A parallel optimization algorithm for predictive control of marine vessel

A N Borisov¹, J L Siek¹

¹ Department of Automatic Control Systems and Onboard Computer, St. Petersburg State Marine Technical University, 3, Locmanskaya st., St. Petersburg, 190121, Russia

E-mail: bor_fond93@mail.ru

Abstract. Control of marine moving objects is a complex task, which requires considering the nonlinearity of the control object, as well as random wind-wave disturbances. One of the modern strategies is the model predictive control. The article investigates the possibilities of parallel computing technologies to solve the problem of finding the optimal sequence of control vectors in predictive control of a marine vessel. The basic concepts and mathematical model of control systems with prediction are given. Then the optimization problem is formulated. The use of SMP-systems and parallel algorithms is substantiated. Then we describe one of the most common parallel programming technologies - OpenMp. A parallel version of the conditional optimization algorithm is presented, as well as the quality functional used in the control. The results of modeling in a multi-core system with shared memory are presented. In conclusion, further directions of research, as well as possible ways to improve performance are proposed. All presented models are implemented in C++.

1. Introduction
Currently, various algorithms of optimal, adaptive control, especially for such marine objects as ships and boats, essentially non-stationary and nonlinear, are becoming more and more widespread. One of the options for such systems is model predictive control that has received significant development in the last decade. The main idea of the method is to predict the behavior of the control object on various input action [1,2,3].

2. Problem formulation
Let the mathematical model of the marine vessel has the form:

\[ \dot{x}(t) = f(t, x(t), u(t)), \quad x(0) = x_0, \quad y(t) = Cx(t) \] (1)

The vectors \( x(t) \in E^n, u(t) \in E^m, y(t) \in E^r \) represent the current state of the object, the control vector, and the observation, respectively. Then the predictive model will look like:

\[ x[i+1] = f(x[i], u[i]), \quad i = k + j, \quad j = 0,1,2... \]

\[ y[i] = Cx[i] \] (2)

The predictive model (2) is initiated at the initial clock cycle \( j = 0 \) by the current state of the control object, and allows you to approximately predict its behavior. Then the final sequence of vectors
\[ x[i], \quad (i = k + 1, \ldots, k + P) \] - calculated by the system (2), is called the motion prediction of the real object with the prediction horizon \( P \).

Let us consider the quality control function [2]:

\[
J_i = J_i(x, \bar{u})
\]

\[
x = (x[k + 1], x[k + 2], \ldots, x[k + P]) \in E^{np}.
\]

\[
\bar{u} = (u[k], u[k + 1], \ldots, u[k + P - 1]) \in E^{np}.
\]

Then the control scheme will take the form:

- The current state of the marine vessel is measured or estimated \( x[k] \).
- A sequence of control vectors is selected that optimizes the movement of the model (2) according to the function (3).
- From the found vector only the first control action is used.
- For the following clock cycles, the operations are repeated.

Thus, the principle of predictive control can be formulated to form a control action in the system, in which the predicted output variable is approaching the desired one.

According to the scheme, to calculate the control action on each clock cycle of the algorithm, it is necessary to solve the optimization problem. From a real-time implementation perspective, calculations must be performed within a single clock cycle. This requirement is a significant obstacle to the implementation of predictive models in the control of marine moving objects with small time constants [1].

Ways to solve these difficulties are to reduce the size of the optimization problem and to develop faster algorithms. The use of parallel computing systems and technologies is considered promising in the task of improving performance [4-7]. A wide class of such systems represent shared memory architectures (SMP). In SMP-systems all processors have simultaneous access to data, which makes it relatively easy to implement parallel processing, in addition, there are no performance losses for inter-process communication. However, such architecture suffers from problems of cache coherence and the race condition between processors.

One of the programming technologies of such systems is for example OpenMP. Parallelization can be represented as the following scheme. The sequential code is split into several threads by calling a special Directive. Additional threads process their part of the data while the master thread waits for its completion and ensures synchronization. When a special Directive is reached, all additional threads are removed from the system and execution continues in sequential mode.

In solving the optimization problem for predictive control of a marine vessel during computer simulation, a parallel version of the conditional optimization algorithm by the complex Box method was used [8]. This algorithm makes it possible to search for the global minimum of a multiextremal function in the presence of explicit restrictions.

3. Algorithm

Step 1. Initializing source data: prediction horizon - \( P \), the multiplication factor of the complex - \( s \), explicit constraints on the control action \( low_j \leq u_j \leq up_j \), \( j = 1 \ldots P \), number of compute nodes - \( N \).

Step 2. Construction of a complex of \( k = sP \) random sequences of control actions \( \bar{u}[q] \), \( q = 1 \ldots k \) is in the domain given by constraints. Each sequence is calculated as follows:

\[
u_{[q]} = low_j + r(up_j - low_j), \quad j = 1 \ldots P, \quad q = 1 \ldots k, \quad \text{where} \quad r \text{- random value between 0 and 1.}
\]

Step 3. Sorting complex points \( \bar{u}[q] \), \( q = 1 \ldots k \) according to the increasing function \( J_q(\bar{u}) : J(u[1]) \leq J_q(u[2]) \leq \ldots \leq J_q(u[k]) \). Calculation of \( J_{\min} = J(u[1]) \).
Step 4. Determination of centre of gravity \( k - N \) points - \( \mathbf{u}^0 \) and \( J_\phi(\mathbf{u}^0) \). \( \mathbf{u}^0 = \left( \sum_{i=1}^{kN} \mathbf{u}[i] \right) (k - N)^{-1} \).

