Defense Threat Reduction Agency
8725 John J. Kingman Road, MS
6201 Fort Belvoir, VA 22060-6201

DTRA-TR-16-17

TECHNICAL REPORT

Multiagent Swarm Based Application Software Development for Optimal Defense Strategy Synthesis of Geospatial Physical Networks in Networked Environments

October 2018

HDTRA1-13-1-0048

Qing Hui

Prepared by:
University of Nebraska-Lincoln
Department of Mechanical Engineering
Lincoln, NE 68588

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
00-10-2018

2. REPORT TYPE
Technical

3. DATES COVERED *(From - To)*
July 1, 2014 - August 29, 2015

4. TITLE AND SUBTITLE
Multiagent Swarm Based Application Software Development for Optimal Defense Strategy Synthesis of Geospatial Physical Networks in Networked Environments

5a. CONTRACT NUMBER

5b. GRANT NUMBER
HDTRA 1-13-1-0048

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Qing Hui

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Department of Mechanical Engineering, Texas Tech University, Lubbock, TX 79409-1021
Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588-0511

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Threat Reduction Agency
8725 John J. Kingman Road, STOP 6201
Fort Belvoir, VA 22060-6201

10. SPONSOR/MONITOR'S ACRONYM(S)
DTRA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
DTRA-TR-16-17

12. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution Statement A. Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This project is to develop an algorithm software package that can be used to synthesize optimal defense strategies for large geospatial physical networks under WMD attacks, particularly for solving cascading resilience based, dynamic multi-objective optimization.

15. SUBJECT TERMS
Swarm optimization, multiagent systems, distributed computation, software development, optimal defense strategies, physical ne

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
					Paul Tandy
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	UU	83	**19b. TELEPHONE NUMBER** *(include area code)*
Unclassified	Unclassified	Unclassified			571-616-4663

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

UNIT CONVERSION TABLE

U.S. customary units to and from international units of measurement[*]

U.S. Customary Units	Multiply by → ← Divide by[†]		International Units
Length/Area/Volume			
inch (in)	2.54	$\times 10^{-2}$	meter (m)
foot (ft)	3.048	$\times 10^{-1}$	meter (m)
yard (yd)	9.144	$\times 10^{-1}$	meter (m)
mile (mi, international)	1.609 344	$\times 10^{3}$	meter (m)
mile (nmi, nautical, U.S.)	1.852	$\times 10^{3}$	meter (m)
barn (b)	1	$\times 10^{-28}$	square meter (m^2)
gallon (gal, U.S. liquid)	3.785 412	$\times 10^{-3}$	cubic meter (m^3)
cubic foot (ft^3)	2.831 685	$\times 10^{-2}$	cubic meter (m^3)
Mass/Density			
pound (lb)	4.535 924	$\times 10^{-1}$	kilogram (kg)
unified atomic mass unit (amu)	1.660 539	$\times 10^{-27}$	kilogram (kg)
pound-mass per cubic foot (lb ft^{-3})	1.601 846	$\times 10^{1}$	kilogram per cubic meter (kg m^{-3})
pound-force (lbf avoirdupois)	4.448 222		newton (N)
Energy/Work/Power			
electron volt (eV)	1.602 177	$\times 10^{-19}$	joule (J)
erg	1	$\times 10^{-7}$	joule (J)
kiloton (kt) (TNT equivalent)	4.184	$\times 10^{12}$	joule (J)
British thermal unit (Btu) (thermochemical)	1.054 350	$\times 10^{3}$	joule (J)
foot-pound-force (ft lbf)	1.355 818		joule (J)
calorie (cal) (thermochemical)	4.184		joule (J)
Pressure			
atmosphere (atm)	1.013 250	$\times 10^{5}$	pascal (Pa)
pound force per square inch (psi)	6.984 757	$\times 10^{3}$	pascal (Pa)
Temperature			
degree Fahrenheit (^oF)	$[T(^oF) - 32]/1.8$		degree Celsius (^oC)
degree Fahrenheit (^oF)	$[T(^oF) + 459.67]/1.8$		kelvin (K)
Radiation			
curie (Ci) [activity of radionuclides]	3.7	$\times 10^{10}$	per second (s^{-1}) [becquerel (Bq)]
roentgen (R) [air exposure]	2.579 760	$\times 10^{-4}$	coulomb per kilogram (C kg^{-1})
rad [absorbed dose]	1	$\times 10^{-2}$	joule per kilogram (J kg^{-1}) [gray (Gy)]
rem [equivalent and effective dose]	1	$\times 10^{-2}$	joule per kilogram (J kg^{-1}) [sievert (Sv)]

[*]Specific details regarding the implementation of SI units may be viewed at http://www.bipm.org/en/si/.
[†]Multiply the U.S. customary unit by the factor to get the international unit. Divide the international unit by the factor to get the U.S. customary unit.

2015-11-16

Final Technical Report for the Defense Threat Reduction Agency

Fundamental Research Award No. HDTRA1-13-1-0048

Multiagent Swarm Based Application Software Development for Optimal Defense Strategy Synthesis of Geospatial Physical Networks in Networked Environments

Qing Hui
Principal Investigator
Department of Electrical and Computer Engineering
University of Nebraska-Lincoln
Lincoln, NE
qing.hui@unl.edu

for

Paul Tandy
Grant Officer's Representative
Fundamental Research Program
Counter-Weapons of Mass Destruction
Defense Threat Reduction Agency
Fort Belvoir, VA
paul.s.tandy.civ@mail.mil

August 27, 2015

Title: Multiagent Swarm Based Application Software Development for Optimal Defense Strategy Synthesis of Geospatial Physical Networks in Networked Environments

1 Abstract

The aim of this project is to transform some new theoretic results, developed in the Defense Threat Reduction Agency (DTRA) Basic Research Award #HDTRA1-10-1-0090, into an application software tool that can be used to synthesize optimal defense strategies for large geospatial physical networks under Weapons of Mass Destruction (WMD) attacks, particularly for solving multi-task and multiobjective optimization, and cascading resilience related dynamic optimization. This application software utilizes a new multiagent swarm based optimization algorithm as a core to provide swift, computationally reliable decision-making strategies to counter WMD in networked environments. To achieve this, we plan to develop a new multiagent swarm based algorithm for solving mixed-integer/binary nonlinear programming in networked environments, design the application software architecture with the integration of the new optimization algorithm, and construct a protocol-based algorithm for interaction between multiple real-time subsystems. The overall project develops a cost-effective technology for enhanced physical network robustness and self-healing response time to detect, predict, diagnose, prevent, and recover from WMD-induced disruptions and cascading failures.

2 Objectives

Our plan to pursue this counter-WMD research lies on two major phases. The following timetable describes our envisioned schedule and key issues that will be addressed:

Year 1 (Development of the networked Binary Hybrid Multiagent Swarm Optimization (BHMSO) algorithm to solve complex network optimization) We are interested in countering WMD applications where the sensor network has large amount of data. It will not be possible to process this information in traditional ways (e.g., object detection and tracking), and hence we must use alternative techniques such as swarm optimization algorithms and wide-field integration. Given the complex tasks that we wish to accomplish, it is likely that we will use parallel pathways that correspond to different objectives, with selection of desired objectives by a protocol-based algorithm.

Year 2 (Development of the Guarded Command Language (GCL) based algorithm and architecture of application software) Throughout the architecture, we seek to allow for multiple computing elements to be operating in parallel. This allows a highly distributed approach to algorithm design, but also drives the system to operate in an asynchronous (or at best loosely synchronized) manner. We anticipate that such asynchronous operations will occur in the inner loop, in the protocol-based algorithm and in the networked communications between real-time agents, all with potentially similar time scales.

Hence, the project can be divided into the following two major goals in two years.

Year 1: Developing the BHMSO algorithm and conducting multicore parallel implementation. The main goal in Year 1 is to develop new multiagent swarm based algorithms for solving mixed-integer nonlinear programming in networked environments. These are the specific issues to be addressed:

1. Design a new multiagent swarm based optimization algorithm and discuss its convergence issue.
2. Develop the new BHMSO algorithm based on Subtask 1 for mixed-integer nonlinear programming.
3. Evaluate the proposed algorithm by conducting some computational experiments for comparison and parallelize the proposed algorithm on multicore supercomputers in HPCC.

Year 2: Developing the GCL-based algorithm and architecture of application software. The main goal in Year 2 is to convert the proposed mathematical algorithm into the GCL-based architecture. The following issues will be addressed:

4. Design the candidate architecture of the application software using the GCL-based framework.
5. Explore design space and improvement for the possible architectures of the proposed software.
6. Analyze and verify the overall behavior of the GCL-based algorithm and software architecture.

This is the final report to this Fundamental Research Award #HDTRA1-13-1-0048. We have finished all the goals in Years 1 and 2 and discovered some interesting new results to further improve the outcome of these goals. More specifically, we have found out 1) a new bat behavior inspired cooperative optimization algorithm to further improve the convergence rate of the BHMSO algorithm, 2) a new hybridized optimization framework for fast solving large-scale constrained optimization problems, particularly in complex network systems, and 3) a new computationally efficient motion planning algorithm for mobile robots with only sonar sensors for WMD threat detection.

3 Proposed Approach

In the first year, a contraction mapping based Multiagent Coordination Optimization (MCO) algorithm, called Paracontracting Multiagent Coordination Optimization (PMCO), was proposed and implemented in a parallel computing way by embedding MATLAB built-in function parfor into PMCO. Then we rigorously analyzed the global convergence of PMCO by means of semistability theory, which is quite novel and different from the conventional sequence analysis in fixed-point optimization. The basic idea is to convert the proposed derivative-free iterative algorithm for PMCO into a discrete-time switched linear system and then discuss its semistability property. The detailed eigenvalue and eigenspace structures of switched linear matrices for this discrete-time switched linear system were explicitly derived. Based on these matrix analysis results and semistability theory, we presented two sufficient conditions for guaranteeing the global convergence of PMCO under different circumstances. Finally, numerical evaluation of the parallel PMCO algorithm was conducted by running the proposed algorithm on multicore supercomputers.

Next, we proposed a coupled spring forced MCO (CSFMCO) algorithm by considering that each particle is a spring and is coupled with the optimal solution found so far as the second abstract spring. The synergistic integration of the coupled springs, multiagent coordination, and swarm intelligence governs and navigates the new algorithm in the searching process. Numerical evaluation was done for the proposed CSFMCO algorithm by conducting comparison with other variations of PSO in the literature, which indicates that the performance of CSFMCO surpasses all the listed variations of PSO significantly. In summary, the proposed CSFMCO algorithm offers a new efficient approach to address complex, large-scale, non-convex nonlinear optimization problems which are normally hard to solve using the conventional methods.

Mixed-binary nonlinear programming (MBNP), which can be used to optimize network structure and network parameters simultaneously for network systems, has been seen widely in many applications of cyber-physical network systems, especially for WMD related problems. However, it is quite challenging to develop efficient algorithms to solve it practically. On the other hand, swarm intelligence based optimization algorithms can simulate the cooperation and interaction behaviors from social or nature phenomena to solve complex, nonconvex nonlinear problems with high efficiency. Hence, motivated by this observation, we proposed a class of new computationally efficient algorithms called binary coupled spring forced multiagent coordination optimization (BCSFMCO) to solve MBNP problems, by exploiting the chaos-like behavior of two-mass two-spring mechanical systems to improve the ability of algorithmic exploration and thus to fast solve MBNP problems. Together with the continuous version of CSFMCO, a binary version of CSFMCO and a switching version between continuous and binary versions were presented. Moreover, to numerically illustrate our proposed algorithms, a formation control problem and resource allocation problem for cyber-physical networks under WMD attacks were investigated by using the proposed algorithms.

Inspired by speed-up and speed-down (SUSD) mechanism observed by the fish swarm avoiding light, an SUSD strategy was proposed to develop new swarm intelligence based optimization algorithms to enhance the accuracy and efficiency of swarm optimization algorithms. By comparing with the global best solution, each particle adaptively speeds up and speeds down towards the best solution. Specifically, a new directed speed term is added to the original particle swarm optimization (PSO) algorithm or other PSO variations. Due to the SUSD mechanism, the algorithm shows a great improvement of the accuracy and convergence rate compared with the original PSO and other PSO variations. The numerical evaluation was conducted by solving recent benchmark functions in IEEE CEC 2013.

Model predictive control (MPC) is a heuristic control strategy to find a consequence of best controllers during each finite-horizon regarding to certain performance functions of a dynamic system. MPC involves two main operations: estimation and optimization. Due to high complexity of the performance functions in WMD problems, such as, nonlinear, non-convex, large-scale objective functions, the optimization algorithms for MPC must be capable of handling those problems with both computational efficiency and accuracy. Multiagent coordination optimization (MCO) was a recently developed heuristic algorithm by embedding multiagent coordination into swarm intelligence to accelerate the searching for the optimal solution in the particle swarm optimization (PSO) algorithm. With only some elementary operations, the MCO algorithm can obtain the best solution extremely fast, which is especially necessary to solve the online optimization problems in MPC. Therefore, we proposed an MCO based MPC strategy to enhance the performance of the MPC controllers when addressing non-convex large-scale nonlinear problems for large-scale physical networks. Moreover, as an application, the network resource balanced allocation problem under WMD attacks was numerically illustrated by the MCO based MPC strategy.

For the last several months we have been developing code to address the network vulnerability problem related to electrical power networks. Our starting point is a standard power distribution model. This model utilizes a directed graph (captured in the matrix A below) to model the power distribution of power systems under stress. The HPCC at

TTU has installed several nodes with dual Xeon Phi. We have experimented with porting our code to this environment. All the relevant algorithms developed above have been converted into pseudo code or C code in our developed software package to DTRA. The prototype of this software package has been tested for a preliminary version.

In the second year, we have developed and refined the GCL-based algorithm as well as the architecture of application software. In particular, we have designed the candidate architecture of the application software using the GCL-based framework and explored design space and improvement for the possible architectures of the proposed software. Although there are a lot of research on large-scale unconstrained optimization (e.g., with 100 to 1000 variables) and small-scale constrained optimization (e.g., with 10 to 30 variables) using nature-inspired algorithms (e.g., evolutionary algorithms and swarm intelligence algorithms), there is no known nature-inspired algorithm developed for large-scale constrained optimization. Here we combined a cooperative co-evolutionary particle swarm optimization (CCPSO) algorithm with the ε-constrained method for solving large-scale real-valued constrained optimization problems. The ε-CCPSO framework was proposed, and three different algorithms based on the framework, i.e., ε-CCPSOd, ε-CCPSOw and ε-CCPSOw2, were developed. The proposed algorithms compare favorably against the state-of-the-art constrained optimization algorithm which is the ε-constrained method adopted by a differential evolutionary (DE) algorithm, i.e., ε-DEag, on large-scale problems. The experimental results further suggest that ε-CCPSOw2 with adaptive improvement detection technique is highly competitive compared to the other algorithms considered in this work for solving large-scale, real-valued constrained optimization problems. Apart from this, we have analyzed and verified the overall behavior of the GCL-based algorithm and software architecture.

