Journal of Biomechanics 38 (2005) 1938–1942

JOURNAL OF BIOMECHANICS

www.elsevier.com/locate/jbiomech
www.JBiomech.com

Short communication

Simulated parallel annealing within a neighborhood for optimization of biomechanical systems

J.S. Higginson[a,b], R.R. Neptune[c], F.C. Anderson[b,*]

[a] VA Palo Alto Rehabilitation R&D Center, Palo Alto, CA 94304
[b] Department of Mechanical Engineering, Biomechanical Engineering Division, Stanford University, Clark Center, S-342 (MC 5450), 318 Campus Drive, Stanford, CA 94305 5450, USA
[c] Department of Mechanical Engineering, University of Texas, Austin, TX 78712, USA

Accepted 4 August 2004

Abstract

Optimization problems for biomechanical systems have become extremely complex. Simulated annealing (SA) algorithms have performed well in a variety of test problems and biomechanical applications; however, despite advances in computer speed, convergence to optimal solutions for systems of even moderate complexity has remained prohibitive. The objective of this study was to develop a portable parallel version of a SA algorithm for solving optimization problems in biomechanics. The algorithm for simulated parallel annealing within a neighborhood (SPAN) was designed to minimize interprocessor communication time and closely retain the heuristics of the serial SA algorithm. The computational speed of the SPAN algorithm scaled linearly with the number of processors on different computer platforms for a simple quadratic test problem and for a more complex forward dynamic simulation of human pedaling.

Keywords: Musculoskeletal modeling; Forward dynamic simulation; Optimization; Simulated annealing; Parallel computing

1. Introduction

The study of biological systems, ranging from reconstruction of genetic chromosomes (e.g., Bhandarkar et al., 2001) to simulation of human movement (e.g., Anderson and Pandy, 2001), has been advanced by the application of optimization techniques. Because optimization problems in biomechanics often have objective functions that are discontinuous, nonlinear, and possess many local optima, these problems can be extremely difficult to solve. Simulated annealing (SA), as well as other stochastic optimization approaches, has been shown to outperform gradient and simplex methods on such problems (van Soest and Casius, 2003; Neptune, 1999). Indeed, SA has been applied effectively in biomechanical studies of walking (Neptune et al., 2001), running (Neptune et al., 2000a; Wright et al., 1998), pedaling (Neptune et al., 2000b), reaching (Zhang et al., 1998), knee implant (Mahfouz et al., 2003) and medical image registration (Matsopoulos et al., 2003; Loeckx et al., 2003; Rueckert et al., 1997), kinematic analysis of joint implants (Sarojak et al., 1999), and balance recovery (Iqbal and Pai, 2000). Despite its robustness and recent advances in computer speeds, optimization with SA algorithms can be computationally prohibitive for many biomechanical problems of interest (Neptune, 1999).

In an effort to increase convergence speed, parallel SA optimization algorithms have been implemented with mixed results in various scientific fields (Chen et al., 1998). Although optimization performance can be greatly improved through parallelization of SA

*Corresponding author. Tel.: +1-650-736-0801; fax: +1-650-724-1922.
E-mail address: fca@stanford.edu (F.C. Anderson).

0021-9290/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jbiomech.2004.08.010

(Diekmann et al., 1992; Kliewer and Tschöke, 2000), there has not been a single application of this technique in biomechanics.

The objective of this study was to formulate a portable parallel version of the SA algorithm and introduce this algorithm to the biomechanics community. This algorithm, called *Simulated Parallel Annealing within a Neighborhood* (SPAN), can greatly improve convergence rates for difficult optimization problems, thereby advancing the state of biomechanical analyses. A simple parabolic test function and a complex pedaling simulation are used to demonstrate SPAN's scalability with increasing numbers of processors. Furthermore, SPAN results for the pedaling optimization are compared with those from a serial version of SA to demonstrate that parallelization does not significantly alter solutions found by the algorithm.

2. Methods

2.1. SPAN algorithm

Simulated annealing executes a search for the optimal solution by randomly altering the control variables within a neighborhood of the solution space and evaluating the performance criterion. Due to the stochastic heuristics of the algorithm, SA is able to overcome local optima and is relatively insensitive to the initial guess for the optimized parameters. Convergence is achieved when a sufficiently large number of performance evaluations have been made without any improvement to the optimum (for details, see Corana et al., 1987; Goffe et al., 1994; Ingber, 1993).

