Distributed Multiple Shooting for Optimal Control of Large Interconnected Systems

Attila Kozma∗ Joel Andersson∗ Carlo Savorgnan∗ Moritz Diehl∗

∗ ESAT-SCD / Optimization in Engineering Center (OPTEC), KU Louvain, Louvain, Belgium
(e-mail: {attila.kozma, joel.andersson, carlo.savorgnan, moritz.diehl}@esat.kuleuven.be)

Abstract: Large interconnected systems consist of a multitude of subsystems with their own dynamics, but coupled with each other via input-output connections. Each subsystem is typically modelled by ordinary differential equations or differential-algebraic equations. Simulation and optimal control of such systems pose a challenge both with respect to CPU time and memory requirements. We address optimal control of such systems by applying “distributed multiple shooting”, a generalization of the direct multiple shooting method, which uses the decomposable structure of the system in order to obtain a highly parallel algorithm. The interconnections are allowed to be infeasible during the iterations but are driven to feasibility by a Newton-type optimization algorithm. We evaluate the performance of the distributed multiple shooting method on a large scale estimation problem.

Keywords: large-scale systems, optimal control, parallel algorithms, dynamic decoupling, decomposition methods

1. INTRODUCTION

In this paper, we employ a highly parallel algorithm called distributed multiple shooting (DMS), originally published in (Savorgnan et al., 2011), to solve large-scale optimal control problems of interconnected systems such as manufacturing systems, process plants and networked systems. In the DMS approach, we assume that the behaviour of each subsystem may be described by a differentiable function that is expensive to evaluate and to linearize. The DMS method allows for parallelization in two different levels. First, with domain decomposition methods (Smith et al., 1996) we split the spatial domain of a boundary value problem into spatial subdomains, and on the subdomains we solve smaller boundary value problems and iterate to coordinate the solution between adjacent subdomains. The problems on the subdomains are independent, which makes domain decomposition methods suitable for parallelization. The connection between the subdomains are approximated with orthogonal polynomials. Second, since differential equations are considered, another way of decomposition is introduced by slicing the time domain into subintervals, on which less complex problems are solved simultaneously. In this decomposition method, which is known as direct multiple shooting (Bock and Plitt, 1984), we also need to ensure the match between boundaries e.g. by a Newton-type optimization method.

The paper is organized as follows. In Section 2, the problem considered is presented and a detailed description of distributed multiple shooting is given. Then, in Section 3, numerical experiments are presented to show the performance of DMS. We solve an ill-posed, but regularized nonlinear parameter-estimation problem having, after discretization, 15342 optimization variables and very costly constraint functions (calls to ODE integrators). In Section 4, some remarks on the software implementation are given, and we conclude the paper in Section 5.

2. DISTRIBUTED MULTIPLE SHOOTING

In this section, we explain the main idea of DMS and illustrate its main advantages with respect to traditional approaches. We regard the optimal control problem of a large interconnected system:

\[
\min_{x(·),u(·),z(·)} \sum_{i=1}^{M} \int_{0}^{T} \ell^{i}(x^{i}(t),u^{i}(t),z^{i}(t))dt \quad (1a)
\]

s.t. \[ \dot{x}^{i}(t) = f^{i}(x^{i}(t),u^{i}(t),z^{i}(t)) \quad (1b) \]

\[ y^{i}(t) = g^{i}(x^{i}(t),u^{i}(t),z^{i}(t)) \quad (1c) \]

\[ x^{i}(0) = \bar{x}^{i}_{0} \quad (1d) \]

\[ z^{i}(t) = \sum_{j=1}^{M} A_{ij} y^{j}(t) \quad (1e) \]

\[ p^{i}(x^{i}(t),u^{i}(t)) \geq 0, \quad t \in [0,T]. \quad (1f) \]

Here, \[ x^{i}(t) \in \mathbb{R}^{n_{i}}, u^{i}(t) \in \mathbb{R}^{u_{i}}, z^{i}(t) \in \mathbb{R}^{n_{i}} \] and \[ y^{i}(t) \in \mathbb{R}^{n_{i}} \] represent the state, the control input, the coupling input and the output of subsystem \( i \), respectively. The terminal time \( T \) and \( \tau^{i}_{0} \) are fixed.

Objective (1a) is composed of the stage costs of the subsystems, constraints (1b)-(1d) characterize the nonlinear dynamic behaviour of subsystem \( i \), while constraint (1e) ensures the linear coupling between the subsystems. Path constraints are imposed on \( z(t) \) and \( u(t) \) by (1f). We
discriminate between control and coupling inputs because they have a different role in our setting; the former can be used to control the system, while the latter take dynamic couplings between the subsystems into account. In the simplest case, one can consider a subsystem that depends directly on the states of another subsystem resulting in input-output interaction.

