Parallel Solution of Large-Scale Dynamic Optimization Problems

Laird, Carl; Wong, Angelica; Åkesson, Johan

2011

21st European Symposium on Computer Aided Process Engineering — ESCAPE 21
E.N. Pistikopoulos, M.C. Georgiadis and A.C. Kokossis (Editors)

Parallel Solution of Large-Scale Dynamic Optimization Problems

Carl D. Laird[a1], Angelica V. Wong[a1], Johan Akesson[c2]

[a]*Artie McFerrin Department of Chemical Engineering, Texas A&M University, TX, USA*
[c]*Department of Automatic Control, Lund University, Sweden*

Abstract

This paper presents a decomposition strategy applicable to DAE constrained optimization problems. A common solution method for such problems is to apply a direct transcription method and to solve the resulting non-linear program using an interior point algorithm, where the time to solve the linearized KKT system at each iteration is dominating the total solution time. In the proposed method, the structure of the KKT system resulting from a direct collocation scheme for approximating the DAE constraint is exploited in order to distribute the required linear algebra operations on multiple processors. A prototype implementation applied to benchmark models shows promising results.

Keywords: dynamic optimization, parallel computing, collocation

1. Introduction

Optimization of dynamic systems has proven to be an effective method for improving operation and profits in the chemical process industry. The size and complexity of optimization problems continue to grow, while the advances in computing clock rates that we once took for granted have slowed dramatically. Computer chip design companies have instead focused on development of parallel computing architetures, and there is a need for the development of advanced parallel algorithms for dynamic optimization that can utilize these architectures. Furthermore, the successful use of advanced solution approaches within industrial settings requires that these algorithms are interfaced with effective problem formulation tools. Modern object-oriented modeling languages like Modelica and Optimica allow for rapid creation of complex dynamic optimization problems and lessen the burden of model development, optimization problem formulation, and solver interfacing. In this paper we make use of the Modelica-based open source software JModelica.org, [1], to transform high-level descriptions of dynamic optimization problems into algebraic nonlinear programming problems through a direct collocation approach. Applying a nonlinear interior-point method to solve this problem, the dominant computational expense is the solution of the KKT system solved at each iteration to produce the step in the primal and dual variables. The block-banded structure is decomposed by forming a Schur-complement with respect to the state continuity equations. The computational expense varies with the number of state variables and the number of processors used in the decomposition, as seen the parallel scaling results. As expected, the approach is most favorable for problems with fewer state variables than algebraic variables.

[1]Corresponding Author: carl.laird@tamu.edu. The authors gratefully acknowledge partial financial support from the National Science Foundation (CAREER Grant CBET# 0955205)

[2]The author gratefully acknowledges financial support from the Swedish Science Foundation through the grant *Lund Center for Control of Complex Engineering Systems (LCCC).*

2. Model Transcription

We consider dynamic optimization problems based on differential algebraic equation (DAE) models on the form

$$\min_u \int_{t_0}^{t_f} L(x,y,u)\,dt \tag{1}$$

subject to

$$F(\dot{x},x,y,u) = 0,\ x(t_0) = x_0 \tag{2}$$

where $\dot{x} \in R^{n_x}$ are the state derivatives, $x \in R^{n_x}$ are the states, $y \in R^{n_y}$ are the algebraic variables and $u \in R^{n_u}$ are the control inputs. It is assumed that the DAE is of index 1.

The optimization problem is discretized using a simultaneous collocation method based on finite elements, with Radau collocation points. See, e.g., [2] for a recent monograph. Lagrange polynomials are used to approximate the state, algebraic and control input profiles. Using this strategy, the discretized optimal control problem can be written on the form

$$\min_z f(z) \tag{3a}$$

$$\text{s.t.}\quad c(z) = 0 \tag{3b}$$

where $z^T = [z_1^T, \ldots, z_{n_e}^T]$ are the discrietized state, algebraic and input variables, and

$$c(z) = \begin{bmatrix} \overline{G}z_1 \\ R(z_1) \\ \underline{G}z_1 + \overline{G}z_2 \\ \vdots \\ \underline{G}z_{n_e-1} + \overline{G}z_{n_e} \\ R(z_{n_e}) \end{bmatrix} \qquad \begin{array}{l} \overline{G} = \begin{bmatrix} I & 0 & \ldots & 0 \end{bmatrix} \\ \underline{G} = \begin{bmatrix} 0 & \ldots & 0 & -I & 0 & 0 \end{bmatrix} \end{array} \tag{4}$$

Here, $\underline{G}z_{i-1} + \overline{G}z_i = 0$ are the coupling constraints linking individual finite elements in time and n_e is the number of finite element. $R(z_i)$ are the DAE residual equations and collocation equations associated with each finite element i. It is important to note that only the state variables are temporally coupled between elements (not the algebraic variables). Therefore, the dimension of these constraints (number of rows in $\underline{G}$ and $\overline{G}$) is dependent on the number of state variables only. It is this property that will be exploited to decompose the problem and develop an efficient parallel solution approach.

