$, and input constraints $H_\sigma x_t \leq h_\sigma \forall t$, with

$$H_s = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix}, \quad h_s = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \quad H_3 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad h_3 = \begin{bmatrix} 3 \\ -1 \end{bmatrix}.$$  

$$H_0 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad h_0 = \begin{bmatrix} 0.25 \\ 0.25 \end{bmatrix}, \quad H_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad h_1 = \begin{bmatrix} 2 \\ -1 \end{bmatrix}.$$  

$$H_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad h_2 = \begin{bmatrix} 5 \\ -0.05 \end{bmatrix}.$$  

Notice that the continuous input under switching signal 2 is bounded away from zero and that the continuous input under switching signal 4 has a different dimension.

Between timestep $t = 50$ and $t = 100$ the system is subject to a constant disturbance

$$x_{t+1} = Ax_t + Bu_t \kappa(x_t) - \begin{bmatrix} 0.1 \\ 0 \end{bmatrix} \forall 50 \leq t \leq 98. \quad (15)$$

The controllers have no information about this disturbance, besides the measured state, and thus do not guarantee state constraint satisfaction. The disturbance is however so small that the Optimal controller remains feasible.

Both MPC with MDS, the Optimal and the Random controller are implemented with prediction horizon $N = 10$ and the following cost

$$J(\varsigma, u, x) = \sum_{k=1}^{N-1} (x_k^T Q x_k + u_k^T R x_k u_k) + x_N^T Q_N x_N, \quad (16)$$

where $R_0 = 1$, $R_1 = R_2 = 0.001$, $R_3 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0 \end{bmatrix}$, $Q = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$ and $Q_N = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. The terminal set is

$$X_f = \{x \mid x_N^T Q_N x_N \leq 0.2\}.$$ Satisfaction of Assumption 1 part 1) and 2) is immediately clear, and satisfaction of Assumption 1 part 3) and 4) and Assumption 2 can be shown using the S procedure.

4.3 Results

The four controllers performed as expected in the conducted experiments. In Figure 2 the realized cost, $x_t^T Q x_t + u_t^T R x_t u_t$, is shown accumulated over time. As expected the Optimal controller is over time the cheapest controller, while the Safe controller is the most expensive. Furthermore, the Optimal controller only accumulates a
little more cost when the system is disturbed between $t = 50$ and $t = 100$ while the Safe controller is costly. MPC with MDS performs well compared to both Safe and Random controllers. MPC with MDS performs significantly better than the Safe controller when the system is disturbed. The safe input was chosen by MPC with MDS mainly when the system was close to the reference value. During the disturbance ($50 \leq t \leq 98$), a better solution was found at 94.4% of the timesteps.

In the experiment, the maximum computation time for one iteration of the Optimal controller was 0.6032 s. MPC with MDS was implemented in serial, with a maximum computation time of 1.6575 s. However, the strength of MPC with MDS is its parallelizability. Computing the parts of the algorithm that must be serial took maximum 0.0189 s and the longest time it took to solve (10)-(14) was 0.3669 s. Computing Algorithm 2 in serial is time consuming due to the large amount of data. Updating the estimated fitness and uncertainty value took 0.4389 s in serial. The computation for each switching signal sequence is independent, and therefore easily parallelizable. It is expected that a parallel implementation of MPC with MDS reaches computation times comparable to that of the Optimal controller on this small system.

5. CONCLUSIONS AND FUTURE WORK

We have shown how ideas from Bayesian optimization can be used to parallelize the computations performed by MPC on switched linear systems. The presented method, MPC with Memory-based Discrete search (MDS) guarantees recursive feasibility and stability of the system. It finds ‘good enough’ inputs fast. The solution is not optimal, but the performance improvements compared to implementing a known safe input makes it an interesting method for many applications. It is expected that stability of MPC with MDS can be established for a broader class of systems, which is a promising future research.

In MPC with MDS, the computational complexity is decreased by reducing the mixed-integer MPC problem to a prespecified number of convex optimization problems. The results indicate that this decreases computation times, if the algorithm is implemented in parallel. How many parallel agents and how large a set of potential sequences to choose will be highly problem dependent and is another interesting line of future research. It is expected that, with sufficient parallelization, the computation time will increase less than that of a mixed-integer MPC when the system size increases.

REFERENCES