The SA algorithm consists of three nested loops (Fig. 1a): (1) an inner-most *control loop* that makes performance evaluations for each perturbation of the N control variables, (2) a *search-radius loop* that executes the *control loop* N_S times before altering the size of the search neighborhood, and (3) a *temperature-reduction loop* that executes the *search-radius loop* N_T times before reducing the temperature. Parallelizing the SA algorithm at the level of the *search-radius loop* offers a large speed-up (ideally by a factor of N_S) without compromising the heuristics of the algorithm (Fig. 1b).

The SPAN algorithm was implemented in FORTRAN on a shared-memory supercomputer (128 processor Silicon Graphics Origin 3800) at the Center for Biomedical Computation at Stanford University and tested for portability at the Texas Advanced Computing Center at The University of Texas at Austin (IBM Power4 System). Interprocessor communications in SPAN were accomplished using the Message Passing Interface (MPI), which makes the code portable across a variety of parallel computing environments (Gropp et al., 1996; Gropp and Lusk, 1996).

2.2. Simple parabola

The computational performance of SPAN on 1, 2, 4, 8, 16, and 32 processors was evaluated by finding the minimum of a 50-dimensional parabola:

$$J_1 = \sum_{i=1}^{50} x_i^2, \tag{1}$$

where each x_i was an independent control variable initially set to 5.0 (range: –10.0 to 10.0). The evaluation time for this function (Eq. (1)) was artificially increased by factors of 2 and 4, respectively, to compare performance for functions that require greater computing time. In addition, the cooling schedule was selected

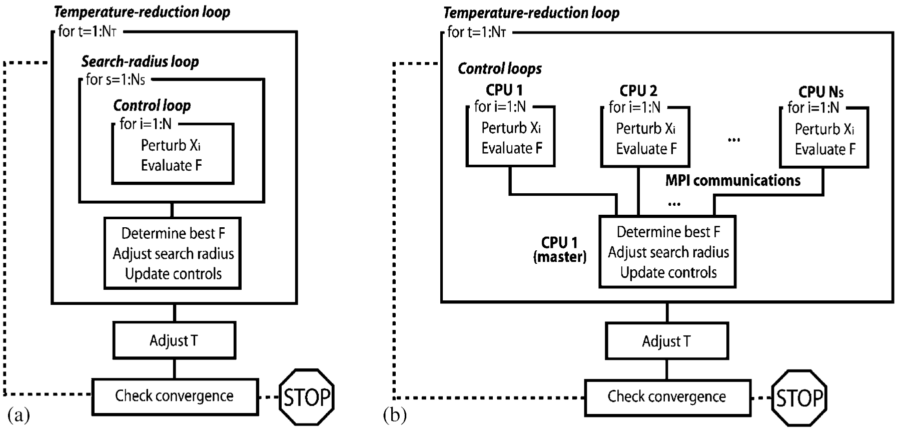

Fig. 1. Sequential and parallel SA schematics. (a) Total number of function evaluations before checking convergence is $N{*}N_S{*}N_T$. Time to convergence using sequential SA depends on time per function evaluation. (b) With parallel decomposition at the level of the *search-radius loop*, the number of processors that can be effectively recruited is N_S. Each processor (CPU) receives input data and perturbs the control parameters by a random fraction of the search radius. After $N{*}N_S$ total function evaluations, each processor communicates its local optimal function value to CPU1 (master), which determines the best solution so far. All processors are updated with the current best function value, the corresponding controls, and a new search radius.

to allow recruitment of up to 32 processors (i.e., $r_T = 0.75$, $N_T = 100$, $N_S = 32$).

2.3. Pedaling simulation

The second evaluation consisted of optimizing the muscle excitation patterns in a simulation of pedaling to reproduce experimentally collected pedaling data. A previously described musculoskeletal model of pedaling (Neptune and Hull, 1998) was driven by nine muscle groups governed by Hill-type muscle properties (Fig. 2). The excitation pattern of each muscle group was defined by an excitation onset, duration, and magnitude (i.e., 9 muscles x 3 excitation variables = 27 controls). The performance criterion was formulated to minimize the sum of the squared residuals between the simulated and experimentally collected pedal forces and joint moments and powers from 10 subjects pedaling at 60 rpm and 250 W as:

$$J_2 = \sum_{j=1}^{m}\sum_{i=1}^{n}\frac{(Y_{ij} - \hat{Y}_{ij})}{\mathrm{SD}_j^2}, \tag{2}$$

where Y_{ij} is the measurement of variable j at time step i, $\hat{Y}_{ij}$ the simulation data corresponding to Y_{ij} and SD_j^2 the average inter-subject variability of variable j.