The numerical solution of this infinite dimensional problem is addressed in two steps: in the first step, the problem is discretized in order to obtain a finite dimensional nonlinear programming problem (NLP), in the second step, the NLP is solved using sequential quadratic programming (SQP).

2.1 Control input discretization

We define the time points
\[ 0 = t_0 < t_1 < \cdots < t_N = T. \]  
which divide the interval \([0, T]\) into \(N\) sub-intervals, also called shooting intervals. The inputs are discretized using a finite parametrization on the time sub-intervals. For simplicity, in this paper we will use a piecewise constant parametrization:
\[ u^i(t) = u^i_n \quad \forall t \in [t_n, t_{n+1}), \quad n = 0, \ldots, N - 1. \]  

2.2 Coupling discretization

To be able to integrate the dynamics of the subsystems independently, the vectors \(x^i(t)\) and \(y^i(t)\) are represented as a linear combination of a basis function. Although other choices are possible, we restrict our attention to a basis composed by normalized Legendre polynomials \(\gamma_q(t)\), which are defined by the relation
\[ \int_{-1}^{1} \gamma_q(t) \gamma_m(t) dt = \begin{cases} 0 & \text{if } q \neq m \\ 1 & \text{otherwise}. \end{cases} \]  

We define the vector of normalized Legendre polynomials up to degree \(S\)
\[ \Gamma(t) = [\gamma_0(t) \gamma_1(t) \cdots \gamma_S(t)]^T. \]

For \(t \in [t_n, t_{n+1})\) and \(p \in \{1, \ldots, u^i_n\}\), we introduce the following approximation
\[ z^i_p(t) = \Gamma_n(t)^T z^i_{n,p}, \]
where \(z^i_{n,p} \in \mathbb{R}^{S+1}\) is a coefficient vector and
\[ \Gamma_n(t) = \Gamma \left( \frac{t - t_n}{t_{n+1} - t_n} - 1 \right). \]

For notational convenience, we define the matrix
\[ Z^i_n = \begin{bmatrix} z^i_{n,1} & \cdots & z^i_{n, u^i_n} \end{bmatrix} \]

such that \(z^i(t) = (Z^i_n)^T \Gamma_n(t)\). To find the coefficient matrix \(y^i_n\) used to represent \(y^i(t)\), we have to solve the following unconstrained quadratic optimization problem
\[ y^i_{n,q} = \arg \min_{y \in \mathbb{R}^{S+1}} \int_{t_n}^{t_{n+1}} (\Gamma_n(t)^T y - y^i_q(t))^2 dt. \]

Due to the orthogonality of \(\Gamma_n(t)\), (9) can be solved performing the following integration
\[ y^i_n = \frac{2}{t_{n+1} - t_n} \int_{t_n}^{t_{n+1}} \Gamma_n(t) (y^i(t))^T dt. \]

Note that the integrand on the right-hand side is a matrix and the integration should be carried out element-wise.

2.3 State discretization

We denote by \(x^i_n\) the value of \(x^i(t_n)\). Given \(x^i_n, u^i_n, z^i_n\), we can simulate the \(i\)-th subsystem on the \(n\)-th interval. Using the corresponding solution \(x^i_n(t), t \in [t_n, t_{n+1}]\), we define the functions \(F^i_n, L^i_n, G^i_n\) as
\[ F^i_n(x^i_n, u^i_n, z^i_n) = x^i_n(t_{n+1}) \]  
\[ L^i_n(x^i_n, u^i_n, z^i_n) = \int_{t_n}^{t_{n+1}} \ell^i(x^i_n(t), u^i_n(t), \Gamma(t)^T z^i_n) dt \]  
\[ G^i_n(x^i_n, u^i_n, z^i_n) = \frac{2}{t_{n+1} - t_n} \int_{t_n}^{t_{n+1}} \Gamma_n(t) (y^i_n(t))^T dt. \]

It should be understood that \(F^i_n, L^i_n, G^i_n\) are defined up to the approximation error introduced by the discretization of the coupling input.

