Parallel Nested-Layer Particle Swarm Optimization for bifurcation parameter detection

Tomo Hasegawa
Department of Electric and Electronic Engineering, Tokyo University of Technology
Hachioji Tokyo Japan
0000-0003-1242-8058

Haruna Matsushita
Department of Electronics and Information Engineering, Kagawa University
Takamatsu Kagawa Japan
0000-0002-7850-5119

Takuji Kousaka
Department of Electrical and Electronic Engineering, Chukyo University
Nagoya Aichi Japan
0000-0002-6368-4089

Hiroaki Kurokawa
Department of Electric and Electronic Engineering, Tokyo University of Technology
Hachioji Tokyo Japan
0000-0003-2340-1424

***Abstract*—Nested-Layer Particle Swarm Optimization (NLPSO) has been proposed to detect bifurcation parameters in nonlinear dynamical systems. Bifurcation parameter detection by NLPSO is a simple method because it does not require precise initialization and does not require differential information from the dynamical system. NLPSO's disadvantage is its high computational complexity. We have proposed parallel NLPSO, which solves this problem using parallel computation and is implemented as a software library. The parallelization is based on optimizing the NLPSO algorithm for execution on multi-core CPUs. In this paper, we show the efficiency of Parallel NLPSO for computational time.**

***Index Terms*—bifurcation parameters, parallel processing, Particle swarm optimization,**

I. Introduction

Many of the phenomena in the world can be described by dynamical systems. Bifurcation analysis is essential in analyzing various phenomena that dynamical systems can describe. The derivation of bifurcation points is formulated as an optimization problem, and methods based on the Newton-Raphson method have been used in conventional studies [1]–[3]. In contrast, several methods based on Particle Swarm Optimization (PSO) [4] have been proposed [5]–[7]. PSO is a versatile optimization method that can be easily applied to various problems due to the simplicity of the algorithm processes, and it does not require gradient information for the objective function. NestedLayer Particle Swarm Optimization (NLPSO) [8]–[11] is a bifurcation parameter search method using PSO. The NLPSO algorithm [8]–[11] is a nested combination of two PSOs, one to derive the periodic points for the bifurcation parameter derivation process and the other to derive the bifurcation parameters corresponding to the periodic points. Bifurcation parameter search with NLPSO has the advantage that precise initial value setting is not required.

On the other hand, high computational complexity is a well-known problem of the NLPSO. To reduce the computation time, which has been a problem of NLPSO, we have proposed the parallel NLPSO and developed a software library that works by inputting only the equations describing the system, the number of cycles in which the bifurcation occurs, and the search range [12]. The software can search bifurcation parameters for period-doubling bifurcation and saddle-node bifurcation in discrete-time and continuous-time dynamical systems. This paper presents an overview of parallel NLPSO and its effectiveness.

II. Nested-Layer PSO

A dynamical system can be classified into two types: discrete-time dynamical systems, which define time changes discretely, and continuous-time dynamical systems, which define time changes continuously. These are respectively described by the following difference equation and differential equation:

$$x(k+1) = f(x(k), \lambda) \tag{1}$$

$$\frac{\mathrm{dx(t)}}{\mathrm{dt}} = g(t, x(t), \lambda) \tag{2}$$

Here, λ is a parameter in the dynamical system.

In a discrete dynamical system, if the n-fold composition of an arbitrary map f is denoted as f^n, then a point x_p that satisfies the following equation with a parameter λ is called an n-periodic point:

$$f^n(x_p, \lambda) = x_p \tag{3}$$

Suppose $Df^n(x_p, \lambda)$ is the Jacobian matrix of f^n at x_p, it gives the characteristic equation as follows:

$$\det(Df^n(x_p, \lambda) - \mu I_N) = 0 \tag{4}$$

Here, when the characteristic multiplier $\mu = -1$, a period-doubling bifurcation occurs, and when $\mu = 1$, a saddle-node bifurcation or pitchfork bifurcation occurs. Therefore, the search for bifurcation parameters can be formulated as a mathematical optimization problem, where the goal is to find