Next, the motion planning problem for an AmigoBot with only sonar sensors was addressed in this year. A line segment based map was firstly constructed incrementally from readings of sonar sensors. Map building algorithms were proposed to reduce the size of the map while maintaining the complete and accurate information about the environment. Then the Voronoi diagram of the line segment based map was generated with Fortune's sweep line algorithm. A shortest accessible path from the initial configuration to the goal was searched from the Voronoi diagram with Dijkstra's algorithm. The notion of Clearance was defined to guarantee the safety of the path encountered with obstacles. Finally, a path tracking control law with the line-of-sight approach was designed to follow the reference path. Simulation was conducted to verify the designed motion planning approach and the results showed that it is effective for such an AmigoBot to accomplish the given task in unknown environments.

Finally, in order to strengthen the communication of interconnected, multi-area power systems for state estimation and distributed computation, we proposed a new method to classify more buses in the overlap areas in the partitioning process. A modified algorithm called maximum overlap fuzzy c-means (FCM) was proposed to address this issue by incorporating bridgeness term and between-cluster measure term into the objective function. Then we considered using B-spline curves to represent membership degrees instead of fuzzy membership degrees. This B-spline based maximum overlap FCM will generate more overlap areas between subsystems. Furthermore, it can also reduce the number of iteration steps comparing with the maximum overlap FCM.

4 Accomplishments for Year 1 (09/01/2013-06/30/2014)

The following research accomplishments were achieved over the first year duration of this project.

4.1 Paracontracting Multiagent Coordination Optimization

Swarm intelligence based optimization is a class of successful, heuristic computational intelligence methods that optimize nonlinear objective functions iteratively by trying to improve a candidate solution with regards to a given measure of quality. Among many of these optimization algorithms, Ant Colony Optimization (ACO) [1] and Particle Swarm Optimization (PSO) [2] are the most popular ones. Compared with other swarm intelligence and derivative-based optimization schemes, PSO has advantages of only requiring elementary operations such as addition and multiplication, and only consisting of two non-derivative based iterative update laws to search for optima. Moreover, PSO does not require that the optimization problem be differentiable as is required by classic optimization solvers such as gradient descent. Thus, PSO can also be used for optimization problems that are partially irregular, noisy, changing over time, etc. Additionally, PSO can be easily implemented in a parallel way to fully utilize high performance, large-scale parallel computing capability of multicore computers, which will significantly improve computational efficiency. Hence, PSO and its variants have been widely used in many applications, see for example, power system vulnerability analysis [3], optimizing arbitrary low-rate denial-of-quality attacks [4,5], and model predictive control [6,7].

There are many variant PSO algorithms in the literature since the first PSO algorithm was proposed by [2], see, for instance [8–13]. All these PSO variants focus either on some highly mathematical skills or on nature-inspired structures to improve their performance, lacking the fundamental understanding of how these algorithms work for *general* problems. Thus, to address this issue, we need to look at the swarm intelligence algorithm design from a new perspective since the traditional way of looking to *natural* network systems appearing in nature for inspiration does not provide a satisfactory answer. Multiagent Coordination Optimization (MCO) is a new algorithm inspired by swarm intelligence and consensus protocols for multiagent coordination [14–16]. Unlike the standard PSO, this new algorithm is an optimization technique based not only on swarm intelligence [17], but also on cooperative control of autonomous agents. By adding a distributed control term and gradient-based adaptation, the convergence speed of MCO can be accelerated and the convergence time of MCO can be shortened compared with the existing techniques due to the finite-time convergence property of certain hybrid and switched cooperative control laws [18, 19]. Moreover, this new algorithm will be more suitable to distributed and parallel computation for solving large-scale physical network optimization problems by means of high performance computing facilities.

In this work, we consider a generalized form of MCO called Paracontracting Multiagent Coordination Optimization (PMCO) by including a paracontracting matrix operation in MCO. It is well-known that contraction mapping has been a fundamental tool to prove the convergence of iterative optimization algorithms. By incorporating some contracting matrix terms in our original MCO algorithm, we aim to bound accumulated numerical errors in every iteration for MCO and to enhance the convergence rate of MCO. To show the parallel computing nature of PMCO, we first implement the proposed PMCO in a parallel computing way by introducing MATLAB built-in function `parfor` into PMCO. Next, we present a global convergence result of PMCO by means of paracontraction and semistability theory.

The MCO algorithm with static graph topology, proposed in [14] to solve a given optimization problem $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, where $f:\mathbb{R}^n\to\mathbb{R}$, $\mathbb{R}^n$ denotes the set of n-dimensional real column vectors, and $\mathbb{R}$ denotes the set of real numbers, can be described in a vector form as follows:

$$\mathbf{v}_k(t+1) = \mathbf{v}_k(t)+\eta\sum_{j\in\mathcal{N}^k}(\mathbf{v}_j(t)-\mathbf{v}_k(t))+\mu\sum_{j\in\mathcal{N}^k}(\mathbf{x}_j(t)-\mathbf{x}_k(t))+\kappa(\mathbf{p}(t)-\mathbf{x}_i(t)), \tag{1}$$

$$\mathbf{x}_k(t+1) = \mathbf{x}_k(t)+\mathbf{v}_k(t+1), \tag{2}$$

$$\mathbf{p}(t+1) = \begin{cases}\mathbf{p}(t)+\kappa(\mathbf{x}_{\min}(t)-\mathbf{p}(t)), & \text{if } \mathbf{p}(t)\notin\mathcal{Z},\\ \mathbf{x}_{\min}(t), & \text{if } \mathbf{p}(t)\in\mathcal{Z},\end{cases} \tag{3}$$

where $k=1,\ldots,q$, $t\in\overline{\mathbb{Z}}_+=\{0,1,2,\ldots\}$, $\mathbf{v}_k(t)\in\mathbb{R}^n$ and $\mathbf{x}_k(t)\in\mathbb{R}^n$ are the velocity and position of particle k at iteration t, respectively, $\mathbf{p}(t)\in\mathbb{R}^n$ is the position of the global best value that the swarm of the particles can achieve so far, η, μ, and κ are three scalar random coefficients which are usually selected in uniform distribution in the range $[0,1]$, $\mathcal{Z}=\{\mathbf{y}\in\mathbb{R}^n : f(\mathbf{x}_{\min})<f(\mathbf{y})\}$, and $\mathbf{x}_{\min}=\arg\min_{1\le k\le q} f(\mathbf{x}_k)$. Note that here $f(\cdot)$ is *not* necessarily continuous and we assume that the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$ has solutions.

Next, we use time-dependent algebraic graph-related notation to describe our generalized algorithm to MCO. More specifically, let $\mathcal{G}(t)=(\mathcal{V},\mathcal{E}(t),\mathcal{A}(t))$ denote a *node-fixed dynamic directed graph* (or *node-fixed dynamic digraph*) with the set of vertices $\mathcal{V}=\{v_1,v_2,\ldots,v_n\}$ and $\mathcal{E}(t)\subseteq\mathcal{V}\times\mathcal{V}$ represent the set of edges, where $t\in\overline{\mathbb{Z}}_+$. The time-varying matrix $\mathcal{A}(t)\in\mathbb{R}^{n\times n}$ with nonnegative adjacency elements $a_{i,j}(t)$ serves as the weighted adjacency matrix. The node index of $\mathcal{G}(t)$ is denoted as a finite index set $\mathcal{N}=\{1,2,\ldots,n\}$. An edge of $\mathcal{G}(t)$ is denoted by $e_{i,j}(t)=(v_i,v_j)$ and the adjacency elements associated with the edges are positive. We assume $e_{i,j}(t)\in\mathcal{E}(t)$ if and only if $a_{i,j}(t)=1$ and $a_{i,i}(t)=0$ for all $i\in\mathcal{N}$. The set of neighbors of the node v_i is denoted by $\mathcal{N}^i(t)=\{v_j\in\mathcal{V}:(v_i,v_j)\in\mathcal{E}(t), j=1,2,\ldots,|\mathcal{N}|, j\neq i\}$, where $|\mathcal{N}|$ denotes the cardinality of $\mathcal{N}$. The degree matrix of a node-fixed dynamic digraph $\mathcal{G}(t)$ is defined as $\Delta(t)=[\delta_{i,j}(t)]_{i,j=1,2,\ldots,|\mathcal{N}|}$, where $\delta_{i,j}(t)=\begin{cases}\sum_{j=1}^{|\mathcal{N}|}a_{i,j}(t), & \text{if } i=j,\\ 0, & \text{if } i\neq j.\end{cases}$ The *Laplacian matrix* of the node-fixed dynamic digraph $\mathcal{G}(t)$ is defined by $L(t)=\Delta(t)-\mathcal{A}(t)$. If $L(t)=L^{\mathrm{T}}(t)$, where $(\cdot)^{\mathrm{T}}$ denotes the transpose operation, then $\mathcal{G}(t)$ is called a *node-fixed dynamic undirected graph* (or simply *node-fixed dynamic graph*). From now on we use short notation $L_t,\mathcal{G}_t,\mathcal{N}_t^i$ to denote $L(t),\mathcal{G}(t),\mathcal{N}^i(t)$, respectively.

We generalize (1) in twofold. First, we extend (1) to the dynamic graph case where $\mathcal{N}^k$ becomes $\mathcal{N}^k(t)=\mathcal{N}_t^k$. Second, we further extend (1) to the *paracontracting* form with dynamic graph topology sequence $\{\mathcal{G}_t\}_{t=0}^{\infty}$ given by

$$\begin{aligned}\mathbf{v}_k(t+1) = {}& P(t)\mathbf{v}_k(t)+\eta P(t)\sum_{j\in\mathcal{N}_t^k}(\mathbf{v}_j(t)-\mathbf{v}_k(t))+\mu P(t)\sum_{j\in\mathcal{N}_t^k}(\mathbf{x}_j(t)-\mathbf{x}_k(t))\\ &+\kappa P(t)(\mathbf{p}(t)-\mathbf{x}_i(t)),\end{aligned} \tag{4}$$

where $P(t) \in \mathbb{R}^{n\times n}$ is a paracontracting matrix, and $\mathcal{N}^k(t) = \mathcal{N}_t^k$ represents the node-fixed dynamic or time-varying graph topology. Thus, this new algorithm is called Paracontracting Multiagent Coordination Optimization (PMCO). Clearly if $P(t) = I_n$ for every $t \in \overline{\mathbb{Z}}_+$, then (4) boils down to the form of (1). Here we use a specific dynamic neighborhood structure called Grouped Directed Structure (GDS) [20] to generate a neighboring set sequence $\{\mathcal{N}_t^k\}_{t=0}^{\infty}$. In this structure, we divide all particles into different groups at every time instant. In each group, particles have the strongly-connected graphical structure. The information exchange between the two groups is directed. For example, in Figure 1, we divide the 6 particles into two groups, one contains particles 1 and 2 called "all-information" group and the other includes particles 3–6 called "half information" group. In each group, the graphical structure is strongly-connected. Particles 1 and 2 can know the information of all the other particles and particles 3–6 cannot know the information of particles 1 and 2. With this technique, if the information from the particle 1 or 2 is not desirable then we can limit the information inside of the group of particles 1 and 2. Meanwhile, if the information from the particle in "all-information" group is desirable then it is highly possible to lead the particles in "all-information" group to a global optimum. The effectiveness of this technique and its parallel computation has been numerically verified in [20,21].

The idea of how to pick the GDS at the beginning is proposed as follows. First we find the minimum value and maximum value of all the particles in the first dimension. Then we break up the interval from the minimum to the maximum into equal sized parts. These become the neighborhoods. Hence, if we want N neighborhoods, then there will be N intervals. Then we go through each particle and place it in the neighborhood, or interval, based on its first dimension value. One can think of it as an N number of cages and each particle is placed in the cage that corresponds to its first dimension value. For example, if you want 5 neighborhoods, and the first dimension minimum value is 1 and the maximum value is 6, then you would have intervals: [1,2), [2,3), [3,4), [4,5), [5,6]. If a particle has a value of 3.5 in its first dimension, then it would be placed in the third cage. The neighborhoods will not necessarily have an equal number of particles.

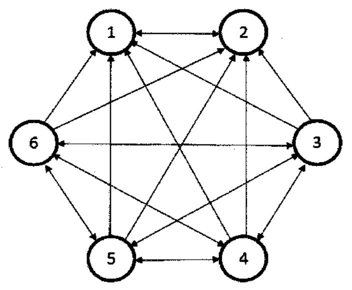


Figure 1: Grouped directed structure.

The introduction of $P(t)$ in (4), motivated by the *projection method* [22, 23] in approximation theory, is to guarantee that PMCO will stay inside the search region by means of nonexpansive operation and to guarantee the semi-convergence of PMCO. In particular, if the position of a particle runs beyond the boundary of the search space at time t, then we choose an appropriate $P(t) \neq I_n$ so that the velocity of the particle can be slowed down and the candidate solution can be pulled back into the search space. A natural question arising from (2)–(4) is the following: Can we always guarantee the semi-convergence of (2)–(4) for a given optimization problem $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$? Here semi-convergence means that all the limits $\lim_{t\to\infty} \mathbf{x}_k(t)$, $\lim_{t\to\infty} \mathbf{v}_k(t)$, and $\lim_{t\to\infty} \mathbf{p}(t)$ exist for every $k = 1, \ldots, q$. The second part of this work tries to answer this question by giving a sufficient condition to guarantee the semi-convergence of (2)–(4). To this end, the basic idea is to convert the iterative algorithm into a discrete-time switched linear system and then discuss its semistability property [24]. More specifically, we consider the discrete-time switched linear system given by

$$X(t+1) = W_{\sigma(t)}X(t), \quad t \in \overline{\mathbb{Z}}_+, \tag{5}$$

where $X(t) \in \mathbb{R}^m$, $\sigma : \overline{\mathbb{Z}}_+ \to \Sigma$ is a piecewise constant switching signal, and Σ is an index set. It is important to note that the PMCO algorithm given by (2), (3), and (4) can be rewritten as the compact form (5) by defining $X(t) = [\mathbf{x}_1^{\mathrm{T}}(t), \ldots, \mathbf{x}_q^{\mathrm{T}}(t), \mathbf{v}_1^{\mathrm{T}}(t), \ldots, \mathbf{v}_q^{\mathrm{T}}(t), \mathbf{p}^{\mathrm{T}}(t)]^{\mathrm{T}}$ for some state-dependent switching signal σ. Thus, the semi-convergence analysis of the PMCO algorithm can be converted into the semi-convergence analysis of (5) by means of semistability theory.

Paracontraction is a nonexpansive property for a class of linear operators which can be used to guarantee semi-convergence of linear iterations [25]. The following definition due to [25] gives the notion of paracontracting matrices.