2.4 NLP formulation

Distributed multiple shooting solves the following NLP
\[
\min_{x^i_n, z^i_n, y^i_n} \sum_{n=0}^{N-1} \left( \sum_{i=1}^{M} L^i_n(x^i_n, u^i_n, z^i_n) \right) \tag{12a}
\]
\[ \text{s.t.} \quad x^i_{n+1} = F^i_n(x^i_n, u^i_n, z^i_n) \quad n = 0, \ldots, N - 1 \tag{12b} \]
\[ y^i_n = G^i_n(x^i_n, u^i_n, z^i_n) \quad n = 0, \ldots, N - 1 \tag{12c} \]
\[ x^i_0 = x^i_0 \tag{12d} \]
\[ z^i_n = \sum_{j=1}^{M} A_{ij} y^j_n \quad n = 0, \ldots, N - 1 \tag{12e} \]
\[ p^i(x^i_n, u^i_n) \geq 0 \quad n = 0, \ldots, N - 1. \tag{12f} \]

Note that the evaluation and linearization of function \(F^i_n\) and function \(G^i_n\) are \(N \times M\) independent tasks that may be performed simultaneously. It should be also noted that this problem formulation incorporates direct multiple shooting as a special case if \(M = 1\). We summarize problem (12) as a nonlinear programming problem of \(Q = N \times M\) subsystems in the form of
\[
\min_{v_k, u_k, \phi_k} \sum_{k=1}^{Q} f_k(v_k, u_k) \tag{13a}
\]
\[ \text{s.t.} \quad w_k = \phi_k(v_k, u_k) \quad k = 1, \ldots, Q \tag{13b} \]
\[ v_k = \sum_{j=1}^{Q} A_{ij} w_j \quad k = 1, \ldots, Q \tag{13c} \]
\[ (u_k, v_k) \in \Omega, \quad k = 1, \ldots, Q. \tag{13d} \]

Here, the main source of difficulties is constraint (13b), which represents an ODE solver with sensitivity generation capabilities.

Also note that optimization problem (12) corresponds to (13) with variables \(k := nN + i, \quad u_k := u^i_n, \quad v_k := (y^i_n)^T, \quad w_k := \frac{2}{t_{n+1} - t_n} \int_{t_n}^{t_{n+1}} \Gamma_n(t) (y^i_n(t))^T dt. \tag{14} \)

The most important benefit of DMS is the possibility to parallelize the evaluation and linearization of \(\phi_i\), as they are the most time-consuming computations that we need for the solution of the NLP.
2.5 Sequential Quadratic Programming

We employ a sequential quadratic programming (SQP) method (Powell, 1978) to solve problem (12), which we summarize as

\[
\min_{x} f(x) \quad (15a)
\]
\[
s.t. \ g(x) = 0 \quad (15b)
\]
\[
h(x) \geq 0. \quad (15c)
\]

Given the actual iterate \(x_k\), the linearization of (15) reads:

\[
\min_{\Delta x_k} \frac{1}{2} \Delta x_k^T B_k \Delta x_k + \nabla f(x_k)^T \Delta x_k \quad (16a)
\]
\[
g(x_k) + \nabla g(x_k)^T \Delta x_k = 0 \quad (16b)
\]
\[
h(x_k) + \nabla h(x_k)^T \Delta x_k \geq 0, \quad (16c)
\]

where \(\Delta x_k = x_{k+1} - x_k, \ B_k \approx \nabla^2 f(x_k) \) with Lagrangian function \(\mathcal{L}(x, \lambda, \mu) = f(x) - \lambda^T g(x) - \mu^T h(x)\). Note that the quadratic programming (QP) problem (16) has a well-defined sparsity structure that has to be exploited within the linear solver. In particular, the Hessian \(B_k\) is block diagonal with blocks corresponding to variables \(x_k\) and \(v_k\) only, without any crossterms. The computation of \(B_k\) may be carried out in different ways, e.g., using exact Hessian, BFGS-update or a Gauss-Newton approximation. In this paper, we employ the latter. Once we obtain the optimal solution \(\Delta x_k^*\), we apply a linear correction to the original variables

\[
x_{k+1} = x_k + t_k \Delta x_k, \quad (17)
\]

with step length \(t_k\). The procedure of linearization and correction continues until reaching convergence. The local convergence of different SQP methods was proven already in (Robinson, 1974). In essence, locally quadratic convergence may be achieved if the first and second derivatives are calculated exactly. The method can be globalized by employing a line-search strategy.

3. NUMERICAL EXPERIMENTS

In this paper, an estimation problem with partial differential equation (PDE) constraints is considered. Discretizing such problems by the method of lines often results in a large set of ordinary differential equations (ODE) that is well suited for DMS algorithm.