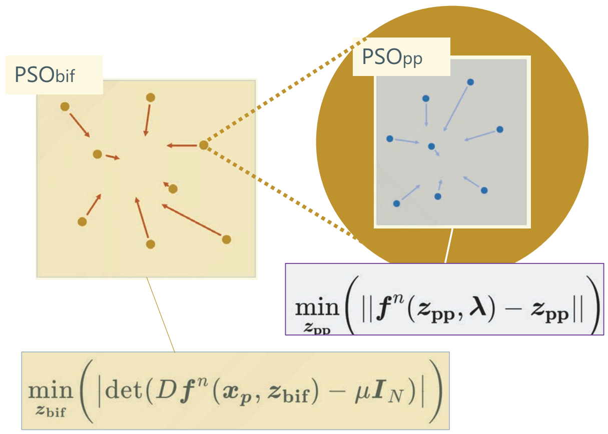


Fig. 1. Schematic of NLPSO: Each particle of $\mathrm{PSO}_{\mathrm{bif}}$ calls $\mathrm{PSO}_{\mathrm{pp}}$ in each calculation to update its position and velocity.

the periodic point x_p and parameter λ that satisfy Equations 3 and 4 under the conditions $\mu = -1$ or $\mu = 1$.

To solve this problem, two objective functions, $F_{\mathrm{bif}}(z_{\mathrm{bif}})$ and $F_{\mathrm{pp}}(z_{\mathrm{pp}})$, are defined with z_{bif} and z_{pp} as the decision variables:

$$F_{\mathrm{bif}}(x_p, z_{\mathrm{bif}}) = \tag{5}$$
$$\begin{cases} |\det(Df^n(x_p, z_{\mathrm{bif}}) - \mu I_N)| & \text{if } F_{\mathrm{pp}}(x_p, z_{\mathrm{bif}}) < C_{\mathrm{pp}}, \\ \infty & \text{otherwise.} \end{cases}$$

$$F_{\mathrm{pp}}(z_{\mathrm{pp}}, \lambda) = \|f^n(z_{\mathrm{pp}}, \lambda) - z_{\mathrm{pp}}\| \tag{6}$$

Here, z_{bif} corresponds to the parameters of the dynamical system, i.e., λ, and z_{pp} corresponds to the n-periodic point.

NLPSO solves this optimization problem by executing two PSOs in a nested configuration. The schematic is shown in Fig. 1. The PSO that minimizes each objective function is referred to as $\mathrm{PSO}_{\mathrm{bif}}$ and $\mathrm{PSO}_{\mathrm{pp}}$, respectively. In NLPSO, each time a particle of $\mathrm{PSO}_{\mathrm{bif}}$ is updated, $\mathrm{PSO}_{\mathrm{pp}}$ is called to proceed with the computation. It should be noted that PSO is an algorithm in which multiple particles explore the solution space. For continuous dynamical systems, bifurcation parameters can be searched by applying a similar method used in discrete dynamical systems to the Poincaré map. For further details, please refer to [8]–[11], [13].

III. Parallel NLPSO

As discussed in the previous section, NLPSO is an algorithm that requires a large amount of computation due to its nested structure. To address this problem, we apply parallel computing to NLPSO. While GPU-based methods are well-known, managing GPU-equipped systems can be complex for non-expert users. Therefore, we focus on parallelization that can be executed on standard PCs with up to 16-core shared-memory multicore CPUs.

In shared-memory multicore CPU parallelization, the overhead of fork-join operations often reduces performance improvements. Here, fork-join refers to the procedure of distributing tasks across threads and then merging the results. Given that we are not targeting many cores, i.e., we assume to use up to 16 cores at most, we parallelize by assigning the computation of each particle in $\mathrm{PSO}_{\mathrm{bif}}$ to a separate thread. This means parallelization is limited to the number of particles in $\mathrm{PSO}_{\mathrm{bif}}$.