Definition 4.1 ([25]). *Let $W \in \mathbb{R}^{n\times n}$. W is called paracontracting if for any $x \in \mathbb{R}^n$, $Wx \neq x$ is equivalent to $\|Wx\| < \|x\|$.*

Recall that $A \in \mathbb{R}^{n\times n}$ is called *nontrivially discrete-time semistable* [26] if A is discrete-time semistable and $A \neq I_n$. It follows from Proposition 3.2 of [23] that if A is paracontracting, then A is nontrivially discrete-time semistable. The converse part is not true in general even if $\|A\| \leq 1$. Such a counterexample is given by $A = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ 0 & 0 \end{bmatrix}$. Now a more fundamental question is: What are the real necessary and sufficient conditions to connect both matrix paracontraction and semistability? This is an important question since it can give us an alternative way to characterize paracontraction without referring to its definition, which is difficult to verify in practice. We answer this question by giving the following new result.

Theorem 4.1. *Let $W \in \mathbb{R}^{q\times q}$ and* $\mathrm{spec}(W) = \{\lambda_1, \ldots, \lambda_r\}$, *where r denotes the number of distinct eigenvalues for W. Then W is nontrivially discrete-time semistable, $\|Wx\| \leq \|x\|$ for any $Wx \neq \lambda_i x$ and every $i = 1, \ldots, r$, and* $\ker(W^{\mathrm{T}}W - I_q) = \ker((W - I_q)^{\mathrm{T}}(W - I_q) + (W - I_q)^2)$ *if and only if W is paracontracting.*

Next, we replace $\|Wx\| \leq \|x\|$ for any $Wx \neq \lambda_i x$ and every $i = 1, \ldots, r$, and $\ker(W^{\mathrm{T}}W - I_q) = \ker((W - I_q)^{\mathrm{T}}(W - I_q) + (W - I_q)^2)$ in Theorem 4.1 by some new eigenvalue-related conditions which can be checked numerically. Recall from [27, p. 608] that the Hölder-induced norm $\|\cdot\|$ for W is defined by $\|W\| = \max_{x\in\mathbb{R}^q\setminus\{\mathbf{0}_{q\times 1}\}} \|Ax\|/\|x\|$.

Corollary 4.1. *Let $W \in \mathbb{R}^{q\times q}$. Then W is nontrivially discrete-time semistable, $\|W\| \leq 1$, and* $\mathrm{rank}(W^{\mathrm{T}}W - I_q) = \mathrm{rank}((W - I_q)^{\mathrm{T}}(W - I_q) + (W - I_q)^2)$
$= \mathrm{rank}\begin{bmatrix} W^{\mathrm{T}}W - I_q & (W - I_q)^{\mathrm{T}}(W - I_q) + (W^{\mathrm{T}} - I_q)^2 \end{bmatrix}$ *if and only if W is paracontracting, where* $\mathrm{rank}(A)$ *denotes the rank of A.*

Motivated by Corollary 4.1 above and the proof of Theorem 5 in [28], we have the following semi-convergence result for sequence matrix functions.

Theorem 4.2. *Let Σ be a finite index set. For every $p \in \Sigma$, assume that $W_p : \mathcal{E} \to \mathbb{R}^{q\times q}$ is continuous, where $\varnothing \neq \mathrm{int}\,\mathcal{E} \subseteq \mathcal{E} \subseteq \mathbb{R}^n$ and $\mathrm{int}\,\mathcal{E}$ denotes the interior of $\mathcal{E}$. Furthermore, assume that for every $p \in \Sigma$ and every $z \in \mathcal{E}$, $W_p(z)$ is nontrivially discrete-time semistable, $\|W_p(z)\| \leq 1$, and* $\mathrm{rank}((W_p(z)-I_q)^{\mathrm{T}}(W_p(z)-I_q)+W_p^{\mathrm{T}}(z)-I_q+W_p(z)-I_q) = \mathrm{rank}((W_p(z)-I_q)^{\mathrm{T}}(W_p(z)-I_q)+(W_p(z)-I_q)^2) = \mathrm{rank}\begin{bmatrix} (W_p(z) - I_q)^{\mathrm{T}}(W_p(z) - I_q) + W_p^{\mathrm{T}}(z) - I_q + W_p(z) - I_q \\ (W_p(z) - I_q)^{\mathrm{T}}(W_p(z) - I_q) + (W_p(z) - I_q)^2 \end{bmatrix}$. *Then for any compact subset $\mathcal{M} \subset \mathcal{E}$ and any sequence $\{x_k\}_{k=0}^{\infty}$ defined by $x_{k+1} = W_{p_k}(z_k)x_k$, where $p_k \in \Sigma$ and $z_k \in \mathcal{M}$,* $\lim_{k\to\infty} x_k$ *exists.*

As an application, in this part we present some theoretic results on the global semi-convergence of PMCO by means of semistability-based matrix paracontraction techniques. In particular, we view the randomized PMCO algorithm as a discrete-time switched linear system and then use Theorem 4.7 to rigorously show its global semi-convergence. Here we will present the main result for the global semi-convergence of PMCO.

Before we state the main result, first we define a series of matrices $A_k^{[j]}$, $A_{\mathrm{c}k}$, and B_k, $j = 1, \ldots, q$, $k = 0, 1, 2, \ldots$, throughout the work as follows:

$$A_k^{[j]} = \begin{bmatrix} \mathbf{0}_{nq\times nq} & I_{nq} & \mathbf{0}_{nq\times n} \\ -\mu_k L_k \otimes P_k - \kappa_k(I_q \otimes P_k) & -\eta_k L_k \otimes P_k & \kappa_k \mathbf{1}_{q\times 1} \otimes P_k \\ \kappa_k E_{n\times nq}^{[j]} & \mathbf{0}_{n\times nq} & -\kappa_k I_n \end{bmatrix}, \tag{6}$$

$$A_{\mathrm{c}k} = \begin{bmatrix} -\mu_k L_k \otimes P_k - \kappa_k I_q \otimes P_k & -\eta_k L_k \otimes P_k & \kappa_k \mathbf{1}_{q\times 1} \otimes P_k \\ \mathbf{0}_{nq\times nq} & \mathbf{0}_{nq\times nq} & \mathbf{0}_{nq\times n} \\ \mathbf{0}_{n\times nq} & \mathbf{0}_{n\times nq} & \mathbf{0}_{n\times n} \end{bmatrix}, \tag{7}$$

$$B_k^{[j]} = \begin{bmatrix} \mathbf{0}_{nq\times nq} & h_k I_{nq} & \mathbf{0}_{nq\times n} \\ -h_k\mu_k L_k \otimes P_k - h_k\kappa I_q \otimes P_k & -h_k\eta_k L_k \otimes P_k & h_k\kappa_k \mathbf{1}_{q\times 1} \otimes P_k \\ E_{n\times nq}^{[j]} & \mathbf{0}_{n\times nq} & -I_n \end{bmatrix}, \tag{8}$$

where $\otimes$ denotes the Kronecker product, $L_k \in \mathbb{R}^{q\times q}$ denotes the Laplacian matrix of a node-fixed dynamic digraph $\mathcal{G}_k$, $\mathbf{1}_{m\times n}$ denotes the $m \times n$ matrix whose entries are all ones, and $E_{n\times nq}^{[j]} \in \mathbb{R}^{n\times nq}$ denotes a block-matrix whose jth block-column is I_n and the rest block-elements are all zero matrices, i.e., $E_{n\times nq}^{[j]} = [\mathbf{0}_{n\times n}, \ldots, \mathbf{0}_{n\times n}, I_n, \mathbf{0}_{n\times n}, \ldots, \mathbf{0}_{n\times n}]$, $j = 1, \ldots, q$. Next, define $D_{11} = h_k^4[\mu_k(L_k\otimes P_k)+\kappa_k(I_q\otimes P_k)][\mu_k(L_k\otimes P_k)^{\mathrm{T}}+\kappa_k(I_q\otimes P_k)^{\mathrm{T}}]+h_k^2 I_{nq}+h_k^4\eta_k^2(L_k\otimes P_k)(L_k\otimes P_k)^{\mathrm{T}} - h_k^3\eta_k(L_k\otimes P_k)^{\mathrm{T}} - h_k^3\eta_k(L_k\otimes P_k)+h_k^4\kappa_k^2(\mathbf{1}_{q\times 1}\otimes P_k)(\mathbf{1}_{q\times 1}\otimes P_k)^{\mathrm{T}} - h_k^2\mu_k(L_k\otimes P_k)^{\mathrm{T}} - h_k^2\mu_k(L_k\otimes P_k) - h_k^2\kappa_k(I_q\otimes P_k)^{\mathrm{T}} - h_k^2\kappa_k(I_q\otimes P_k)$, $D_{12} = h_k^3[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)][\mu_k(L_k \otimes P_k)^{\mathrm{T}} + \kappa_k(I_q \otimes P_k)^{\mathrm{T}}] - h_k^2\eta_k(L_k \otimes P_k)^{\mathrm{T}} + h_k^3\eta_k^2(L_k \otimes P_k)(L_k \otimes P_k)^{\mathrm{T}} + h_k^3\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k)(\mathbf{1}_{q\times 1} \otimes P_k)^{\mathrm{T}} - h_k\mu_k(L_k \otimes P_k)^{\mathrm{T}} - h_k\kappa_k(I_q \otimes P_k)^{\mathrm{T}} + h_k I_{nq} - h_k^2\eta_k(L_k \otimes P_k)$, $D_{13} = -h_k^3\kappa_k[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)](E_{n\times nq}^{[j]})^{\mathrm{T}} - h_k^3\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k) + h_k\kappa_k(E_{n\times nq}^{[j]})^{\mathrm{T}} + h_k^2\kappa_k(\mathbf{1}_{q\times 1} \otimes P_k)$, $D_{22} = h_k^2[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)][\mu_k(L_k \otimes P_k)^{\mathrm{T}} + \kappa_k(I_q \otimes P_k)^{\mathrm{T}}] + h_k^2\eta_k^2(L_k \otimes P_k)(L_k \otimes P_k)^{\mathrm{T}} + h_k^2\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k)(\mathbf{1}_{q\times 1} \otimes P_k)^{\mathrm{T}} - h_k\eta_k(L_k \otimes P_k) - h_k\eta_k(L_k \otimes P_k)^{\mathrm{T}}$, and $D_{23} = -h_k^2\kappa_k[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)](E_{n\times nq}^{[j]})^{\mathrm{T}} - h_k^2\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k) + h_k\kappa_k(\mathbf{1}_{q\times 1} \otimes P_k)$. Finally, for any matrix $A \in \mathbb{R}^{nq\times nq}$, define $\mathscr{E}^{r,s}_{l,m}(A) = (\mathbf{g}_r \otimes \mathbf{e}_s)^{\mathrm{T}} A (\mathbf{g}_l \otimes \mathbf{e}_m)$ and $\mathscr{R}^{r,s}(A) = \sum_{l=1}^{q} \sum_{m=1}^{n} \mathscr{E}^{r,s}_{l,m}(A)$, where $[\mathbf{g}_1, \ldots, \mathbf{g}_q] = I_q$ and $[\mathbf{e}_1, \ldots, \mathbf{e}_n] = I_n$. The next result gives the *exact* value to the 2-norm or maximum singular value for $I_{2nq+n} + h_k A_k^{[j]} + h_k^2 A_{\mathrm{c}k}$ under some conditions on positive $\eta_k, \mu_k, \kappa_k, h_k$.

Lemma 4.1. *Consider the matrices $A_k^{[j]}$ and $A_{\mathrm{c}k}$ defined by (6) and (7) respectively, where $\eta_k, \mu_k, \kappa_k, h_k > 0$, $j = 1, \ldots, q$, and $k = 0, 1, 2, \ldots$. Assume that $h_k\kappa_k \leq 1$ and for every $r = 1, \ldots, q$ and every $s = 1, \ldots, n$, the following inequalities hold:*

$$\begin{aligned}
&2h_k\kappa_k(1-h_k\kappa_k) - \mathscr{R}^{r,s}(D_{11}+D_{12}) \geq -nq(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{11}),\\
&\min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{11})\} + \min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12})\}), \qquad (9)\\
&-2h_k\kappa_k(1-h_k\kappa_k)\mathscr{R}^{r,s}(D_{11}+D_{12}) - \mathscr{R}^{r,s}(D_{13}D_{13}^{\mathrm{T}} + D_{13}D_{23}^{\mathrm{T}}) \geq -2nqh_k\kappa_k(1-h_k\kappa_k)\\
&(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{11}), \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12})\} + \min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{11})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12})\}) - nq(\min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{13}D_{13}^{\mathrm{T}}), \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{13}D_{23}^{\mathrm{T}})\} + \min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{13}D_{13}^{\mathrm{T}})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{13}D_{23}^{\mathrm{T}})\}), \qquad (10)\\
&2h_k\kappa_k(1-h_k\kappa_k) - \mathscr{R}^{r,s}(D_{22}+D_{12}^{\mathrm{T}}) \geq -nq(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{22}),\\
&\min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12}^{\mathrm{T}})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{22})\} + \min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12}^{\mathrm{T}})\}), \qquad (11)\\
&-2h_k\kappa_k(1-h_k\kappa_k)\mathscr{R}^{r,s}(D_{22}+D_{12}^{\mathrm{T}}) - \mathscr{R}^{r,s}(D_{23}D_{23}^{\mathrm{T}} + D_{23}D_{13}^{\mathrm{T}}) \geq -2nqh_k\kappa_k(1-h_k\kappa_k)\\
&(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{22}), \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12}^{\mathrm{T}})\} + \min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{22})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{12}^{\mathrm{T}})\}) - nq(\min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{23}D_{23}^{\mathrm{T}}), \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{23}D_{13}^{\mathrm{T}})\} + \min\{0,\\
&\min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs} \mathscr{E}^{r,s}_{l,m}(D_{23}D_{23}^{\mathrm{T}})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n} \mathscr{E}^{r,s}_{l,m}(D_{23}D_{13}^{\mathrm{T}})\}). \qquad (12)
\end{aligned}$$

Then for every $j = 1, \ldots, q$ and every $k = 0, 1, 2, \ldots$, $\|I_{2nq+n} + h_k A_k^{[j]} + h_k^2 A_{\mathrm{c}k}\| = 1$.