Step 5. The calculation of \( N \) reflected points - of the respective worst \( \mathbf{u}[k - N + t] \): \( \mathbf{u}'[k - N + t] = \mathbf{u}^0 - \alpha \left( \mathbf{u}[k - N + t] - \mathbf{u}^0 \right) \), reflection coefficient \( \alpha > 1 \).

Step 6. If \( J(\mathbf{u}^0) < J(\mathbf{u}[k - N]) \), then go to step 7. Otherwise, the centre of gravity \( \mathbf{u}^0 \) is replaced with the best point of the complex \( \mathbf{u}[1] : \mathbf{u}^0 = \mathbf{u}[1] \).

Step 7. Checking for improvement of the reflected points is carried out. For each of the \( N \) worst points of the complex \( \mathbf{u}[k - N + t] \), \( 1 \leq t \leq N \), an improvement program is run on a separate compute node. The main node reports the value of the \( N \) worst points of the complex \( \mathbf{u} \), the corresponding value - \( J(\mathbf{u}) \) and the reflected point - \( \mathbf{u}' \), as well as the centre of gravity of \( k - N \) the best points \( \mathbf{u}^0 \). If \( J(\mathbf{u}') \geq J(\mathbf{u}[k - N + id]) \), where \( id \) - the number of the compute node, then the reflected point \( \mathbf{u}' \) moves half the distance between \( \mathbf{u}' \) and \( \mathbf{u}^0 \): \( \mathbf{u}' = 0.5(\mathbf{u}' + \mathbf{u}^0) \) and the process returns to step 7. If \( J(\mathbf{u}') < J(\mathbf{u}[k - N + id]) \), then the worst point \( \mathbf{u}[k - N + id] \) replaced by \( \mathbf{u}' \) and the complex is sorted according to the increase of the target function.

Step 8. The value of the function at the best point is estimated - \( J(\mathbf{u}[1]) \) and if \( J(\mathbf{u}[1]) = J_{\text{min}} \), that is, the smallest value is not reduced, then the process returns to step 4. Otherwise, a convergence test is performed. The square of the standard deviation of the target function values \( \sigma^2 \) and maximum distance between points of the complex \( d_{\text{max}} \) is calculated. If \( \sigma^2 < \varepsilon_1 \) and \( d_{\text{max}} < \varepsilon_2 \), where \( \varepsilon_1 \) and \( \varepsilon_2 \) - pre-set accuracy, then the search procedure is complete, otherwise proceed to step 3.

The parallel version of the algorithm includes simultaneous improvement of \( K \) worst points of the complex, where \( K \) – number of compute nodes. The algorithm diagram is shown in figure 1.

**Figure 1.** Algorithm diagram.
4. Simulation
With predictive control, the function of the integral error that is projected to \( P \) steps forward may take the form of:

\[
J_k = \sum_{i=0}^{P} e(k+i)^2 + \rho \sum_{i=0}^{P} (u(k+i)-u(k+i-1))^2.
\]  

(4)

Here \( e(k) \) - the error of the system output, \( \rho \) - the contribution of the control signal change to the total cost function \( J_k \).

The optimization module receives the target trajectory on the clock cycle by \( P \) cycles forward, and if it is not present, then \( P \) times duplicates the value of the current setpoint and uses it as the target trajectory. During one control cycle, the optimization module inputs a series of different actions to the model \( \hat{u}(k+t) \) and obtains different variants of the system behavior, calculating the cost function for them \( J_k \). Thus, the best control strategy of the object is calculated on this cycle, the first control signal is used at its input, and the strategy is recalculated again.

According to Flynn's taxonomy, our algorithm has SIMD parallelism, so it can be implemented relatively easily in SMP architecture. The simulation was carried out on a computing system with shared memory. The main goal of the experiment was to compare the execution time of the algorithm on a different number of computational nodes with different multiplication factors of the complex \( s \). The simulation results are shown in figure 2.

![Speedup of the algorithm](image)

**Figure 2.** Speedup of the algorithm.

5. Conclusion
With the sequential use of a different number of computing nodes, the maximum possible speedup reached 3.62. However, already on 6 threads, a sharp decline in performance was observed. This effect is due to the peculiarities of SMP-architectures when accessing shared memory and the presence of code that cannot be parallelized. Further development of the method involves the development of an algorithm for use in systems with distributed memory (MPP). On such systems, hundreds of compute nodes can run simultaneously, and the potential speedup is much higher.
6. Acknowledgments
The reported study was funded by the Ministry of Science and Higher Education of the Russian Federation, according to research project No. 2.5464.2017.

References
[1] Camacho E F and Bordons C 2004 Model Predictive Control (London: Springer-Verlag)
[2] Skrzypczyk K and Mellado M 2017 Vehicle navigation in populated areas using predictive control with environmental uncertainty handling Archives of Control Sciences 27(2) 351
[3] Shao-lei Zhou, Yu-hang Kang, Hong-de Dai and Zhou 2016 Multi-UAVs Formation Autonomous Control Method Based on RQPSO-FSM-DMPC Mathematical Problems in Engineering vol 2016 14 p
[4] Borisov A N and Siek J L 2018 Visualization of the underwater scene in the parallel computing system Marine Intelligent Technologies 2 119
[5] Dębski R 2016 An adaptive multi-spline refinement algorithm in simulation based sailboat trajectory optimization using onboard multi-core computer systems International Journal of Applied Mathematics and Computer Science 26(2) 351
[6] Fryza T, Svobodova J, Adamec F, Marsalek R and Prokopec J 2012 Overview of parallel platforms for common high performance computing Radioengineering 21(1) 436
[7] Kang S J, Lee S Y, Lee K M 2015 Performance comparison of OpenMp, MPI, and MapReduce in practical problems Advances in Multimedia
[8] Box M J 1965 A new method of constrained optimization and a comparison with other methods The Computer Journal 8 42