3.1 Smoke detection

We consider a two-dimensional model of \(M = 2\) rooms connected by a door as illustrated in Figure 1. In one of the rooms, an object in a fixed but unknown position emits smoke for a fixed but unknown period of time. At every space point \(x\), the evolution of the smoke concentration \(\psi(t, x)\) is described by

\[
\frac{\partial \psi}{\partial t} = D \Delta \psi - v^T \nabla \psi + u(t, x) + h(\psi) \quad (18)
\]

where \(D\) is the diffusion coefficient, \(v^T\) is a source-free flow, whereas \(h(\cdot)\) is a nonlinear function acting on the concentration. The term \(u(t, x)\) denotes the smoke that enters the system at time \(t\) in space coordinate \(x\). By employing the method of lines, using central differences for the diffusion terms and a first order upwind approximation for the convection terms, we can discretize this equation into an interconnected system of two ODEs, which can be summarized as

\[
\dot{x}^1(t) = f^1(x^1(t), u^1(t)) \quad (19a)
\]
\[
y^1(t) = x^1 C(t) \quad (19b)
\]
\[
z^2(t) = z^2(t) \quad (19c)
\]

where \(x^1 C(t)\) denotes the states of the first room right in front of the door. Note that we neglect the diffusion coming from subsystem 2 to subsystem 1, since there is constant airflow going through the door, thus \(f_1\) does not depend on \(z_2(t)\). If we consider rooms having \(K \times L\) finite volumes, each subsystem has \(K \cdot L\) state variables \(x^i\) and the same amount of variables \(u^i\). We discretize the problem using the approach described in Section 2 resulting in

\[
x^1_{n+1} = F^1_n(x^1_n, u^1_n) \quad (20a)
\]
\[
y^1_n = G^1_n(x^1_{n}) \quad (20b)
\]
\[
x^2_{n+1} = F^2_n(x^1_n, u^2_n, z^2_n) \quad (20c)
\]
\[
z^2_n = y^1_n, \quad (20d)
\]

where \(n\) denotes the index of the shooting interval.

3.2 Estimation problem

Given the model (19) with zero smoke concentration at the initial time, we place a set of sensors to certain positions (see Figure 1) in each room described by the measurement matrix \(C\). The sensors measure the smoke concentration \(\eta^i_n := C x^i_n + \epsilon^i_n\) at time step \(n\) and subsystem \(i\), where \(\epsilon^i_n\) denotes the measurement noise. Our goal is to localize the smoke sources \(u^i_n\) in time and space. Given the measurements, we introduce the inverse problem that we want to solve

\[
\min_{x^1, \eta^i} \sum_{n=0}^{N-1} \sum_{i=1}^{2} \frac{1}{2} \left( || C x^i_n - \eta^i_n ||^2 + \gamma || u^i_n ||_1 \right), \quad (21)
\]

s.t. (20).

Note that the separable cost function is composed by a number of L1 and L2 terms and that the number of measurements is much smaller than the number of finite volumes and thus the problem formulation would be ill-posed without regularization. The L2 term penalizes the deviation of states \(C x^i_n\) from the corresponding measurements. We suppose that smoke originates from only a small number of finite cells, hence the L1-term, which yields
sparse solutions in $u^*_t$, is a suitable regularization method. In our experiments, we use a grid of $19 \times 19 = 361$ finite volumes and place 16 sensors in each room. The measurements $u^*_t$ were generated by simulation, to which 1% Gaussian noise was added. The smoke was emitted from position $(0,1)$ for $t \in [2, 4)$. The 3 coupling variables were approximated by polynomials of order two on each time interval. The evolution of the smoke in time is depicted in Figure 2.

![Smoke distribution at different times](image)

Fig. 2. Smoke distribution at different times: $t = 4s$ (top), $t = 9s$ (bottom)

3.3 Results

We solved the nonlinear estimation problem (21) on a time interval of $[0,9]$ with ten time intervals. To have a comparison, we have measured the runtime of one SQP iteration using a) single shooting (SS), b) direct multiple shooting (MS) and c) distributed multiple shooting (DMS). In an SQP iteration, there are two expensive operations that we have to consider: the sensitivity calculation by forward differentiation and the centralized, but parallelized, QP solution. If we employ the SS algorithm the sensitivity calculation requires 452.4s on the full horizon of $t \in [0,9]$. Linearizing the system on the time interval $t \in [0,1]$ took 79.8s with the parallel MS method. When the DMS algorithm was used, sensitivity generation of subsystem 1 required 15.16s, while subsystem 2 needed 15.7s of computation time. If we compare the fully serial SS approach with the fully distributed DMS algorithm, we observed that the latter is about 29 times faster in sensitivity calculation (see Table 1).