3. Parallel Solution of the Dynamic Optimization Problem

Solution of this large-scale nonlinear programming problem is possible with a number of potential algorithms. The dominant cost of an SQP-based or Interior-Point algorithm is the solution of the linear KKT system at each iteration to find the full step in the primal and dual variables. The structure of the objective and constraints in the optimal control problem induces a block structure within the linear KKT system. This linear system can be decomposed by selecting break-points in time between elements and performing a Schur-complement decomposition with respect to the coupling constraints.

For the interior-point algorithm IPOPT [3], the linear KKT system (also called the augmented system) solved at each iteration of the optimization algorithm can be written in the following block-bordered structure. For simplicity of notation, the structure is

written with a break-point at every finite element.

$$\begin{bmatrix} K_1 & & & & A_1^T \\ & K_2 & & & A_2^T \\ & & \ddots & & \vdots \\ & & & K_{n_e} & A_{n_e}^T \\ A_1 & A_2 & \dots & A_{n_e} & -\delta_c I \end{bmatrix} \begin{bmatrix} \Delta v_1 \\ \Delta v_2 \\ \vdots \\ \Delta v_{n_e} \\ \Delta v_s \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_{n_e} \\ r_s \end{bmatrix} \tag{5}$$

where

$$K_1 = \begin{bmatrix} H_1+\delta_H I & \overline{G}^T & \nabla_{z_1} R(z_1) \\ \overline{G} & & \\ \nabla_{z_1} R(z_1)^T & & \end{bmatrix}, K_k = \begin{bmatrix} H_i+\delta_H I & \nabla_{z_i} R(z_i) \\ \nabla_{z_i} R(z_i)^T & \end{bmatrix}, i = 2, ..., n_e, \tag{6}$$

Here, H_i is the Hessian of the Lagrangian for z_i, and δ_H, δ_c may be zero or positive depending on the need of the algorithm to handle non-convexity and/or singlularity in the Jacobian. The Δv_i vectors include the primal and dual variables for element i, and Δv_s is the dual variables for the coupling constraints. In this permutation, the coupling constraints (i.e. the Jacobian matrices $\underline{G}$ and $\overline{G}$), contained in the matrices A_i, and their corresponding dual variables have been permuted to the borders of the KKT system. The step in these dual variables can be decoupled from the remaining variables by eliminating the A_i matrices, resulting in the following Schur-complement decomposition,

$$\left[-\delta_c I - \sum_i A_i K_i^{-1} A_i^T \right] \Delta v_s = r_s - \sum_i A_i K_i^{-1} r_i. \tag{7}$$

This decomposition allows solution of the KKT system using the following algorithm.

Algorithm: Schur-Complement Solve of KKT System

1: for each i in $1, ..., n_e$
  1.1: factor K_i (using MA27 from Harwell Subroutine Library)
2: let $S = [-\delta_c I]$
3: let $r_{sc} = r_s$
4: for each i in $1, ..., n_e$
  4.1: for each column j in A_i^T
    4.1.1: solve the system $K_i q_i^{<j>} = [A_i^T]^{<j>}$
    4.1.2: let $S^{<j>} = S^{<j>} + A_i q_i^{<j>}$
  4.2: solve the system $K_i p_i = r_i$
  4.3: let $r_{sc} = r_{sc} - A_i p_i$
5: solve $S\Delta v_s = r_{sc}$ for Δv_s
6: for each i in $1, ..., n_e$
  6.1: solve $K_i \Delta v_i = r_i - A_i^T \Delta v_s$ for Δv_i

There are several levels of parallelism that can be exploited in this algorithm. If there is one processor available for each element, then Steps 1, 4, and 6 can all be parallelized. Furthermore, if more processors are available, individual column backsolves in Step 4.1 can be parallelized. Also, only a small number of columns in the matrices A_i contain non-zeros, a property that is exploited in Step 4.1.

The basic algorithm outlined above is described with one block for each individual finite element. However, the actual implementation is able to decompose the problem with multiple finite elements per block. For example, a problem with 128 finite elements can be separated into two blocks of 64 finite elements each, 4 blocks of 32 finite elements each, etc. At a minimum, there should be one processor available for each block.

The computational time of this algorithm is dominated by either Step 4 (forming the Schur-complement), or Step 5 (solving the Schur-complement). The cost of forming the Schur-complement scales linearly with the number of required backsolves (but is easily parallelized), whereas the cost of solving the Schur-complement using a typical dense linear solver is cubic in the size of the Schur-complement.

In previous work we have shown excellent parallel scalability using this strategy for problems with complicating variables[4, 5] where the size of the Schur-complement is determined by the number of coupling variables only. In the approach described here, the size of the Schur-complement increases with the number of states and the number of processors used. As the size of the Schur-complement grows, the increased cost of solving the large Schur-complement system (Step 5) will erode parallel speedup.