Analogously, one can give the *exact* value to the 2-norm or maximum singular value for $I_{2nq+n} + B_k^{[j]} + h_k^2 A_{\mathrm{c}k}$ under some conditions on $\eta_k, \mu_k, \kappa_k, h_k$. Define $F_{11} = h_k^4[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)][\mu_k(L_k \otimes P_k)^{\mathrm{T}} + \kappa_k(I_q \otimes P_k)^{\mathrm{T}}] + h_k^2 I_{nq} + h_k^4\eta_k^2(L_k \otimes P_k)(L_k \otimes P_k)^{\mathrm{T}} - h_k^3\eta_k(L_k \otimes P_k)^{\mathrm{T}} - h_k^3\eta_k(L_k \otimes P_k) + h_k^4\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k)(\mathbf{1}_{q\times 1} \otimes P_k)^{\mathrm{T}} - h_k^2\mu_k(L_k \otimes P_k)^{\mathrm{T}} - h_k^2\mu_k(L_k \otimes P_k) - h_k^2\kappa_k(I_q \otimes P_k)^{\mathrm{T}} - h_k^2\kappa_k(I_q \otimes P_k)$, $F_{12} = h_k^3[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)][\mu_k(L_k \otimes P_k)^{\mathrm{T}} + \kappa_k(I_q \otimes P_k)^{\mathrm{T}}] - h_k^2\eta_k(L_k \otimes P_k)^{\mathrm{T}} + h_k^3\eta_k^2(L_k \otimes P_k)(L_k \otimes P_k)^{\mathrm{T}} + h_k^3\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k)(\mathbf{1}_{q\times 1} \otimes P_k)^{\mathrm{T}} - h_k\mu_k(L_k \otimes P_k)^{\mathrm{T}} - h_k\kappa_k(I_q \otimes P_k)^{\mathrm{T}} + h_k I_{nq} - h_k^2\eta_k(L_k \otimes P_k)$, $F_{13} = -h_k^2[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)](E_{n\times nq}^{[j]})^{\mathrm{T}} + (E_{n\times nq}^{[j]})^{\mathrm{T}}$, $F_{22} = h_k^2[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)][\mu_k(L_k \otimes P_k)^{\mathrm{T}} + \kappa_k(I_q \otimes P_k)^{\mathrm{T}}] + h_k^2\eta_k^2(L_k \otimes P_k)(L_k \otimes P_k)^{\mathrm{T}} + h_k^2\kappa_k^2(\mathbf{1}_{q\times 1} \otimes P_k)(\mathbf{1}_{q\times 1} \otimes P_k)^{\mathrm{T}} - h_k\eta_k(L_k \otimes P_k) - h_k\eta_k(L_k \otimes P_k)^{\mathrm{T}}$, and $F_{23} = -h_k^2\kappa_k[\mu_k(L_k \otimes P_k) + \kappa_k(I_q \otimes P_k)](E_{n\times nq}^{[j]})^{\mathrm{T}}$.

Lemma 4.2. *Consider the matrices $B_k^{[j]}$ and $A_{\mathrm{c}k}$ defined by (8) and (7) respectively, where $\eta_k, \mu_k, \kappa_k, h_k > 0$, $j =$*

$1,\ldots,q$, *and* $k=0,1,2,\ldots$. *Assume that for every* $r=1,\ldots,q$ *and every* $s=1,\ldots,n$, *the following inequalities hold:*

$$\begin{aligned}
&\mathscr{R}^{r,s}(F_{11}+F_{12}) \leq nq(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{11}), \min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{12})\}\\
&+\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{11})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{12})\}), \qquad (13)\\
&\mathscr{R}^{r,s}(F_{13}F_{13}^{\mathrm{T}}+F_{13}F_{23}^{\mathrm{T}}) \leq nq(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{13}F_{13}^{\mathrm{T}}),\\
&\min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{13}F_{23}^{\mathrm{T}})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{13}F_{13}^{\mathrm{T}})\}\\
&+\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{13}F_{23}^{\mathrm{T}})\}), \qquad (14)\\
&\mathscr{R}^{r,s}(F_{22}+F_{12}^{\mathrm{T}}) \leq nq(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{22}), \min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{12}^{\mathrm{T}})\}\\
&+\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{22})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{12}^{\mathrm{T}})\}), \qquad (15)\\
&\mathscr{R}^{r,s}(F_{23}F_{23}^{\mathrm{T}}+F_{23}F_{13}^{\mathrm{T}}) \leq nq(\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{23}F_{23}^{\mathrm{T}}),\\
&\min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{23}F_{13}^{\mathrm{T}})\} + \min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n,lm\neq rs}\mathscr{E}^{r,s}_{l,m}(F_{23}F_{23}^{\mathrm{T}})\}\\
&+\min\{0, \min_{l=1,\ldots,q,m=1,\ldots,n}\mathscr{E}^{r,s}_{l,m}(F_{23}F_{13}^{\mathrm{T}})\}). \qquad (16)
\end{aligned}$$

Then for every $j=1,\ldots,q$ *and every* $k=0,1,2,\ldots$, $\|I_{2nq+n}+B_k^{[j]}+h_k^2A_{\mathrm{c}k}\|=1$.

It follows from Lemma 4.5 of [29] that 0 is an eigenvalue of $A_k^{[j]}$ for every $j=1,\ldots,q$ and every $k\in\overline{\mathbb{Z}}_+$. Next, we further investigate some relationships of the null spaces between a row-addition transformed matrix of $A_k^{[j]}$ and $A_k^{[j]}$ itself in order to unveil an important property of semisimplicity of this eigenvalue 0.

Lemma 4.3. *Consider* $A_k^{[j]}+h_kA_{\mathrm{c}k}$, $j=1,\ldots,q$, $k=0,1,2,\ldots$, *where* $A_k^{[j]}$ *is defined by (6),* $A_{\mathrm{c}k}$ *is defined by (7),* $\mu_k,\kappa_k,\eta_k\geq 0$, $h_k>0$, *and* $P_k\in\mathbb{R}^{n\times n}$ *is paracontracting. If* $\kappa_k\neq 0$ *and* $\mu_k\neq 0$, *then* $\operatorname{rank}(A_k^{[j]}+h_kA_{\mathrm{c}k})=2nq-(q-1)(n-\operatorname{rank}(P_k))$ *for every* $j=1,\ldots,q$, $k\in\overline{\mathbb{Z}}_+$. *In this case, for every* $j=1,\ldots,q$ *and every* $k\in\overline{\mathbb{Z}}_+$, *0 is a semisimple eigenvalue of* $A_k^{[j]}+h_kA_{\mathrm{c}k}$ *if and only if* $\operatorname{rank}(P_k)=n$. *Alternatively, 0 is a semisimple eigenvalue of* $B_k^{[j]}+h_k^2A_{\mathrm{c}k}$ *if and only if* $\operatorname{rank}(P_k)=n$.

Remark 4.1. *It follows from Lemma 4.3 that for every* $j=1,\ldots,q$, *1 is a semisimple eigenvalue of* $I_{nq}+h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k}$, *where* $\mu_k,\kappa_k,\eta_k\geq 0$ *and* $h_k>0$, *if and only if* $\kappa_k\neq 0$ *and* $\operatorname{rank}(P_k)=n$, $k\in\overline{\mathbb{Z}}_+$. *A similar conclusion holds for* $I_{nq}+B_k^{[j]}+h_k^2A_{\mathrm{c}k}$. ♦

Now we have the main result for the semi-convergence of PMCO.

Theorem 4.3. *Consider the following discrete-time switched linear model to describe the iterative process for PMCO:*

$$\begin{aligned}
x_i[k+1] &= x_i[k]+h_kv_i[k+1], \quad x_i[0]=x_{i0}, && (17)\\
v_i[k+1] &= P_kv_i[k]+h_k\eta_kP_k\sum_{r\in\mathcal{N}_k^i}(v_r[k]-v_i[k])+h_k\mu_kP_k\sum_{r\in\mathcal{N}_k^i}(x_r[k]-x_i[k])\\
&\quad +h_k\kappa_kP_k(p[k]-x_i[k]), \quad v_i[0]=v_{i0}, && (18)\\
p[k+1] &= p[k]+h_k\kappa_k(x_j[k]-p[k]), \quad p[k]\notin\mathcal{Z}_p, \quad p[0]=p_0, && (19)\\
p[k+1] &= x_j[k], \quad p[k]\in\mathcal{Z}_p, \quad k=0,1,2,\ldots, && (20)
\end{aligned}$$

where $i=1,\ldots,q$, $x_i\in\mathbb{R}^n$, $v_i\in\mathbb{R}^n$, $p\in\mathbb{R}^n$, $P_k\in\mathbb{R}^{n\times n}$, $h_k\in\Omega\subseteq[0,\infty)$, μ_k,η_k,κ_k *are randomly selected in* Ω, $\mathcal{Z}_p=\{p\in\mathbb{R}^n: f(x_j)<f(p)\}$, *and* $x_j=\arg\min_{1\leq i\leq q}f(x_i)$. *Assume that for every* $k\in\overline{\mathbb{Z}}_+$ *and every* $j=1,\ldots,q$:

H1) $P_k\in\mathbb{R}^{n\times n}$ *is discrete-time semistable,* $\|P_k\|\leq 1$, $\operatorname{rank}(P_k^{\mathrm{T}}P_k-I_n)=\operatorname{rank}((P_k-I_n)^{\mathrm{T}}(P_k-I_n)+(P_k-I_n)^2)=\operatorname{rank}\begin{bmatrix}P_k^{\mathrm{T}}P_k-I_n & (P_k-I_n)^{\mathrm{T}}(P_k-I_n)+(P_k-I_n)^2\end{bmatrix}$, *and* $\operatorname{rank}(P_k)=n$.

H2) $\sup_{k\in\overline{\mathbb{Z}}_+}(h_k+\frac{\lambda+\lambda^*}{|\lambda|^2})<0$ *for every* $\lambda\in\{-1,-\frac{h_k^2\kappa_k}{2}\pm\frac{1}{2}((h_k^2\kappa_k)^2-4h_k^2\kappa_k)^{1/2},-\kappa_k,-\frac{\kappa_k(1+h_k)}{2}\pm\frac{1}{2}(\kappa_k^2(1+h_k)^2-4\kappa_k)^{1/2},-\frac{\kappa_kh_k}{2}\pm\frac{1}{2}(\kappa_k^2h_k^2-4\kappa_k)^{1/2},\lambda_0,\lambda_1,\lambda_2\in\mathbb{C}:\forall\frac{\lambda_0^2+\kappa_kh_k\lambda_0+\kappa_k}{\eta_k\lambda_0+\mu_kh_k\lambda_0+\mu_k}\in\operatorname{spec}(-L_k)\backslash\{0\},\forall\frac{\lambda_1^2+\kappa_kh_k^2\lambda_1+\kappa_kh_k^2}{\eta_kh_k\lambda_1+\mu_kh_k^2\lambda_1+\mu_kh_k^2}\in\operatorname{spec}(-L_k)\backslash\{0\},\lambda_2^3+(1+h_k^2\kappa_k)\lambda_2^2+(2h_k^2\kappa_k-h_k\kappa_k)\lambda_2+h_k^2\kappa_k=0\}$;

H3) $h_k\kappa_k \leq 1$ *and for every* $r = 1, \ldots, q$ *and every* $s = 1, \ldots, n$, *(9)–(12) and (13)–(16) hold.*

H4) $\mathrm{rank}((h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k}) + (h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k})^{\mathrm{T}} + h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k}) = \mathrm{rank}((h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k}) + (h_kA_k^{[j]} + h_k^2A_{\mathrm{c}k})^2) = \mathrm{rank} \begin{bmatrix} (h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k})+(h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k})^{\mathrm{T}}+h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k} \\ (h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k})+(h_kA_k^{[j]}+h_k^2A_{\mathrm{c}k})^2 \end{bmatrix}$ *and* $\mathrm{rank}((B_k^{[j]} + h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(B_k^{[j]} + h_k^2A_{\mathrm{c}k}) + (B_k^{[j]} + h_k^2A_{\mathrm{c}k})^{\mathrm{T}} + B_k^{[j]} + h_k^2A_{\mathrm{c}k}) = \mathrm{rank}((B_k^{[j]} + h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(B_k^{[j]} + h_k^2A_{\mathrm{c}k}) + (B_k^{[j]} + h_k^2A_{\mathrm{c}k})^2) = \mathrm{rank} \begin{bmatrix} (B_k^{[j]}+h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(B_k^{[j]}+h_k^2A_{\mathrm{c}k})+(B_k^{[j]}+h_k^2A_{\mathrm{c}k})^{\mathrm{T}}+B_k^{[j]}+h_k^2A_{\mathrm{c}k} \\ (B_k^{[j]}+h_k^2A_{\mathrm{c}k})^{\mathrm{T}}(B_k^{[j]}+h_k^2A_{\mathrm{c}k})+(B_k^{[j]}+h_k^2A_{\mathrm{c}k})^2 \end{bmatrix}$.

H5) $0 \notin \Omega$ *is compact.*

Then $x_i[k] \to p^\dagger$, $v_i[k] \to \mathbf{0}_{n\times 1}$, *and* $p[k] \to p^\dagger$ *as* $k \to \infty$ *for every* $x_{i0} \in \mathbb{R}^n$, $v_{i0} \in \mathbb{R}^n$, $p_0 \in \mathbb{R}^n$, *and every* $i = 1, \ldots, q$, *where* $p^\dagger \in \mathbb{R}^n$ *is some constant vector.*

Remark 4.2. *While Theorem 4.3 states that* $\lim_{k\to\infty} x_i[k]$ *exists, it does not necessarily mean that* $\min_{1\leq i\leq q} f(\lim_{k\to\infty} x_i[k]) = \min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$. *Hence, the convergence issue of the iterative process given by (17)–(20) to a global optimum of the minimization problem* $\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$ *remains an open problem. On the other hand, if we know some information about the cost function f when using (17)–(20), then we may have some clue on this optimality issue. To see this, we first assume that f is continuous and all the conditions in Theorem 4.8 hold. Then it follows from Theorem 4.3 that* $\lim_{k\to\infty} x_i[k] = p^\dagger$ *and* $\lim_{k\to\infty} p[k] = p^\dagger$. *Since* $\lim_{k\to\infty} p[k] = p^\dagger$, *it follows from (19) and (20) that* $\lim_{k\to\infty} x_j[k] = p^\dagger$. *By definition of* x_j, *we have* $p^\dagger = \lim_{k\to\infty} \arg\min_{1\leq i\leq q} f(x_i[k])$. *Next, since by assumption f is continuous, it follows that* $f(p^\dagger) = f(\lim_{k\to\infty} x_j[k]) = \lim_{k\to\infty} f(x_j[k]) = \liminf_{k\to\infty} f(x_j[k]) \leq \liminf_{k\to\infty} f(x_i[k]) \leq \lim_{k\to\infty} f(x_i[k]) = f(\lim_{k\to\infty} x_i[k]) = f(p^\dagger)$ *for every* $i = 1, \ldots, q$. *Thus,* $\lim_{k\to\infty} \arg\min_{1\leq i\leq q} f(x_i[k]) = \lim_{k\to\infty} x_i[k]$ *for every* $i = 1, \ldots, q$, *which implies that* $\lim_{k\to\infty} \min_{1\leq i\leq q} f(x_i[k]) = \lim_{k\to\infty} f(x_j[k]) = f(\lim_{k\to\infty} x_j[k]) = f(\lim_{k\to\infty} x_i[k]) = f(p^\dagger)$. ♦

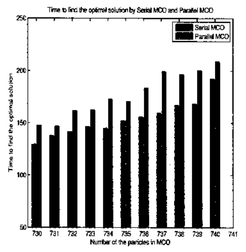
In order to show the performance of the parallel PMCO, we conduct a comparison evaluation between the standard PSO, serial PMCO, and parallel PMCO. In particular, we use the following 8 test functions chosen from [13, 30] to evaluate the proposed algorithm.