One of the advantages of DMS is that we can save runtime already in a serial implementation due to two reasons. First, the independent integrators can adapt their stepsize to their own system and thus fast subsystems do not have a negative effect on other subsystems with slower and thus more easily integrable dynamics. Second, the Jacobians of decomposed subsystems may have less columns altogether compared to the centralized one, since the number of coupling variables is typically much smaller than the number of "foreign" states. This results in a smaller number of forward derivatives. For instance, if we consider the smoke detection problem, in a centralized setting the states of the second room depend on all of the states of the first room. In DMS, this dependency is reduced to a set polynomial coefficient vectors that have altogether much less dimension than the one of the state space of the first room.

We have compared the runtime of the QP solution within the SQP method, which was 3.4s with SS, 35.8 with MS, 23.8s once we used DMS (see Table 1). It should be emphasized that in all cases a parallel QP solver was used with 16 threads. The difference between MS and DMS may be explained by the fact that even though in DMS we have more optimization variables, the resulting QP is sparser than the one in MS allowing for faster linear algebra operations.

<table>
<thead>
<tr>
<th>Method</th>
<th>Sens.</th>
<th>QP sol.</th>
<th># opt. vars.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SS</td>
<td>452.4s</td>
<td>3.4s</td>
<td>8664</td>
</tr>
<tr>
<td>MS</td>
<td>79.8s</td>
<td>35.8s</td>
<td>15162</td>
</tr>
<tr>
<td>DMS</td>
<td>15.7s</td>
<td>23.8s</td>
<td>15342</td>
</tr>
</tbody>
</table>

Table 1. Runtime of sensitivity generation and QP solution with single shooting (SS), parallel direct multiple shooting (MS) and distributed multiple shooting (DMS), together with problem dimensions.

In the measurements that were generated, the smoke was injected during the time period $[2,4)$ in position $(0,1)$. After four full step SQP iterations the DMS method converged and localized the smoke source perfectly in the time period $[2,3)$, while in the interval $[3, 4)$ the detected source is slightly above the original source (see Figure 3). It can be also seen that in position $(0, 6)$ another source was detected, although with smaller intensity compared to the others. This behaviour is a feature of the considered NLP and not a side-effect of the DMS algorithm.

4. REMARKS ON THE IMPLEMENTATION

The software implementation needs to carry out the following tasks: symbolic computations, ODE solution, message passing and QP solution. Symbolic computations are necessary to evaluate derivatives of functions and to generate sensitivity equations of ODEs. For this purpose, we used the optimization framework CasADi (Andersson et al., 2012). To solve ODEs and calculate their sensitivities, we have employed the open-source software CVODES (Serban and Hindmarsh, 2005) from the SUNDIALS suite via the interface available in CasADi. The communication between the parallel processes was performed by OpenMPI (Gabriel et al., 2004) and its python interface mpi4py. The QP subproblems in the SQP method were solved by the parallel sparsity exploiting barrier solver of CPLEX (IBM.
5. CONCLUSIONS

In this paper, we discussed and applied the distributed multiple shooting algorithm, which allows for high parallelizability by splitting the spatial domain as well as the time domain of optimal control problems. The method is designed to be used for large-scale interconnected systems, for which the centralized solution is not sufficiently fast or is not implementable due to memory limitations. The feasibility gaps between the subdomains are closed by a fast Newton-type optimization method with a sparsity exploiting QP solver.

We have seen that distributed multiple shooting offers advantages already in a serial implementation due to less directional derivatives and locally adjusted stepsizes within the integrators. We have shown via an academic example that DMS in a parallel implementation may perform 29 times better than single shooting and 5 times better than a parallel direct multiple shooting.

It has also become clear that by distributing the integration in the time and space domains the QP solution phase becomes the bottleneck in DMS. Methodologies to solve QPs arising in DMS efficiently in a distributed manner are the subject of ongoing and future research.

ACKNOWLEDGEMENTS

Research Council KUL: GOA/11/05 Ambiorics, GOA/10/09 MaNet, CoE EF/05/006 Optimization in Engineering (OPTEC) en PFV/10/002 (OPTEC), IFOSCORES4CHEM, several PhD/postdoc & fellow grants; Flemish Government: FWO: PhD/postdoc grants, projects: G0226.06 (cooperative systems and optimization), G0321.06 (Tensors), G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) research communities (WOG: ICCoS, ANMMM, MLDM); G.0377.09 (Mechatronics MPC) IWT: PhD Grants, Eureka-Flite+, SBO LeCoPro, SBO Climaqs, SBO POM, O&O-Dsquare
Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011);
EU: ERSNI; FP7-HD-MPC (INFSO-ICT-223854), COST intelliCIS, FP7-EMBOCON (ICT-248940), FP7-SADCO (MC ITN-264735), ERC HIGHWIND (259 166) Contract Research: AMINAL Helmholtz: viCERP ACCM

REFERENCES