An aggressive cooling schedule (i.e., $r_T = 0.75$, $N_T = 5$, $N_S = 20$) was applied to speed-up the convergence on 1, 2, 5, 10 and 20 processors (Neptune and Hull, 1998). In both test functions, all other parameters associated with the SA algorithm were set to the default values suggested by Corana et al. (1987).

3. Results

Minimization of the 50-dimensional parabola (Eq. (1)) produced the same solution on 1, 2, 4, 8, 16 and 32 processors. Each function evaluation took less than 1 ms. Communication time was directly proportional to the number of processors used, with the time per communication event ranging between 0.1 ms for two processors and 1 ms for 32 processors. Computational speed-up scaled nearly linearly with the number of processors (Fig. 3), and results were similar on both the SGI and IBM computer platforms.

Similar scaling results were obtained for the pedaling optimization. Communication time was negligible compared to the average function evaluation time of 39 s. Low communication time relative to function evaluation time yielded a nearly perfect linear performance increase with the number of processors used (Fig. 3). SPAN identified the near-optimal muscle excitation patterns that reproduced the experimental pedal forces and pedaling kinematics (Fig. 4) with an error comparable to the solution found using a serial SA algorithm (Neptune and Hull, 1998).

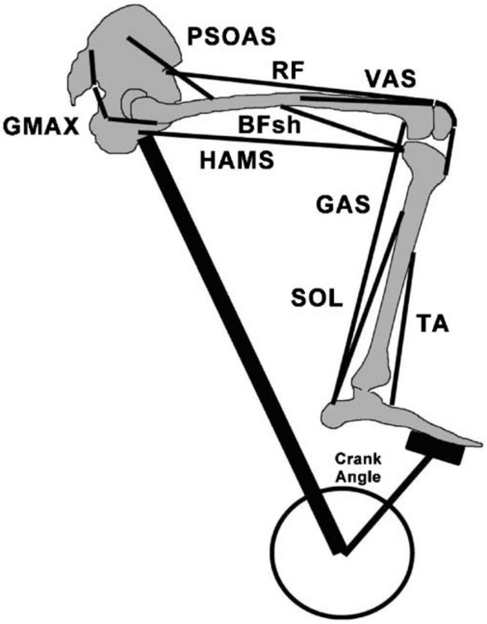

Fig. 2. Musculoskeletal model of pedaling. Twenty-seven control parameters specify muscle excitation patterns defined by onset, offset, and magnitude for nine muscle groups shown. Symmetry was assumed between the left and right legs.

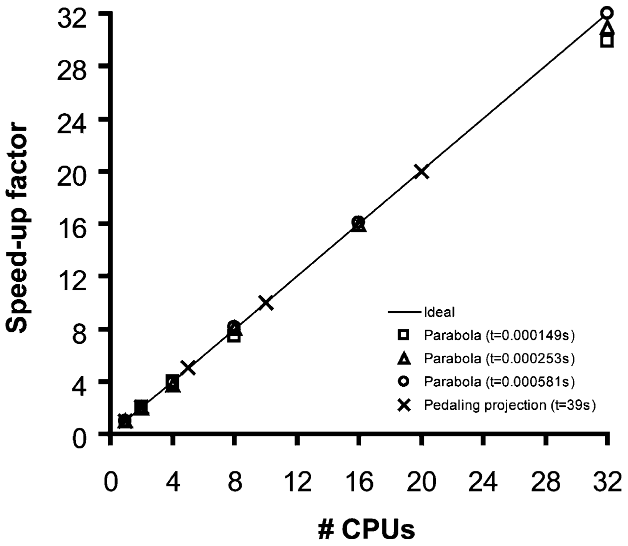

Fig. 3. Speed-up factor (or ratio of time to convergence for 1 processor compared to n processors) for simple and complex functions. Evaluation times for parabolic and pedaling functions are shown in the legend. Even for small function evaluation times, performance scaling was nearly ideal. Only for evaluation times less than 0.5 ms on 32 processors was there any significant decrement in performance below perfect performance scaling. Results for pedaling are based on projections of 16,000 function evaluations performed before convergence.