- Sphere function: $f(x) = \sum_{i=1}^{n} x_i^2$. The test area is usually restricted to the hypercube $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.
- Rosenbrock's valley: $f(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$. The test area is usually restricted to the hypercube $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 1$.
- Rastrigin function: $f(x) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]$. The test area is usually restricted to the hypercube $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0.
- Griewank function: $f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$. The test area is usually restricted to the hypercube $-600 \leq x_i \leq 600$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.
- Ackley function: $f(x) = -20\exp(-0.2 \times \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$. The test area is usually restricted to the hypercube $-32.768 \leq x_i \leq 32.768$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.
- De Jong's f4 function: $f(x) = \sum_{i=1}^{n}(ix_i^4)$. The test area is usually restricted to the hypercube $-20 \leq x_i \leq 20$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.
- Zakharov function: $f(x) = \sum_{i=1}^{n} x_i^2 + (0.5ix_i)^2 + (0.5ix_i)^4$. The test area is usually restricted to the hypercube $-10 \leq x_i \leq 10$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.
- Levy function: $f(x) = \sin^2(\pi x_1) + (x_n - 1)^2(1 + \sin^2(2\pi x_n)) - \sum_{i=1}^{n-1}(x_i - 1)^2(1 + 10\sin^2(\pi x_i + 1))$. The test area is usually restricted to the hypercube $-10 \leq x_i \leq 10$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 1$.

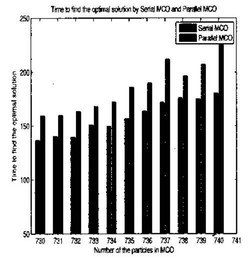
We first evaluate the computational time of the parallel PMCO for different test functions. Specifically, eight 2.8 GHz cores equipped supercomputers in the High Performance Computing Center at Texas Tech University are used to run the parallel PMCO algorithm for all the 8 benchmark functions in which the search areas and dimensions of objective functions are listed before with $n = 30$. We choose each graph $\mathcal{G}_k$ in PMCO to be a complete graph and $P_k = I_n$, $k \in \overline{\mathbb{Z}}_+$, for simplicity. In this case, PMCO just collapses to MCO. The simulation results on the computational time of the serial MCO and parallel MCO for solving the optimization problems are shown in Fig. 2–5. From the

Table 1: Numerical Comparison Between PSO, Serial MCO, and Parallel MCO for the Eight Test Functions

Function	Min			Max			Median			Average		
	PSO	Serial MCO	Parallel MCO	PSO	Serial MCO	Parallel MCO	PSO	Serial MCO	Parallel MCO	PSO	Serial MCO	Parallel MCO
Sphere	9.525E1	3.3E-3	3.0E-3	2.716E2	1.51E-2	1.11E-2	4.278E2	1.85E-2	1.73E-2	1.785E2	8.3E-3	7.2E-3
Rosenbrock	1.981E5	1.708E1	1.840E2	1.139E6	7.649E1	1.561E2	1.425E6	1.262E2	1.429E2	5.415E5	4.471E1	5.973E1
Rastrigin	2.802E2	1.027E2	1.252E2	7.639E2	2.585E2	2.916E2	1.125E3	4.050E2	4.030E2	4.773E3	1.687E2	1.773E2
Griewank	1.268E1	6.735E-1	4.674E-1	3.953E1	5.165	4.084	5.961E1	1.710	1.883	2.294E1	8.003E-1	7.894E-1
Ackley	8.551	1.541	2.355	1.292E1	3.792	5.744	2.414E1	6.987	7.955	1.110E1	2.889	3.477
De Jong's f4	3.206E2	1.565E-6	7.156E-7	1.328E3	1.905E-5	1.793E-5	1.428E3	2.097E-5	1.553E-5	6.048E2	7.7106E-6	6.346E-6
Zakharov	6.288E4	1.394	1.101	2.938E5	5.746	4.084	1.689E+5	5.165	7.310	7.307E4	2.484	2.476
Levy	1.041E2	2.053E1	2.483E1	4.029E2	1.813E2	9.079E2	5.867E2	7.545E1	1.300E2	2.286E2	4.690E1	5.545E1



(a) Sphere function



(b) Rosenbrock function

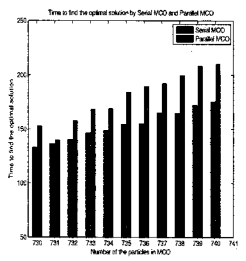
Figure 2: Computational time comparison between serial MCO and parallel MCO

simulation results, the parallel MCO algorithm can shorten the computational time about 5% to 30% compared with the serial MCO for solving those problems.

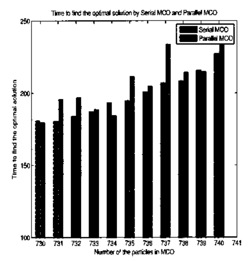
To evaluate numerical accuracy for the parallel PMCO, the statistical results of the optimal values obtained from the standard PSO, serial PMCO and parallel PMCO algorithms are compared numerically. Similarly, the search areas and dimensions of objective functions are listed before with $n = 30$. Again, we choose each graph $\mathcal{G}_k$ in PMCO to be a complete graph and $P_k = I_n$, $k \in \overline{\mathbb{Z}}_+$. In this case, PMCO just boils down to MCO. The maximum of the objective values, the minimum of the objective values, the average of objective value, and the median objective values are compared in Table 1. Based on these results, it follows that the serial MCO and parallel MCO algorithms are more accurate for solving those problems than the PSO algorithm.

4.2 Coupled Spring Forced Multiagent Coordination Optimization: Algorithm and Performance Evaluation

The development of swarm intelligence in optimization has been witnessed during the last decade, some famous swarm intelligence optimization algorithms became more and more popular, such as the particle swarm optimization (PSO) algorithm [2]. In the PSO algorithm, each particle is assigned to a position in the solution space and a velocity randomly. Each particle has a memory of its current best value and the corresponding current best position. In addition, every

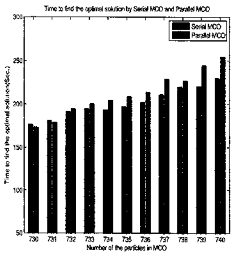


(a) Rastrigin function

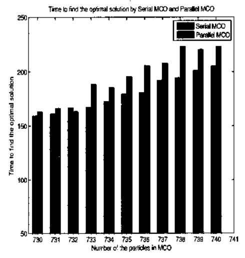


(b) Griewank function

Figure 3: Computational time comparison between serial MCO and parallel MCO

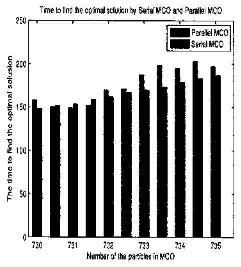


(a) Ackley function

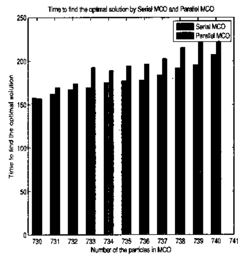


(b) De Jone's f4 function

Figure 4: Computational time comparison between serial MCO and parallel MCO



(a) Zakharov function



(b) Levy function

Figure 5: Computational time comparison between serial MCO and parallel MCO

particle in the swarm can know the global best value among all particles and the corresponding global best position. During every iteration, the velocity of each particle is updated so that the particle is guided by the current best position and the global best position of the particle stochastically. A large variety of applications have been tackled using PSO or its variations as a heuristic optimization algorithm, see for example, power system vulnerability analysis [3] and optimizing arbitrary low-rate denial-of-quality attacks [4].

Multiagent coordination optimization (MCO) [14–16, 31] is a new algorithm inspired by swarm intelligence and consensus protocols for multiagent coordination. Different from the standard PSO algorithm, the particles in the MCO algorithm embed the multiagent coordination terms in the update formula, and share the velocity and position information with neighbors through a communication topology, which is frequently seen in consensus or synchronization research for multiagent coordination problems [18, 19, 32–36]. By adding a distributed control term and gradient-based adaptation, the convergence speed of MCO can be accelerated and the convergence time of MCO can be shortened compared with the existing techniques due to the finite-time convergence property of certain hybrid and switched cooperative control laws [18, 19].

The standard coupled spring model is fully introduced in [37]. The movement between two springs has been discussed under different circumstances and the graphical representation of two springs' trajectories has also been demonstrated. Inspired by the synchronization and diversity of the coupled springs motions, in this work, we propose a new *coupled spring forced MCO* (CSFMCO) algorithm. Specifically, we treat each particle in the MCO algorithm as a spring and the optimal solution found so far as the other spring. Governed by coupled springs motions, the next positions and velocities for both two springs will be discretely updated, and then the swarm intelligence logic will be switched on to update the local optimal solution found by the particle itself and also the best optimal solution found among all the particles. The iterative searching processes come to end when the stop criteria are satisfied. Numerical evaluation has been conducted compared with other variations of PSO, and the proposed CSFMCO algorithm surpasses those algorithms when solving unconstrained nonlinear optimization problems given by standard benchmark functions.

Based not only on swarm intelligence [17] which simulates the bio-inspired behavior, but also on cooperative control of multiple agents [18, 32–34], multiagent coordination optimization (MCO) proposed in [14–16, 31] starts with a set of random solutions for agents that can communicate with each other. The agents then move through the solution space based on the evaluation of their cost and neighbor-to-neighbor rules inspired by multiagent consensus protocols [18, 19, 32–34]. As the algorithm progresses, the agents will accelerate towards individuals with better cost

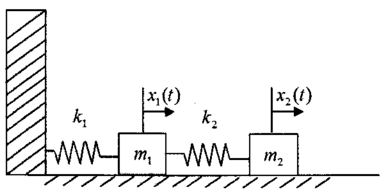


Figure 6: A Coupled Spring System

values. The new update formulas for particles under MCO are shown as follows:

$$\begin{aligned} \mathbf{v}_{k+1}^i &= \mathbf{v}_k^i + \check{\eta} \sum_{j \in \mathcal{N}_i} (\mathbf{v}_k^j - \mathbf{v}_k^i) \\ &\quad + \check{\mu} \sum_{j \in \mathcal{N}_i} (\mathbf{x}_k^j - \mathbf{x}_k^i) + \check{\kappa}(p - \mathbf{x}_k^i), \qquad (21) \\ \mathbf{x}_{k+1}^i &= \mathbf{x}_k^i + \mathbf{v}_{k+1}^i, \quad i = 1, \ldots, q, \qquad (22) \end{aligned}$$

where $k = 0, 1, 2, \ldots$, $\mathbf{v}_k^i \in \mathbb{R}^n$, $\mathbf{x}_k^i \in \mathbb{R}^n$, $\check{\eta} \sim U(0, 2\eta)$, $\check{\mu} \sim U(0, 2\mu)$, $\check{\kappa} \sim U(0, 2\kappa)$, $\eta, \mu, \kappa > 0$ are random numbers, $U(\cdot, \cdot)$ denotes the uniform distribution with certain lower and upper bounds, $\mathcal{N}_i$ is the neighbor set of particle i, and $p = \arg\min_{1 \le i \le q, 0 \le s \le k} f(\mathbf{x}_s^i)$.

It follows from (21) and (1) that the difference between MCO and PSO lies in the velocity update. The new velocity update (21) for MCO accelerates the search for the desired solution by use of the neighboring agents' information and two new search directions. The first new search direction $\sum_{j \in \mathcal{N}_i} (\mathbf{v}_k^j - \mathbf{v}_k^i)$ accelerates the update for the agents' velocities to converge to the same value that is not pre-described, while the second new search direction $\sum_{j \in \mathcal{N}_i} (\mathbf{x}_k^j - \mathbf{x}_k^i)$ accelerates the convergence of all the neighboring agents' positions to the same one that is also not pre-described.

The coupled spring system [37] consists of two springs which is shown in Fig. 6, where two springs move in a smooth ground with masses m_1 and m_2, spring constants k_1 and k_2, and positions $x_1(t) \in \mathbb{R}$ and $x_2(t) \in \mathbb{R}$, respectively. The equations of motion of the two masses are given by

$$\begin{aligned} m_1 \ddot{x}_1 &= -k_1 x_1 - k_2 (x_1 - x_2), \qquad (23) \\ m_2 \ddot{x}_2 &= -k_2 (x_2 - x_1). \qquad (24) \end{aligned}$$

A comprehensive discussion on this coupled spring system is provided in [37]. Here we want to point out a striking similarity between the coupled spring system (23) and (24), and the iterative process (21) and (22) when $q = 2$ and $n = 1$. However, besides continuity, the big difference between these two sets of equations is that for (23) and (24), the velocity information $\dot{x}_1$ or $\dot{x}_2$ does not appear explicitly in these equations. This causes oscillatory behaviors in the dynamical system (23) and (24) (e.g., Fig. 7). By adding some terms on $\mathbf{v}_k^i$ and $\mathbf{x}_k^i$, one can reduce oscillation and make the trajectories of the system more monotonic. On the other hand, from a numerical point of view, having too many terms on $\mathbf{v}_k^i$ and $\mathbf{x}_k^i$ in (21) and (22) may cause the numerical error of (21) and (22) to accumulate to a point where the absolute values of $\mathbf{v}_k^i$ and $\mathbf{x}_k^i$ become so large that they are far beyond the limit of the stored memory, which is unacceptable for practical implementation. In this case, the iterative process may be trapped at a point where we may never get to the optima.

Motivated by the coupled spring system (23) and (24), in this work we will propose a simpler, new iterative algorithm for (21) to avoid the stagnation of (21) and (22) by dropping off *part* of the position information in (21). To our surprise, such a modification can greatly improve the performance of the original MCO algorithm as well as a large class of other PSO variations as shown by the simulation later. Next, we begin with the introduction to our new proposed algorithm and discussion on its convergence issue.