4. Performance Results

The performance of this parallel decomposition approach is a function of the number of state variables (i.e. the dimension of the coupling constraints), and the number of processors used in the decomposition. As we increase the number of processors, we have the potential for greater parallelization, however, the size of the Schur-complement (and hence the cost of Step 5) also increases. Therefore, we test the parallel speedup of this approach on a straightforward scalable problem where we can easily vary the number of state and algebraic variables, as well as the number of processors used in solution. The DAE models used in the benchmarks are composed of compartment models in series, where some compartments are approximated using a steady state assumption.

Table 1: Case Study Characteristics

Case Study	# State Vars.	# Algebraic Vars.
1	3	97
2	5	95
3	10	90
4	25	75

Four separate case studies are explored as we increase the number of processors as indicated in Table 1. Timing results represent ideal parallel timing as the problem was actually solved in serial and the time for Steps 1, 4, and 6 were divided by the number of blocks (available processors). The top plot in Figure 1 shows the idealized speedup using this approach as a function of the number of processors/blocks for each particular case study. Here, we see significant potential for speedup when the number of state variables is outnumbered by the number of algebraic variables. The bottom plot in Figure 1 shows the ratio of the time to solve the Schur-complement over the time to form the Schur-complement in parallel. The deterioration of the overall speedup corresponds to the point where the size of the Schur-complement increases such that the solution time is dominated by the time to solve the Schur-complement. All timing results were obtained on a 3.2 GHz Intel Xeon processor.

5. Conclusions and Future Work

This paper presents a decomposition approach that is applicable for parallel solution of the linear systems resulting from an interior-point solution of dynamic optimization prob-

lems formulated using the simultaneous approach. The dominant cost in this algorithm are Steps 4 (forming the Schur-complement) and 5 (solving the Schur-complement).

For problems with few states and many algebraics, this solution approach has the potential for significant speedup, however, as the size of the Schur-complement increases (more states or processors), parallel speedup is eroded. Nevertheless, this approach can be improved significantly. For blocks A_k of arbitrary structure, the Schur-complement may indeed be dense. However, for the dynamic optimization problem studied here, the structure of the A_k blocks is not arbitrary, and the resulting Schur-complement is both block structured and may be sparse. For the case studies using 32, 64, and 128 processors, the sparsity is below 10%, 5%, and 3% respectively. The use of an efficient sparse linear solver or even a parallel dense linear solver will dramatically decrease the time to solve the large Schur-complement and allow for significantly improved speedup.

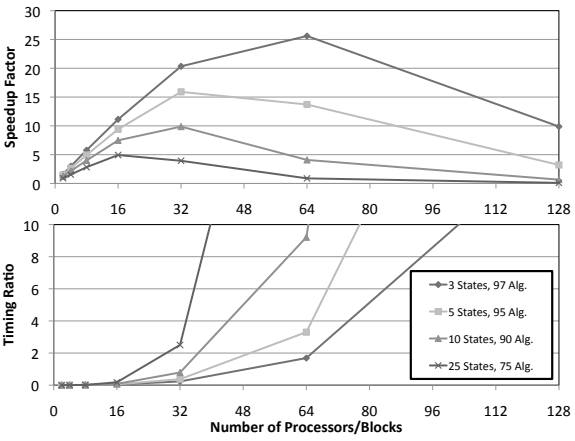

Figure 1: Case Study Timing Results: The top figure shows speedup results for different numbers of processors and state variables, while the bottom figure shows the ratio of the time to solve the Schur-complement over the time to form the Schur-complement.

References

[1] Johan Åkesson, Karl-Erik Årzén, Magnus Gäfvert, Tove Bergdahl, and Hubertus Tummescheit. Modeling and optimization with Optimica and JModelica.org—languages and tools for solving large-scale dynamic optimization problem. *Computers and Chemical Engineering*, 34(11):1737–1749, November 2010.

[2] Lorenz T. Biegler. *Nonlinear programming: concepts, algorithms, and applications to chemical processes*. SIAM, 2010.

[3] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–58, 2006.

[4] Zavala, V. M.; Laird, C.D.; Biegler L.T. Interior-point Decomposition Approaches for Parallel Solution of Large-scale Nonlinear Parameter Estimation Problems. Chemical Engineering and Science. 2008, 63, 4834-4845.

[5] Zhu, Y.; Legg, S.; and Laird, C. D. Optimal Design of Cryogenic Air Separation Columns under Uncertainty. Computers & Chemical Engineering, Volume 34, Issue 9, Selected papers from the 7th International Conference on the Foundations of Computer-Aided Process Design (FOCAPD, 2009, Breckenridge, Colorado, USA., 7 September 2010, Pages 1377-1384.