# Using Direct Collocation for Solving Bi-level Optimization Problems for Human Walking

**Vinh Q. Nguyen**[1], Frank C. Sup IV[1], Brian R. Umberger[2]

[1]Department of Mechanical & Industrial Engineering, University of Massachusetts, Amherst, MA, USA
[2]School of Kinesiology, University of Michigan, Ann Arbor, USA
Email: umberger@umich.edu

## Summary

Determining weights among terms in the cost function for walking was formulated as a bi-level optimization problem (BLO). The direct collocation (DC) method effectively solved the lower level problem, making it practical to solve the BLO using a nested evolutionary approach.

## Introduction

The problem of determining the cost function form for human walking may be cast as a BLO consisting of two coupled optimization problems. The lower level optimization is a standard optimal control of human walking [1] which solves for the gait solution, while the upper level solves for the cost function form [2] (e.g., weights among different candidate performance terms).

The BLO may be solved using a nested evolutionary approach. Since the nested evolutionary approach is computationally expensive, it is important that the lower level be solved in a reasonable time. DC is an efficient method [1, 3] that may be well-suited to solve the lower level. Therefore, we evaluated how DC may be used within the nested evolutionary approach, and further how parallelization may be used within the BLO and within the DC algorithm to reduce computation time.

## Methods

The problem of determining the weights among three performance criteria (muscle endurance, stability, and smoothness) in the walking cost function was formulated as a BLO. The BLO was solved with a nested evolutionary approach [2]. The upper level was solved with a genetic algorithm (GA), and the lower level was solved with the DC method using a 2-D, 11-DOF, 18-muscle OpenSim model. For computational efficiency, one step of walking was simulated on a 15-node grid using an Euler scheme. IPOPT solver was used to solve the lower level with the sparsity structure of the constraint Jacobian matrix provided [2].

### *A, Parallel computing within the GA*

GA is well-suited for parallel computing implementation for speeding up the simulation. Therefore, the GA was parallelized using the Matlab Parallel Computing toolbox on an Intel i9 3.5 GHz 10-core computer (ParGA).

### *B, Parallel computing within DC*

In DC, evaluating the dynamic equations at multiple nodes can be done simultaneously to reduce the computation time. We evaluated this approach by solving a lower level problem with parallel computing on the same computer (ParDC). To evaluate the effectiveness of these parallelization, we ran three configurations namely: parallelizing GA only (ParGA), parallelizing DC only (ParDC), and parallelizing both levels.

## Results and Discussion

Each lower level simulation, representing a full optimal control solution for walking, was efficiently solved with serial computing in about 12 minutes, making it practical to solve the overall BLO. The solution time was reduced with parallel computing. For ParGA, the run-time reduced almost linearly with the number of cores (Fig.1A). ParDC showed some improvements (Fig.1B), but not as much as in ParGA. A maximum improvement of 1.35 times speed-up was achieved with parallelizing on 6 cores.

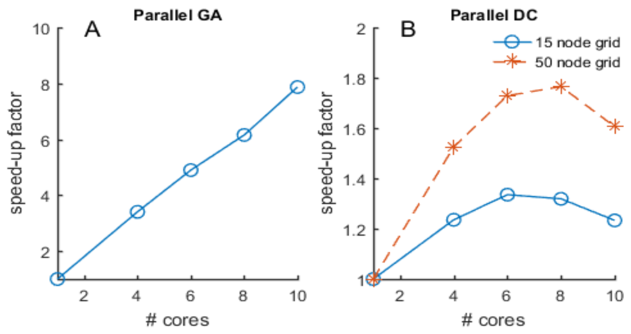

**Figure 1**: Speed-up by parallel computing with the GA (A) and DC (B)

Therefore, for the BLO, it is more beneficial to use ParGA than ParDC. Recently, we solved the BLO in 139 hr using 10 cores with the GA [2]. While ParDC was not as beneficial for the BLO, standard optimal control simulations may still benefit from parallel implementation. We tested performance of ParDC with a more typical grid density of 50 nodes, and found greater improvements compared to the coarse grid (15-nodes) used for the BLO (Fig.1B).

Parallel computing for both levels of the BLO should give further improvements. However, on the single multiple-core CPU, parallelizing both levels resulted in no improvement compared with ParGA only. Potentially, on a multiple CPU computer with each CPU having multiple cores, ParDC on each CPU and ParGA using multiple CPUs, may yield greater improvements in performance for solving the BLO.

## Conclusions

DC is effective for use within parallel GA for solving BLO problems. DC may also be sped-up by solving the dynamic equation constraints in parallel on multiple core CPUs.

## Acknowledgments

This work was supported in part by a grant from the National Science Foundation (IIS-1526986).

## References

[1] Ackermann & Bogert (2010), *J Biomech*, 43(6):1055–60
[2] Nguyen, et al. (2019). *TNRSE*, (in review)
[3] Lee & Umberger (2016) *PeerJ*, 4:e1638