The proposed coupled spring forced MCO algorithm is described in Algorithm 1. In particular, the update formulas for the CSFMCO algorithm are given by

$$\mathbf{v}_{k+1}^i = \mathbf{v}_k^i + \check{\eta}\sum_{j\in\mathcal{N}_i}(\mathbf{v}_k^j - \mathbf{v}_k^i) - \check{\mu}\mathbf{x}_k^i + \check{\kappa}(\mathbf{p}_k - \mathbf{x}_k^i), \tag{25}$$

$$\mathbf{x}_{k+1}^i = \mathbf{x}_k^i + \mathbf{v}_{k+1}^i, \tag{26}$$

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \check{\kappa}(\mathbf{p}_k^i - \mathbf{p}_k), \quad \mathbf{p}_k \notin \mathcal{Z}_p, \quad i = 1,\ldots,q, \tag{27}$$

$$\mathbf{p}_{k+1} = \mathbf{p}_k^i, \quad \mathbf{p}_k \in \mathcal{Z}_p, \quad i = 1,\ldots,q, \tag{28}$$

where $k = 0,1,2,\ldots$, $\mathbf{v}_k^i \in \mathbb{R}^n$, $\mathbf{x}_k^i \in \mathbb{R}^n$, $\mathbf{p}_k^i = \arg\min_{1\le i\le q} f(\mathbf{x}_k^i)$, $\mathcal{Z}_p = \{\mathbf{p}\in\mathbb{R}^n : f(\mathbf{p}_k^i) < f(\mathbf{p})\}$ is called a *swarm intelligence logic set*, and $\check{\eta}\in\Omega_1\subseteq(0,2\eta]$, $\check{\mu}\in\Omega_2\subseteq(0,2\mu]$, and $\check{\kappa}\in\Omega_3\subseteq(0,2\kappa]$ are randomly selected, $\eta,\mu,\kappa>0$.

Algorithm 1 Coupled Spring Forced MCO Algorithm

```
for each agent $i = 1,\ldots,q$ do
  Initialize the agent's position with a uniformly distributed random vector: $x_i \sim U(\underline{x},\overline{x}) \in \mathbf{R}^{n\times 1}$,
  where $\underline{x}$ and $\overline{x}$ are the lower and upper boundaries of the search space;
  Initialize the agent's velocity: $v_i \sim U(\underline{v},\overline{v})$, where $\underline{v}$ and $\overline{v} \in \mathbf{R}^{n\times 1}$ are the lower and upper
  boundaries of the search speed;
  Update the agent's best known position to its initial position: $p_i \leftarrow x_i$;
  If $f(p_i) < f(p)$ update the multiagent network's best known position: $p \leftarrow p_i$.
end for
repeat
  $k \leftarrow k+1$;
  for each agent $i = 1,\ldots,q$ do
    each agent $i = 1,\ldots,q$
    Choose random parameters: $\check{\eta} \sim U(0,2\eta)$, $\check{\mu} \sim U(0,2\mu)$, $\check{\kappa} \sim U(0,2\kappa)$, $\eta,\mu,\kappa > 0$;
    Update the agent's velocity: $v_i \leftarrow v_i + \check{\eta}\sum_{j\in N_i}(v_j - v_i) - \check{\mu}x_i + \check{\kappa}(p - x_i)$;
    Update the agent's position: $x_i \leftarrow x_i + v_i$;
    for $f(x_i) < f(p_i)$ do
      Update the agent's best known position: $p_i \leftarrow x_i$;
      Update the multiagent network's best known position: $p \leftarrow p + \check{\kappa}(p_i - p)$;
      If $f(p_i) < f(p)$ update the multiagent network's best known position: $p \leftarrow p_i$;
    end for
  end for
until $k$ is large enough or the value of $f$ has small change
return $p$
```

If we consider $\mathbf{p}_k$ as a coupled abstract spring, then the above update formulas for the CSFMCO algorithm are similar to (23) in the case where m_1 and m_2 are unit. Since $\mathbf{p}_k$ is viewed as an abstract spring, the initial velocity at each iterative step will be reset to zero in the proposed algorithm. A numerical illustration of the trajectories shown for a three coupled spring particle system is given by Fig. 7 and the phase portrait relationship between the position and velocity is shown in Fig. 8. The trajectories are traveling through the searching space and the best optimal solution will be obtained eventually by a swarm intelligence logic in Algorithm 1.

Next, we present a theoretic result on global convergence of the iterative process (25)–(28). In particular, we view the proposed CSFMCO algorithm as a discrete-time switched linear system [24] or discrete linear inclusion [38] and then give some sufficient conditions for its global convergence by discussing its *semistability* property. To begin with, we define a series of matrices $A_k^{[j]}$ and $B_k^{[j]}$ (on the top of the next page), where $j = 1,\ldots,q$, $k = 0,1,2,\ldots$, $\eta_k\in\Omega_1\subseteq(0,2\eta]$, $\mu_k\in\Omega_2\subseteq(0,2\mu]$, $\kappa_k\in\Omega_3\subseteq(0,2\kappa]$, $\eta,\mu,\kappa>0$, $\otimes$ denotes the Kronecker product, $\mathbf{0}_{m\times n}$ denotes the $m\times n$ zero matrix, $\mathbf{1}_{m\times n}$ denotes the $m\times n$ matrix whose entries are all ones, $E_{n\times nq}^{[j]}\in\mathbb{R}^{n\times nq}$ denotes a block-matrix whose jth block-column is I_n and the rest block-elements are all zero matrices, $L\in\mathbb{R}^{q\times q}$ is the Laplacian matrix of certain graph topology underlying the CSFMCO algorithm, and $W^{[j]} = (\mathbf{1}_{q\times 1}\otimes I_n)E_{n\times nq}^{[j]}$.

Lemma 4.4. *Consider the matrices $A_k^{[j]}$ and $B_k^{[j]}$ given by (29) and (30), $j = 1,\ldots,q$, $k = 0,1,2,\ldots$. Then for every $j = 1,\ldots,q$ and every $k = 0,1,2,\ldots$, $\{0\}\subseteq\mathrm{spec}(A_k^{[j]})\subseteq\{0,-1,-\kappa_k,-(\mu_k+\kappa_k)\pm((\mu_k+\kappa_k)^2-(\mu_k+\kappa_k))^{1/2},-\frac{1}{2}(\mu_k+\kappa_k)\pm\frac{1}{2}((\mu_k+\kappa_k)^2-4(\mu_k+\kappa_k))^{1/2},\lambda\in\mathbb{C}:\forall(\lambda^2+(\mu_k+\kappa_k)\lambda+(\mu_k+\kappa_k))/(\eta_k\lambda)\in\mathrm{spec}(-L)\backslash\{0\}\}$ and $\{0\}\subseteq\mathrm{spec}(B_k^{[j]})\subseteq\{0,-1,-\frac{1}{2}(\mu_k+\kappa_k)\pm\frac{1}{2}((\mu_k+\kappa_k)^2-4(\mu_k+\kappa_k))^{1/2},\lambda,\zeta\in\mathbb{C}:\forall(\lambda^2+(\mu_k+\kappa_k)\lambda+(\mu_k+\kappa_k))/(\eta_k\lambda)\in\mathrm{spec}(-L)\backslash\{0\},\zeta^3+$*

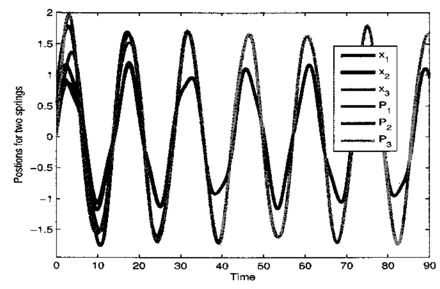


Figure 7: Three coupled springs' trajectories

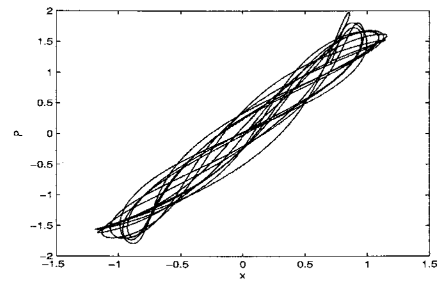


Figure 8: Phase portrait for the three coupled springs

$$A_k^{[j]} = \begin{bmatrix} -(\mu_k + \kappa_k)I_{nq} & I_{nq} - \eta_k L \otimes I_n & \kappa_k \mathbf{1}_{q\times 1} \otimes I_n \\ -(\mu_k + \kappa_k)I_{nq} & -\eta_k L \otimes I_n & \kappa_k \mathbf{1}_{q\times 1} \otimes I_n \\ \kappa_k E_{n\times nq}^{[j]} & \mathbf{0}_{n\times nq} & -\kappa_k I_n \end{bmatrix} \tag{29}$$

$$B_k^{[j]} = A_k^{[j]} + \begin{bmatrix} \mathbf{0}_{2nq\times nq} & \mathbf{0}_{2nq\times nq} & \mathbf{0}_{2nq\times n} \\ (1-\kappa_k)E_{n\times nq}^{[j]} & \mathbf{0}_{n\times nq} & -(1-\kappa_k)I_n \end{bmatrix} \tag{30}$$

$(1+\mu_k+\kappa_k)\zeta^2+(2\mu_k+\kappa_k)\zeta+\mu_k+\kappa_k=0\}$, where spec$(A)$ *denotes the spectrum of A. Furthermore, 0 is semisimple for both $A_k^{[j]}$ and $B_k^{[j]}$.*

Recall from [27] that a matrix $A \in \mathbb{R}^{n\times n}$ is *discrete-time semistable* if and only if $\mathrm{spec}(A)=\{1\}\cup\{\lambda\in\mathbb{C}:|\lambda|<1\}$ and if $1\in\mathrm{spec}(A)$, then 1 is semisimple. Let $\ker(A)$ denote the null space of matrix A. Now we have the main result for the global convergence of the iterative process (25)–(28).

Theorem 4.4. *Consider the following discrete-time switched linear model to describe the iterative process for CSFMCO:*

$$x_{k+1}^i = x_k^i + v_{k+1}^i, \quad x_0^i = x_{i0}, \tag{31}$$

$$v_{k+1}^i = v_k^i + \eta_k \sum_{j\in\mathcal{N}_i}(v_k^j - v_k^i) - \mu_k x_k^i + \kappa_k(p_k - x_k^i), \quad v_0^i = v_{i0}, \tag{32}$$

$$p_{k+1} = p_k + \kappa_k(x_k^j - p_k), \quad p_k \notin \mathcal{Z}_p, \tag{33}$$

$$p_{k+1} = x_k^k, \quad p_k \in \mathcal{Z}_p, \quad k=0,1,2,\ldots, \quad i=1,\ldots,q, \tag{34}$$

where $x_k^i\in\mathbb{R}^n$, $v_k^i\in\mathbb{R}^n$, $p_k\in\mathbb{R}^n$, $\eta_k\in\Omega_1\subseteq(0,2\eta]$, $\mu_k\in\Omega_2\subseteq(0,2\mu]$, $\kappa_k\in\Omega_3\subseteq(0,2\kappa]$, $\eta,\mu,\kappa>0$, $\mathcal{Z}_p=\{p\in\mathbb{R}^n: f(x_k^j)<f(p)\}$, and $x_k^j=\arg\min_{1\le i\le q} f(x_k^i)$. Let $h_1^\dagger=\min\Big\{-\frac{\lambda+\lambda^*}{|\lambda|^2}:\lambda\in\{-1,-\kappa_k,-(\mu_k+\kappa_k)\pm((\mu_k+\kappa_k)^2-(\mu_k+\kappa_k))^{1/2},-\frac{1}{2}(\mu_k+\kappa_k)\pm\frac{1}{2}((\mu_k+\kappa_k)^2-4(\mu_k+\kappa_k))^{1/2},\lambda\in\mathbb{C}:\forall(\lambda^2+(\mu_k+\kappa_k)\lambda+(\mu_k+\kappa_k))/(\eta_k\lambda)\in\mathrm{spec}(-L)\backslash\{0\}\},\forall k=0,1,2,\ldots\Big\}$ and $h_2^\dagger=\min\Big\{-\frac{\lambda+\lambda^*}{|\lambda|^2}:\lambda\in\{-1,-\frac{1}{2}(\mu_k+\kappa_k)\pm\frac{1}{2}((\mu_k+\kappa_k)^2-4(\mu_k+\kappa_k))^{1/2},\lambda,\zeta\in\mathbb{C}:\forall(\lambda^2+(\mu_k+\kappa_k)\lambda+(\mu_k+\kappa_k))/(\eta_k\lambda)\in\mathrm{spec}(-L)\backslash\{0\},\zeta^3+(1+\mu_k+\kappa_k)\zeta^2+(2\mu_k+\kappa_k)\zeta+\mu_k+\kappa_k=0\},\forall k=0,1,2,\ldots\Big\}$. If Ω_i is a finite discrete set for every $i=1,2,3$, $\min\{h_1^\dagger,h_2^\dagger\}>1$, $\ker((A_k^{[j]})^{\mathrm{T}}A_k^{[j]}+(A_k^{[j]})^{\mathrm{T}}+A_k^{[j]})=\ker((A_k^{[j]})^{\mathrm{T}}A_k^{[j]}+(A_k^{[j]})^2)$, $\ker((B_k^{[j]})^{\mathrm{T}}B_k^{[j]}+(B_k^{[j]})^{\mathrm{T}}+B_k^{[j]})=\ker((B_k^{[j]})^{\mathrm{T}}B_k^{[j]}+(B_k^{[j]})^2)$, $\|I_{2nq+n}+A_k^{[j]}\|\le 1$, and $\|I_{2nq+n}+B_k^{[j]}\|\le 1$ for every $k=0,1,2,\ldots$, then $x_k^i\to p^\dagger$, $v_k^i\to\mathbf{0}_{n\times1}$, and $p_k\to p^\dagger$ as $k\to\infty$ for every $x_{i0}\in\mathbb{R}^n$, $v_{i0}\in\mathbb{R}^n$, $p_0\in\mathbb{R}^n$, and every $i=1,\ldots,q$, where $p^\dagger\in\mathbb{R}^n$.

While Theorem 4.4 states that $\lim_{k\to\infty}x_k^i$ exists, it does not necessarily mean that $\min_{1\le i\le q}f(\lim_{k\to\infty}x_k^i)=\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$. Hence, the convergence issue of the iterative process given by (31)–(34) to a global optimum of the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n}f(\mathbf{x})$ remains an open problem. On the other hand, if we know some derivative information about the cost function f when using (31)–(34), then we may have some clue on this optimality issue. To see this, recall from Definition 8.6 of [39] that $f:\mathbb{R}^n\to\mathbb{R}$ is said to be *strictly convex* if for all $x,y\in\mathbb{R}^n$ with $x\neq y$ and any $t\in(0,1)$, $f(tx+(1-t)y)<tf(x)+(1-t)f(y)$. We assume that f is strictly convex and differentiable. Furthermore, we assume that all the conditions in Theorem 4.4 hold. In this case, it follows from Corollary 25.15 of [40] that the solution to the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n}f(\mathbf{x})$ is unique. Let $\mathbf{x}_*$ denote the solution to the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n}f(\mathbf{x})$ and $p^\dagger=\lim_{k\to\infty}x_k^i$ for every $i=1,\ldots,q$. Moreover, it follows from Theorem 25.5 of [41] that ∇f is continuous on $\mathbb{R}^n$, where ∇ denotes the nabla operator. Now it follows from Theorem 25.1 of [41] that $f(\mathbf{x}_*)\ge f(x_k^i)+\nabla^{\mathrm{T}}f(x_k^i)(\mathbf{x}_*-x_k^i)$ for every $i=1,\ldots,q$. If $\nabla f(p^\dagger)=\mathbf{0}_{n\times1}$, then it follows that $\lim_{k\to\infty}\nabla f(x_k^i)=\nabla f(\lim_{k\to\infty}x_k^i)=\nabla f(p^\dagger)=\mathbf{0}_{n\times1}$ for every $i=1,\ldots,q$. Thus, $f(\mathbf{x}_*)\ge\lim_{k\to\infty}f(x_k^i)+\lim_{k\to\infty}\nabla^{\mathrm{T}}f(x_k^i)(\mathbf{x}_*-x_k^i)=f(p^\dagger)$ for every $i=1,\ldots,q$. Alternatively, $f(p^\dagger)\ge f(\mathbf{x}_*)$, and hence, $f(p^\dagger)=f(\mathbf{x}_*)$. By uniqueness of $\mathbf{x}_*$, we have $p^\dagger=\mathbf{x}_*$, which implies that $\lim_{k\to\infty}x_k^i$ is indeed the solution to the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n}f(\mathbf{x})$ for every $i=1,\ldots,q$.