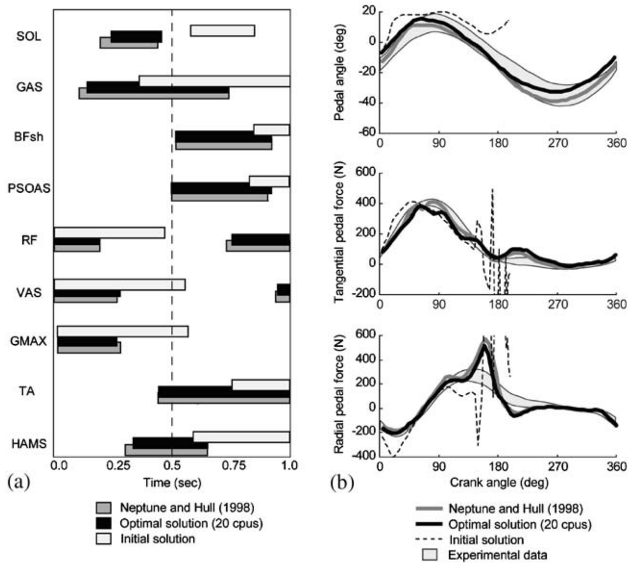

Fig. 4. Optimal solution using SPAN for pedaling simulation compared to experimental data and the solution in Neptune and Hull (1998). (a) Convergence from initial muscle excitation patterns (white bars) to current optimal results (black bars) demonstrates similarity with Neptune and Hull (1998) solution (gray bars). The initial solution was derived from an inverted cost function that maximized the tracking error, thus producing a poor simulation. (b) Initial solution (dashed line) did not complete a full revolution. Optimal pedaling kinematics and kinetics with SPAN (black line) were similar to experimental data (shaded gray area) and those reported with a serial SA algorithm used by Neptune and Hull (1998) (gray line).

4. Discussion

Our objective was to develop a portable parallel simulated annealing algorithm that offers sufficient performance benefits to render previously intractable problems in biomechanics tractable. The SPAN algorithm retains the essential characteristics of serial simulated annealing (Goffe et al., 1994) and its computational performance scales nearly ideally with the number of processors recruited. This performance scaling was obtained for a simple 50-dimensional parabola that required very little evaluation time (i.e., less than 1 ms) and for a more complex optimal tracking problem that is representative of the kinds of optimization problems encountered in biomechanics. Thus, for optimization problems of practically any scale, SPAN is capable of reducing time to convergence by about a factor of N_S. The larger the optimization problem, the more confident one can be that SPAN will achieve a speed-up factor of N_S. In addition, because interprocessor communications are carried out using the MPI standard (Gropp et al., 1996; Gropp and Lusk, 1996), the SPAN algorithm is portable across a large number of computer systems, including workstation clusters.

We designed the SPAN algorithm to minimize the overhead associated with interprocessor communications, which may vary considerably across computing environments. Our parallelization strategy requires an interprocessor communication event only once every $N{*}N_S$ function evaluations, which means that near linear scaling can be achieved even for objective functions that require very little evaluation time (e.g., Eq. (1)). The low communication overhead makes SPAN well suited to run on workstation clusters, which typically have longer communication latencies than shared-memory supercomputers. As long as the time for N function evaluations is greater than the time required for an interprocessor communication event, the SPAN algorithm will result in improved performance.

While the suitability of SPAN for solving truly large-scale problems (i.e., problems with hundreds or thousands of control variables) has yet to be demonstrated, it is clear that SPAN can offer substantial reductions in convergence time. For $N_S = 20$ (the recommended value for the serial algorithm), an optimization problem that would take a year of computer time with a serial simulated annealing algorithm (e.g., simulation of walking in three dimensions) would require only about 2.5 weeks with SPAN. Given processor availability, N_S can be increased with little or no detriment and may improve the convergence characteristics of SPAN, making the algorithm a possible alternative to recent parallel genetic algorithms (e.g., van Soest and Casius, 2003) or particle swarm techniques (e.g., Zhou et al., 2003). Future work will be directed at comparing the effectiveness of these algorithms in solving large-scale problems in biomechanics. The source code for SPAN is freely available at www.isbweb.org.

Acknowledgements

We would like to acknowledge the computational resources of the Center for Biomedical Computation at Stanford and the Texas Advanced Computing Center at The University of Texas at Austin. This research was supported by the US Department of Veterans Affairs Rehabilitation Research and Development Service, NIH GM 63495, and NIH R01-HD38962 to Scott Delp.

References

Anderson, F.C., Pandy, M.G., 2001. Dynamic optimization of human walking. Journal of Biomechanical Engineering 123, 381–390.

Bhandarkar, S.M., Machaka, S.A., Shete, S.S., Kota, R.N., 2001. Parallel computation of a maximum-likelihood estimator of a physical map. Genetics 157, 1021–1043.