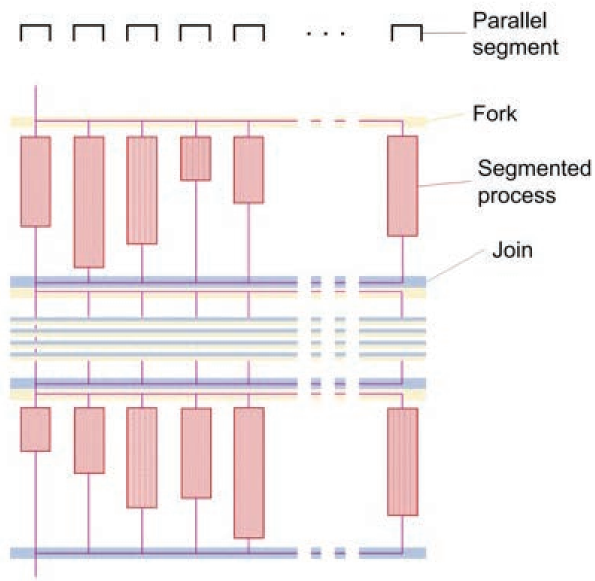


Fig. 2. Concept of naive Parallel NLPSO. Synchronous updating by each particle is required to implement the PSO algorithm strictly.

Generally, the PSO requires synchronous updating by each particle because each particle updates its position and velocity and then updates the global best solution for the entire particle swarm. If the PSO algorithm is implemented strictly, this would require thread synchronization, as shown in Fig. 2, leading to frequent fork-join overhead, thereby reducing the effectiveness of parallelization. Additionally, unnecessary waiting time for synchronization among threads further diminishes the parallelization effect.

To solve these issues, we proposed an asynchronous PSO update method, as shown in Fig. 3. Although each particle may not always refer to the most recent best solution, our numerical experimental validation has demonstrated that this relaxation of conditions has little negative effect on the solution search.

IV. Examples

We implemented an asynchronous update parallel NLPSO using multi-threading with OpenMP. Also, the differentiation required to obtain the Jacobian matrix was automated using numerical differentiation. We performed bifurcation parameter searches on several discrete-time nonlinear dynamical systems. The results for computation times are shown in Table I. Note that NNS [17] represents a higher-dimensional system with increased computational complexity. The CPUs used were Intel's 12900K (16 cores), 10900K (10 cores), and 1065G7 (4 cores). While the effect of parallelization varies with the number of cores, computation times were reduced in all cases. The number of particles for $\mathrm{PSO}_{\mathrm{bif}}$ was set to 30. Although the

TABLE I
COMPARISON OF COMPUTATIONAL TIMES BETWEEN NLPSO AND PARALLEL NLPSO ON THE DISCRETE-TIME DYNAMICAL SYSTEM. PARALLEL NLPSO IS EXECUTED ON DIFFERENT CPUS

		Serial NLPSO	Parallel NLPSO		
Target		**time [s] (12900K)**	**time [s] (12900K)**	**time [s] (10900K)**	**time [s] (106567)**
Hénon map	**PD 5**	3.137	0.251	0.449	0.727
	PD 6	9.743	0.499	0.890	2.257
	SN 5	[illegible]	[illegible]	[illegible]	[illegible]
	SN 6	[illegible]	[illegible]	[illegible]	[illegible]
NNS	**SN 64**	[illegible]	[illegible]	[illegible]	[illegible]
	SN 65	[illegible]	[illegible]	[illegible]	[illegible]