This analysis suggests that including the derivative information in our proposed CSFMCO algorithm may attain the global minimum of the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n}f(\mathbf{x})$, and thus, improve the convergence of the proposed algorithm. This is not surprising since almost all the global convergence optimizers in the literature are the gradient-based algorithms. These gradient-based algorithms utilize the derivative term with dynamic steps in the iterative process to guarantee the global convergence of the techniques. Nevertheless, what intrigues us here for CSFMCO is that we do not need to directly include the derivative term (if it exists) in the update formulas (25)–(28) to improve the convergence. Instead we can embed this derivative information in the swarm intelligence logic set $\mathcal{Z}_p$. More specifically, if f is merely convex, not necessarily differentiable, then we modify $\mathcal{Z}_p$ as

$$\mathcal{Z}_p=\{p\in\mathbb{R}^n: f(x_k^j)<f(p),\mathbf{0}_{n\times1}\notin\partial f(p)\},$$

where $\partial f(p)$ denotes the *subdifferential* of f at p [41, p. 215]. In this case, the judging condition $f(x_i)<f(p_i)$ in Algorithm 1 should include $\mathbf{0}_{n\times1}\notin\partial f(p_i)$.

In order to show the performance of the CSFMCO algorithm, we conduct a comparison evaluation between CSFMCO and other variations of swarm optimization. In particular, we use the following test functions chosen from [42] to evaluate the proposed CSFMCO algorithm. Reference [42] is a global optimization competition conducted at the 2013 IEEE Congress on Evolutionary Computation in Cancun, Mexico, and gives the latest benchmark functions to test the effectiveness of evolutionary algorithms. These new functions have been widely accepted as a standard in the area of computational intelligence.

- Sphere function: $f(x) = \sum_{i=1}^{n} x_i^2$. The test area is usually restricted to the hypercube $-5.12 \leq x_i \leq 5.12$ or $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$.
- Rastrigin function:
  $f(x) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]$. The test area is usually restricted to $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$.
- Rosenbrock's valley:
  $f(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$. Test area is usually restricted to hypercube $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$.
- Griewank function: $f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$. The test area is usually restricted to the hypercube $-600 \leq x_i \leq 600$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.
- Ackley function: $f(x) = -20\exp(-0.2 \times \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$. The test area is usually restricted to the hypercube $-32.768 \leq x_i \leq 32.768$, $i = 1, \ldots, n$. The global minimum of $f(x)$ is 0 at $x_i = 0$.

Before we introduce the new test functions, some subfunctions are needed.

- Λ^{α}: An n-dimensional diagonal matrix with the diagonal elements $\lambda_{i,i} = \alpha^{\frac{i-1}{2(n-1)}}$. This matrix is used to create ill-conditioning [43]. The parameter α is the condition number.
- X^{opt} is the optimum decision vector for which the value of the objective function is minimum. This is also used as a shift vector to change the location of the global optimum.
- T_{osz} is a transformation function to create smooth local irregularities [43].
- T_{asy}^{β} is a transformation function to break the symmetry of the symmetric functions [43].

Based on these nonlinear transformations and operations, we introduce two more complicated functions induced from the above basic functions. ★ Shifted Rastrigin function: $f(x) = \sum_{i=1}^{n}[z_i^2 - 10\cos(2\pi z_i) + 10]$.

- $Z = \Lambda^{10} T_{asy}^{0.2}(T_{osz}(X - X^{opt}))$.
- $X(i) \in (-5, 5)$, where $X(i)$ denotes the ith component of X.
- Global optimum: $f(X^{opt}) = 0$.

★ Shifted Ackley function: $f(x) = -20\exp(-0.2 \times \sqrt{\frac{1}{n}\sum_{i=1}^{n} Z_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi Z_i) + 20 + e$.

- $Z = \Lambda^{10} T_{asy}^{0.2}(T_{osz}(X - X^{opt}))$.
- $X(i) \in (-5, 5)$.
- Global optimum: $f(X^{opt}) = 0$.

In this work, X^{opt} for both shifted Rastrigin function and Ackley function is $\texttt{rand}^n \in \mathbb{R}^n$, where the MATLAB command rand generates a random real number between [01] and $\texttt{rand}^n$ denotes a vector whose every component is generated by rand.

For $n = 30$, $\texttt{rand}^n$ generates X^{opt} as [0.5759 0.1490 0.5844 0.6050 0.0224 0.9444 0.6806 0.8094 0.4486 0.5921 0.0517 0.8177 0.8144 0.4230 0.9635 0.5372 0.7677 0.9022 0.8058 0.4668 0.2006 0.5209 0.4287 0.0398 0.5966 0.5687 0.5644 0.1430 0.0666 0.5222].

In this part, statistical results of the best values obtained from CSFMCO and other variant swarm algorithms are compared numerically. PSO we used here is the standard swarm algorithm and the center PSO (CPSO) algorithm used here is proposed in [44] by introducing a center among all the particles. Moreover, comprehensive learning PSO

Table 2: Numerical Comparison Between BPSO, BMCO, NBPSO, BDE and BCSFMCO

Function	Min						Max					
	PSO	CPSO	CLPSO	FullPSO	MCO	CSFMCO	PSO	CPSO	CLPSO	FullPSO	MCO	CSFMCO
Sphere	1.9464E1	9.1612E-05	3.8987	9.4531E-54	1.900E-3	5.0061E-82	1.0949E2	2.01E-2	1.0933E1	7.0367E-8	7.4E-3	2.5213E-78
Rastrigin	1.5075E2	1.1244E2	9.8179E1	2.855E1	8.166E1	0	5.1968E2	2.8456E2	1.4607E2	2.2226E2	3.0523E2	0
Rosenbrock	2.4632E3	4.554E1	2.1372E2	1.5431E2	2.544E1	2.7767E1	1.9071E4	4.1783E2	9.1991E2	4.9414E2	1.1552E2	2.890E1
Griewanks	3.47E-2	7.5102E-8	9.52E-1	9.905E-1	8.652E-1	0	8.557E-1	2.2386E-4	9.958E-1	9.988E-1	9.974E-1	0
Ackley	5.2166	2.8872	3.3316	1.1102E-13	2.0196	8.8818E-16	1.0592E1	6.3280	4.1993	2.5428	3.9990	8.8818E-16
Shifted Rastrigin	2.7358E2	1.8965E2	2.0146E2	1.5281E2	2.6060E2	1.6713E2	4.7798E2	4.5287E2	2.8420E2	3.0784E2	5.2487E2	2.5691E2
Shifted Ackley	7.8215	6.8052	5.8670	4.1044	5.9411	5.2082	1.0106E1	1,0055E1	7.2915	6.7953	9.7824	5.9521
Function	Median						Average					
	PSO	CPSO	CLPSO	FullPSO	MCO	CSFMCO	PSO	CPSO	CLPSO	FullPSO	MCO	CSFMCO
Sphere	5.427E1	2.100E-3	6.9221	8.5716E-27	4.300E-3	4.4568E-80	5.52369E2	3.8E-3	6.9337	4.2619E-09	4.4E-3	2.1386E-79
Rastrigin	3.1258E2	1.7163E2	1.2930E2	7.736E1	1.9313E2	0	3.1363E2	2.8456E2	1.2742E2	8.656E1	1.9792E2	0
Rosenbrock	5.6407E3	1.2779E2	4.5091E2	2.5143E2	6.446E1	2.8443E1	6.1813E3	1.6701E2	4.8552E2	2.7036E2	6.671E1	2.846E1
Griewanks	1.858E-1	2.0368E-6	9.921E-1	9.976E-1	9.857E-1	0	2.400E-1	2.4927E-5	9.891E-1	9.971E-1	9.706E-1	0
Ackley	7.3016	4.7751	3.7753	1.4131	2.7454	8.8818E-16	7.3744	4.6635	3.7971	1.3252	2.8617	8.8818E-16
Shifted Rastrigin	3.7832E2	3.0205E2	2.3874E2	2.3742E2	3.7983E2	2.1977E2	3.7134E2	3.1169E2	2.3818E2	2.3486E2	3.8535E2	2.1363E2
Shifted Ackley	9.3525	7.8225	6.4417	8.2779	5.4123	5.6085	9.1389	7.9699	6.5187	5.4849	8.1692	5.6036

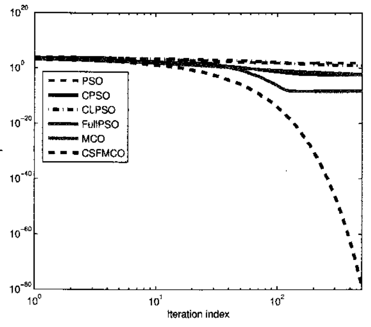


Figure 9: Test function: Sphere

(CLPSO) [45], fully informed PSO (FullPSO) [46], and MCO are also compared with CSFMCO. The benchmark functions are listed before with $n = 30$. After running 20 times, the maximum, minimum, average, and median of best values obtained by all the algorithms are compared in Table 2. Moreover, the averaging trajectories of all the listed algorithms approaching the optimal solution are shown in Fig. 9–Fig. 15. Based on those simulation results, one can conclude that the performance of the CSFMCO algorithm is superb over all the other cited variations.

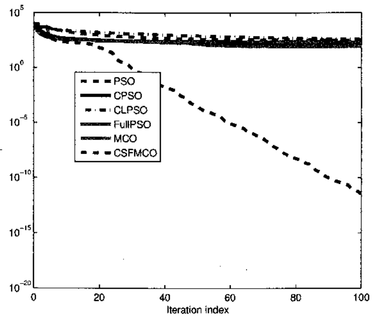


Figure 10: Test function: Rastrigin

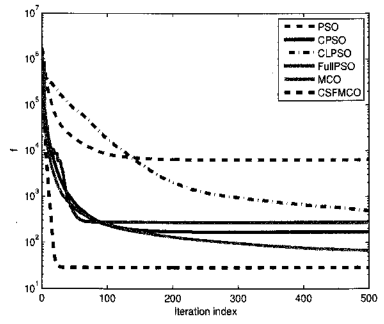


Figure 11: Test function: Rosenbrock

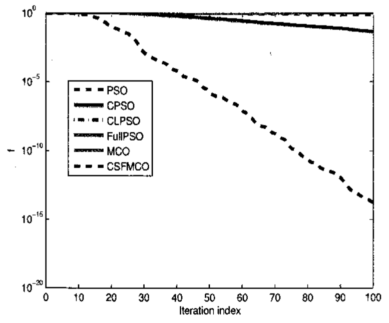


Figure 12: Test function: Griewanks

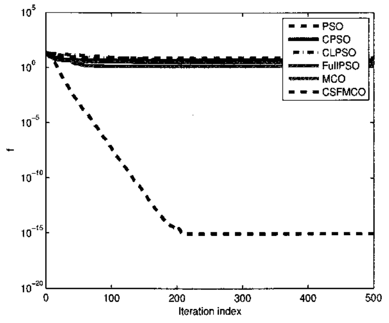


Figure 13: Test function: Ackley

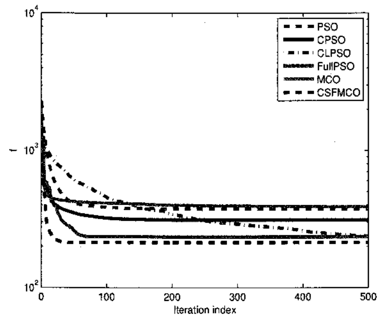


Figure 14: Test function: Shifted Rastrigin

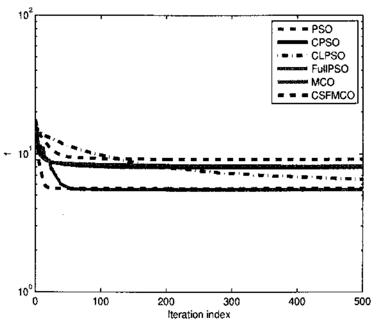


Figure 15: Test function: Shifted Ackley

4.3 New Multiagent Coordination Optimization Algorithms for Mixed-Binary Nonlinear Programming

Mixed-integer nonlinear programming has attracted a lot of research efforts recently, which are heavily applied to solve the problems in transportation and manufacturing [47], airline crew scheduling [48], vehicle routing [49], power system planning [50], to cite but a few examples. Specifically, a mixed-integer programming problem is a minimization or maximization of an objective function subject to nonlinear constraints including both continuous variables and discrete variables. In this research, we focus on a particular subset of mixed-integer nonlinear programming problems by restricting the discrete variables to be binary, and so called "mixed-binary nonlinear programming problem". Binary variables consist of only 0 and 1, which have significant meanings in engineering and science disciplines. One of the main representations of binary variables is *on* or *off* for a communication link.

To solve the MBNP problem efficiently, the challenge lies in the *convexity*, although there are a subset of convex MBNP problems, and some approaches have been proposed, see [51, 52] for instance, the research efforts are highly needed to solve a larger (even nonconvex) class of MBNP problems. Due to the difficulty of solving MBNP problems in different situations by use of the conventional methods, we resort to heuristic optimization algorithms which can serve as an efficient approach to find the best solutions. Swarm intelligence has attracted a lot of attention recently and swarm based optimization algorithms, for instance, particle swarm optimization (PSO) [53], are one of these efficient solvers.