Chen, H., Flann, N.S., Watson, D.W., 1998. Parallel genetic simulated annealing: a massively parallel SIMD algorithm. IEEE Transactions on Parallel Distributed Systems 9, 126–136.

Corana, A., Marchesi, M., Martini, C., Ridella, S., 1987. Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. ACM Transactions on Mathematical Software 13, 262–280.

Diekmann, R., Luling, R., Simon, J., 1992. A general purpose distributed implementation of simulated annealing. Fourth IEEE Symposium on Parallel and Distributed Processing, pp. 94–101.

Goffe, W.L., Ferrier, G.D., Rogers, J., 1994. Global optimization of statistical functions with simulated annealing. Journal of Econometrics 60, 65–100.

Gropp, W., Lusk, E., 1996. User's Guide for MPICH, A Portable Implementation of MPI: Argonne National Laboratory.

Gropp, W., Lusk, E., Doss, N., Skjellum, A., 1996. A high-performance, portable implementation of the MPI message passing interface standard. Parallel Computing 22, 789–828.

Ingber, L., 1993. Simulated annealing: practice versus theory. Mathematical and Computer Modelling 18, 29–57.

Iqbal, K., Pai, Y., 2000. Predicted region of stability for balance recovery: motion at the knee joint can improve termination of forward movement. Journal of Biomechanics 33 (12), 1619–1627.

Kliewer, G., Tschöke, S., 2000. A general parallel simulated annealing library and its application in airline industry. In: Proceedings of the 14th International Parallel and Distributed Processing Symposium, Cancun, Mexico, pp. 55–62.

Loeckx, D., Maes, F., Vandermeulen, D., Suetens, P., 2003. Temporal subtraction of thorax CR images using a statistical deformation model. IEEE Transactions on Medical Imaging 22 (11), 1490–1504.

Mahfouz, M.R., Hoff, W.A., Komistek, R.D., Dennis, D.A., 2003. A robust method for registration of three-dimensional knee implant models to two-dimensional fluoroscopy images. IEEE Transactions on Medical Imaging 22 (12), 1561–1574.

Matsopoulos, G.K., Delibasis, K.K., Mouravliansky, N.A., Asvestas, P.A., Nikita, K.S., Kouloulias, V.E., Uzunoglu, N.K., 2003. CT-MRI automatic surface-based registration schemes combining global and local optimization techniques. Technology and Health Care 11 (4), 219–232.

Neptune, R.R., 1999. Optimization algorithm performance in determining optimal controls in human movement analyses. Journal of Biomechanical Engineering 121, 249–252.

Neptune, R.R., Hull, M.L., 1998. Evaluation of performance criteria for simulation of submaximal steady-state cycling using a forward dynamic model. Journal of Biomechanical Engineering 120, 334–341.

Neptune, R., Wright, I., van den Bogert, A., 2000a. The influence of orthotic devices and vastus medialis strength and timing on patellofemoral loads during running. Clinical Biomechanics 15, 611–618.

Neptune, R.R., Kautz, S.A., Zajac, F.E., 2000b. Muscle contributions to specific biomechanical functions do not change in forward versus backward pedaling. Journal of Biomechanics 33, 155–164.

Neptune, R.R., Kautz, S.A., Zajac, F.E., 2001. Contributions of the individual ankle plantar flexors to support, forward progression and swing initiation during walking. Journal of Biomechanics 34, 1387–1398.

Rueckert, D., Burger, P., Forbat, S.M., Mohiaddin, R.D., Yang, G.Z., 1997. Automatic tracking of the aorta in cardiovascular MR images using deformable models. IEEE Transactions on Medical Imaging 16 (5), 581–590.

Sarojak, M., Hoff, W., Komistek, R., Dennis, D., 1999. An interactive system for kinematic analysis of artificial joint implants. Biomedical Science Instruments 35, 9–14.

van Soest, A.J., Casius, L.J.R.R., 2003. The merits of a parallel genetic algorithm in solving hard optimization problems. Journal of Biomechanical Engineering 125, 141–146.

Wright, I.C., Neptune, R.R., van den Bogert, A.J., Nigg, B.M., 1998. Passive regulation of impact forces in heel-toe running. Clinical Biomechanics 13, 521–531.

Zhang, X., Kuo, A., Chaffin, D., 1998. Optimization-based differential kinematic modeling exhibits a velocity-control strategy for dynamic posture determination in seated reaching movements. Journal of Biomechanics 31, 1035–1042.

Zhou, Y., Zeng, G., Yu, F., 2003. Particle swarm optimization-based approach for optical finite impulse response filter design. Applied Optics 42 (8), 1503–1507.