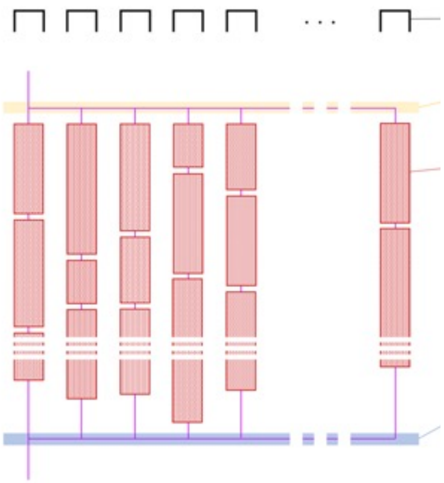
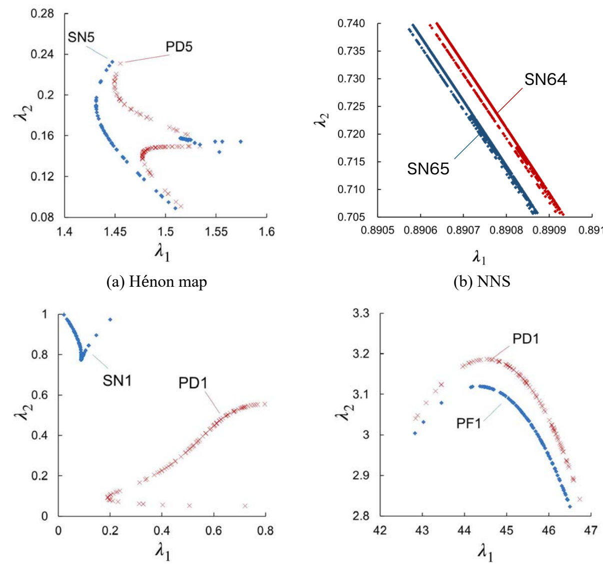


Fig. 3. Concept of proposed Parallel NLPSO. Asyn[illegible] applied.



(a) Hénon map (b) NNS

(c) non-autonomous (d) Chen's equation

Fig. 4. Results of the bifurcation parameter detection using our parallelized algorithm.

number of CPU cores does not always match the number of threads, the appropriate bifurcation parameters were obtained.

TABLE II
COMPARISON OF COMPUTATIONAL TIMES BETWEEN NLPSO AND PNLPSO ON THE CONTINUOUS-TIME DYNAMICAL SYSTEMS.THE PROGRAM IS EXECUTED BY INTEL 12900K.

		Serial NLPSO	Parallel NLPSO
Target		**time [s]**	**time [s]**
(non-autonomous) [15]	**PD 1**	5025.23	1006.667
	SN 1	48915.951	7153.806
Chen's equation [16]	**PD 1**	78544.968	14913.277
	PF 1	184663.846	41495.298

Additionally, we developed a software library that simplifies the use of the proposed PNLPSO. The library was implemented using RUST. We executed bifurcation parameter searches using this software library on several continuous-time nonlinear dynamical systems. The results for computation times are shown in Table II, where only the results using Intel's 12900K (16 cores) are presented. These results show the efficiency of parallelization. As a result of the bifurcation parameter detection using our parallelized algorithm, we obtained a set of bifurcation parameters, as shown in Fig. 4.

V. CONCLUSION

In this paper, we describe an overview of parallel NLPSO and its implementation and demonstrate the improved computation speed when applied to several dynamical systems. We developed highly versatile software that operates on a general-purpose computer and performs bifurcation parameter searches using only the system definition, period number, and search range. This enables bifurcation parameter searches to be executed without special knowledge of bifurcation analysis or parallel computing.

As mentioned in Section III, utilizing GPUs is an effective means of achieving parallel computing. In our previous studies, we have parallelized the NLPSO algorithm using GPU [18]. In GPU-based parallelization, all particles of PSO_{pp} are computed in parallel simultaneously, and the positions and velocities of both PSO_{pp} and PSO_{bif} particles are updated using these results. At this point, the particles of PSO_{pp} and PSO_{bif} are updated in strict synchronization. Although there are possibilities for improvement in the algorithm when considering the efficiency of GPU resource utilization and parallelism, applying this approach to several nonlinear dynamical systems has demonstrated a successful acceleration effect. Here, the more complex the system being analyzed, the greater the acceleration effect achieved through parallelization.

In contrast, the shared-memory multicore CPU implementation of PNLPSO presented in this paper has shown relatively higher acceleration effects for simpler systems with shorter computation times, unlike the case with GPU-based parallelization. Thus, it is suggested that parallelization using GPUs and parallelization using multicore CPUs show different characteristics due to the differences in their respective algorithms. Providing a precise explanation of these characteristic differences remains a challenge for future work.

Furthermore, while a direct comparison is difficult due to differences in the required computing systems and compilers, in general, GPU-based parallelization tends to perform faster. However, as demonstrated in this paper, the acceleration achieved with PNLPSO using multicore CPUs is sufficiently effective. Considering that, as discussed in Section III, this method does not require relatively expensive devices such as GPUs or complicated system management, it proves to be a highly practical approach.

REFERENCES

[1] H. Kawakami, "Bifurcation of periodic responses in forces dynamic nonlinear circuits: Computation of bifurcation values of the system parameters," IEEE Transactions on Circuits and Systems, vol. 31, no. 3, pp.248–260, March 1984.

[2] T. Ueta, M. Tsueike, H. Kawakami, T. Yoshinaga, and Y. Katsura, "A computation of bifurcation parameter values for limit cycles," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E80-A, no. 9, pp. 1725–1728, September 1997.

[3] K. Tsumoto, T. Ueta, T. Yoshinaga, and H. Kawakami, "Bifurcation analyses of nonlinear dynamical systems: From theory to numerical computations," NOLTA, IEICE, vol. 3, no. 4, pp. 458–476, October 2012.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, pp. 1942–1948, vol.4, 1995.

[5] C. O. Ourique, E. C. Biscaia Jr and J. C. Pinto, "The use of particle swarm optimization for dynamical analysis in chemical processes," Computers & Chemical Engineering, vol. 26, no. 12, pp. 1783–1793, December 2002.

[6] J. Barrera, J. J. Flores and C. Fuerte-Esquivel, "Generating complete bifurcation diagrams using a dynamic environment particle swarm optimization algorithm," Journal of Artificial Evolution and Applications, vol. 2008, 745694, December 2008.

[7] H. Matsushita and T. Saito, "Application of particle swarm optimization to parameter search in dynamical systems," NOLTA, IEICE, vol. 2, no. 4, pp. 458–471, October 2011.

[8] H. Matsushita, H. Kurokawa, and T. Kousaka, "Period doubling bifurcation point detection strategy with nested layer particle swarm optimization," International Journal of Bifurcation and Chaos, vol. 27, no. 7, 2017.

[9] H. Matsushita, H. Kurokawa, and T. Kousaka, "Saddle- node bifurcation parameter detection strategy with nested-layer particle swarm optimization," Chaos, Solitons and Fractals, vol. 119, pp. 126–134, February 2019.

[10] H. Matsushita, H. Kurokawa, and T. Kousaka, "Bifurcation analysis by particle swarm optimization," NOLTA, IEICE, vol. 11, no. 4, pp. 391–408, October 2020.

[11] T. Gotoh, H. Kurokawa, H. Matsushita, T. Kousaka: "Derivation of Local Bifurcation Points in Autonomous Systems by Using Particle Swarm Optimization," Transactions of the Institute of Systems, Control and Information Engineers, Vol. 36, No. 5, pp. 121–129, May 2023.

[12] T. Hasegawa, "pnlpso-rust," https://github.com/tm- hsgw/pnlpso-rust, 2023.

[13] T. Hasegawa, H. Matsushita, T. Kousaka, and H. Kurokawa, "Modified parallel nested-layer particle swarm optimization algorithm for fast bifurcation point detection and its software implementation," NOLTA, IEICE, vol.E14-N, no.2, pp. 308-318, April. 2023.

[14] M. Hénon, "A two-dimensional mapping with a strange attractor," Communications in Mathematical Physics, 50, pp. 69–77, February 1976.

[15] H. Kitajima, H. Kawakami, "Cascade of Period-Doubling and Neimark-Sacker Bifurcations," IEICE Trans. A, Vol.J80-A, No.3, pp.491–498, March 1997.(in Japanese)

[16] T. Ueta and G. Chen, "Bifurcation analysis of Chen's Equation," International Journal of Bifurcation and Chaos, Vol.10, No. 8, pp. 1917–1931, August 2000.

[17] K. Fujimoto, M. Musashi, and T. Yoshinaga, "Discrete-time dynamic image segmentation system," Electron. Lett, vol. 44, no. 12, pp. 727–729, June 2008.

[18] T. Hasegawa, H. Matsushita, T. Kousaka, and H. Kurokawa, "Bifurcation point detection with parallel nested layer particle swarm optimization," NOLTA, IEICE, vol.E13-N, no.2, pp.312–317, April 2022.