Multiagent coordination optimization (MCO) [14] is a new algorithm inspired by PSO and consensus protocols for multiagent coordination. Different from the standard PSO, the particles in the MCO algorithm embed the multiagent coordination terms in the update formula, and share the velocity and position information with neighbors through a communication topology. Inspired by the synchronization and diversity of the coupled springs motions, a new coupled spring forced MCO (CSFMCO) algorithm has been proposed [54]. Specifically, each particle in the MCO algorithm is treated as a spring and the best solution found so far as the other spring. Governed by coupled springs motions, the next positions and velocities for both two springs will be discretely updated, then the swarm intelligence logic will be switched on to update the local optimal solution found by the particle itself and also the best optimal solution found among all the particles. Numerical evaluation has been conducted compared with other variations of PSO, indicating that the proposed CSFMCO surpasses other algorithms when solving some benchmark functions.

With the aim to solve MBNP problems, firstly, motivated by the idea of binary PSO [55], a new binary CSFMCO algorithm is proposed. Numerical evaluation has been conducted and compared with other variations of binary PSO, showing the superiority of binary CSFMCO. As an application of binary CSFMCO algorithm, the multiagent formation control problem is investigated. Formation control for multiagent systems has gained considerable attention in control theory and topological heterogeneity analysis has been conducted for both single and double integrators systems for undirected topologies [56]. However, directed topology is a more general scenario than the undirected case, and thus provides more alternatives when designing formation control protocols. For example, the number of undirected topologies for the 5 robot system is 2^{10} while the number of directed topologies is 2^{20}. Therefore, in this work, as a binary optimization application, we investigate the topological heterogeneity of digraphs for multiagent formation.

Next, together with the continuous CSFMCO, a mix-CSFMCO algorithm for MBNP problems is proposed by introducing a switching logic between the binary optimizer and the continuous optimizer. As an application to this algorithm, we consider balanced coordination for damage mitigation and resource allocation in network systems. The balanced coordination problem for network systems with a fixed-structure, which can be characterized as a high-order nonlinear matrix equation involved, nonconvex optimization problem, has been investigated [57]. However, network structure plays a significant role when communicating between different nodes to allocate the resource, and thus affects the convergence rate for the allocation algorithms. The independent study of network structure and network parameters cannot give a comprehensive understanding of the resource allocation algorithms. Therefore, to better investigate the network structure and optimal balanced coordination algorithms in the network, in this work, we formulate the design problem as an MBNP problem and numerically solve it by the proposed mix-CSFMCO algorithm.

The binary PSO (BPSO) proposed by Kennedy and Eberhart [55] can be described as the following mathematical iterative algorithm to minimize the objective function $f(\mathbf{x})$:

$$\begin{aligned} \mathbf{v}_{k+1}^{i} &= \mathbf{v}_{k}^{i} + b_1 r_1(\mathbf{g}_{loc,i} - \mathbf{x}_{k}^{i}) \\ &\quad + b_2 r_2(\mathbf{g} - \mathbf{x}_{k}^{i}), \quad k = 0, 1, 2, \ldots, \end{aligned} \tag{35}$$

$$\mathbf{x}_{k+1}^{i,j} = \begin{cases} 1 & \text{if } r_{i,j} < \mathtt{sig}(\mathbf{v}_{k+1}^{i,j}) \\ 0 & \text{otherwise} \end{cases}, \tag{36}$$

where $\mathbf{v}_k^i$ and $\mathbf{x}_k^i$ are the velocity and position of particle i at iteration k, respectively, $\mathbf{g}_{loc,i}$ is the position of the previous best value that particle i has obtained so far, $\mathbf{g}$ is the position of the global best value that the swarm of the particles can achieve so far, b_1 and b_2 are weight coefficients, r_1 and r_2 are two random coefficients which are usually selected in uniform distribution in the range $[0, 1]$, $r_{i,j}$ are random numbers in [0,1], and `sig` function is defined by $\mathtt{sig}(x) = \frac{1}{1+e^{-x}}$.

In this work, undirected and directed graphs are used to represent a topology of communication networks. Specifically, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a *directed graph* (or digraph) denoting the communication network with the set of *nodes* (or vertices) $\mathcal{V} = \{1, \ldots, q\}$ involving a finite nonempty set denoting the agents, and the set of *edges* $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ involving a set of ordered pairs (i, j) denoting the direction of communication. A *graph* or *undirected graph* $\mathcal{G}$ is a directed graph for which the *edge set* is symmetric. For graphs, we use unordered pairs $\{i, j\}$ for edges. The set of neighbors of node i is thus defined by $\mathcal{N}_i = \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}$.

The *connectivity matrix* $L = L^{\mathcal{G}}$ associated with the graph $\mathcal{G}$ is defined by

$$L_{i,j}^{\mathcal{G}} \triangleq \begin{cases} 0, & \text{if } (i,j) \notin \mathcal{E}, \\ 1, & \text{otherwise,} \end{cases} \quad i \neq j, \quad i, j = 1, \ldots, q, \tag{37}$$

$$L_{i,i}^{\mathcal{G}} \triangleq -\sum_{k=1, k\neq i}^{q} L_{i,k}, \quad i = 1, \ldots, q, \tag{38}$$

where q is the number of agents.

Binary multiagent coordination optimization (BMCO) is proposed by introducing a communication topology for the particles in binary PSO algorithm and using recently developed multiagent consensus protocols from control theory. The velocity update formula for the BMCO algorithm is shown in (39), and the position update formula is the same as the binary PSO algorithm in (36).

$$\begin{aligned} \mathbf{v}_{k+1}^{i} &= \mathbf{v}_{k}^{i} + \check{\eta} \sum_{j \in \mathcal{N}_i} (\mathbf{v}_{k}^{j} - \mathbf{v}_{k}^{i}) \\ &\quad + \check{\mu} \sum_{j \in \mathcal{N}_i} (\mathbf{x}_{k}^{j} - \mathbf{x}_{k}^{i}) + \check{\kappa}(p - \mathbf{x}_{k}^{i}) \end{aligned} \tag{39}$$

where $\check{\eta} \sim U(0, 2\eta)$, $\check{\mu} \sim U(0, 2\mu)$, $\check{\kappa} \sim U(0, 2\kappa)$, $U(\cdot, \cdot)$ is the uniform distribution, and $\eta, \mu, \kappa > 0$ are random numbers.

In this part, the coupled spring forced multiagent coordination optimization algorithm is proposed to address MBNP problems. First, to handle the binary variables, the binary CSFMCO (BCSFMCO) algorithm is proposed and described in Algorithm 2.

- Sphere function: $f(x) = \sum_{i=1}^{n} x_i^2$. The test area is usually restricted to the hypercube $-5.12 \leq x_i \leq 5.12$ or $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$.

- Rastrigin function:
$f(x) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]$. The test area is usually restricted to the hypercube $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$.

Algorithm 2 Binary Coupled spring forced Multiagent Coordination Optimization Algorithm

```
for each agent $i = 1, \ldots, q$ do
  Initialize the agent's position with a uniformly distributed random vector: $x_i \sim U(\underline{x}, \overline{x})$, where $\underline{x}$ and $\overline{x}$ are the lower and upper boundaries of the search space;
  Initialize the agent's velocity: $v_i \sim U(\underline{v}, \overline{v})$, where $\underline{v}$ and $\overline{v}$ are the lower and upper boundaries of the search speed;
  Update the agent's best known position to its initial position: $p_i \leftarrow x_i$;
  If $J(p_i) < J(p)$ update the multiagent network's best known position: $p \leftarrow p_i$.
end for
repeat
  $k \leftarrow k + 1$;
  for each agent $j = 1, \ldots, q$ do
    Choose random parameters: $\eta, \mu, \lambda, \kappa \sim U(0, \Theta)$, $\Theta > 0$;
    Update the agent's velocity: $v_i \leftarrow v_i + \check{\eta} \sum_{j \in N_i}(v_j - v_i) - \check{\mu} x_i + \check{\kappa}(p - x_i)$;
    Update the agent's position:
      $$x_{i,j} \leftarrow \begin{cases} 1 & \text{if } r_{i,j} < \mathrm{sig}(v_{i,j}) \\ 0 & \text{otherwise} \end{cases} \quad (40)$$
    where $r_{i,j} \sim U(0, 1)$;
    $x_i \leftarrow [x_{i,1}, \ldots, x_{i,q}]^{\mathrm{T}}$;
    for $J(x_i) < J(p_i)$ do
      Update the agent's best known position: $p_i \leftarrow x_i$;
      Update the multiagent network's best known position: $p \leftarrow \mathrm{round}(p + \check{\kappa}(p_i - p))$;
      If $f(p_i) < f(p)$ update the multiagent network's best known position: $p \leftarrow p_i$;
    end for
  end for
until $k$ is large enough or the change of $f$ becomes small
return $p$
```

- Rosenbrock's valley:
  $f(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$. Test area is usually restricted to the hypercube $-30 \leq x_i \leq 30$, $i = 1, \ldots, n$.

- De Jong's f4:
  $f(x) = \sum_{i=1}^{n}(ix_i^4)$. Test area is usually restricted to the hypercube $-20 \leq x_i \leq 20$, $i = 1, \ldots, n$.

- Zakharov function:
  $f(x) = \sum_{i=1}^{20} x_i^2 + (0.5ix_i)^2 + (0.5ix_i)^4$. The test area is usually restricted to the hypercube $-10 \leq x_i \leq 10$, $i = 1, \ldots, 20$.

In this part, statistic results of the BPSO(A1), BMCO(A2), another variation NBPSO(A3) in [58], binary differential evolution (BDE)(A4) [59] and binary CSFMCO(A5) algorithms are given for solving the benchmark functions. Together with the above test functions, two shifted functions are also used in this work. Different with the standard test function, the shifted Sphere and shifted Ackly functions translate the **0** optimal solution into a particular one p^*. The problem dimension is $n = 100$. By running 20 times for every binary optimization algorithm, the maximum, minimum, average, and median objective values are compared between the standard BPSO, NBPSO, BDE and BMCO algorithms, which are shown in Table 3. From Table 3, one can conclude that the proposed binary CSFMCO algorithm is better than the BPSO, BMCO and NBPSO algorithms.

Moreover, the Wilcoxon sum rank test is provided for all the binary algorithms and the p values are shown in Table 4. Based on the results, it is concluded that the median values of the BCSFMCO algorithm are smaller than the rest binary algorithms with the significance level 0.05.

Table 3: Numerical Comparison Between BPSO, BMCO, NBPSO, BDE and BCSFMCO

Function	Min					Max				
	BPSO	BMCO	NBPSO	BDE	BCSFMCO	BPSO	BMCO	NBPSO	BDE	BCSFMCO
Sphere	0	0	0	0	1,15E2	1.27E2	0	1.22E2	0	1.30E2
Shifted Sphere	3.1E1	3.1E1	2.8E1	2.8E1	3.0E1	4.0E1	4.0E1	4.0E1	3.9E1	3.8E1
Rastrigin	0	0	0	0	0	4.825E3	4.02E2	8	0	1.0384E4
Rosenbrock	0	0	0	0	3	9	0	1.1E3	0	9
Ackly	8.8818E-16	8.8818E-16	8.8818E-16	8.8818E-16	2.3764	2.5111	8.8818E-16	2.4583	8.8818E-16	2.5024
Shifted Ackly	2.1076	1.9391	2.0421	1.9391	2.0752	2.4313	2.3198	2.3483	2.2318	2.3198
De Jone F4	0	0	0	0	1.21E2	1.129E3	0	1.36E2	0	1.33E2
Zakharov	0	0	0	0	1.0008E10	1.2341E10	0	1.1131E10	0	1.2303E10
Function	Median					Average				
	BPSO	BMCO	NBPSO	BDE	BCSFMCO	BPSO	BMCO	NBPSO	BDE	BCSFMCO
Sphere	9.95E1	0	1.45E1	0	1.235E2	7.22E1	0	3.275E1	0	1.239E2
Shifted Sphere	3.7E1	3.6E1	3.45E1	3.3E1	3.5E1	3.66E1	3.605E1	3.465E1	3.34E	3.455E1
Rastrigin	0	0	1.9105E3	0	0	6.426E2	1.9435E2	4.0208E3	0	1.6058E3
Rosenbrock	5	0	0	0	7	4.05	0	2	0	6.55
Ackly	2.3472	8.8818E-16	1.1314	8.8818E-16	2.4358	1.3404	8.8818e-16	1.1101	8.8818E-16	2.4397
Shifted Ackly	2.2763	2.2318	2.2318	2.1861	2.2015	2.2703	2.2176	2.2222	2.1451	2.2067
De Jone F4	0	0	1.550E1	0	1.25E2	3.525E1	0	1.30E2	0	1.255E2
Zakharov	0	0	5.9159E8	0	1.1054E10	4.6462E9	0	2.3410E9	0	1.1151E10

Table 4: Wilcoxon sum rank test for BPSO, BMCO, NBPSO, BDE and BCSFMCO

Function	Sphere	Shifted Sphere	Rastrigin	Rosenbrock	Ackly	Shifted Ackly	De Jone F4	Zakharov
A5>A1	1.2765E-5	2.0483E-4	4.4846E-4	8.2073E-5	8.3018E-5	6.3144E-5	2.3E-3	1.0E-3
A5>A2	1	9.9172E-4	7.9202E-6	1	1	3.7E-3	1	1
A5>A3	5.5082E-7	5.68E-2	1.7473E-07	1.2289E-5	1.6492E-6	5.9E-3	5.2163E-8	1.7527E-7
A5>A4	3.8234E-9	4.04E-2	1.0E-3	3.4078E-9	3.8517E-9	3.16E-2	3.6847E-9	4.0033E-9

In this part, the averaged convergence rates of BPSO, BMCO, NBPSO, BDE and BCSFMCO are compared by solving the relevant optimization problems for the eight test functions 20 times and taking the average of the objective values in each iteration, which are shown in Fig. 16–23. Through all the figures, the CSFMCO algorithm achieves the best solutions among all the other algorithms in a faster convergence rate.

Based on the proposed binary CSFMCO algorithm and continuous CSFMCO algorithm, the overall mix-CSFMCO algorithm for MBNP problems is presented. The flow diagram of the algorithm addressing MBNP problems is shown in Fig. 24. Therefore, a heuristic optimization algorithm will be equipped as a numerical approach to address MBNP problems. The promising success of this approach lies in the fact that the mix-CSFMCO algorithm does not require the convexity of the objective function and the algorithm can be easily implemented to rapidly find the best solution for the problem.

In this part, the formation control problem for multiagent systems underlying heterogeneous directed topologies is considered, which extends the results from the undirected case to a general digraph scenario in [60].

The system we consider here is a double-integrator multiagent system given by $\dot{x}_i = v_i \quad \dot{v}_i = u_i$ where x_i and v_i, $i = 1, 2, \cdots, M$ are the position and velocity of agent i respectively. To achieve the desired formation, we propose the

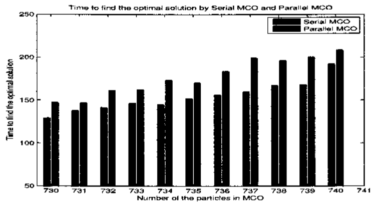


Figure 16: Test function: Sphere

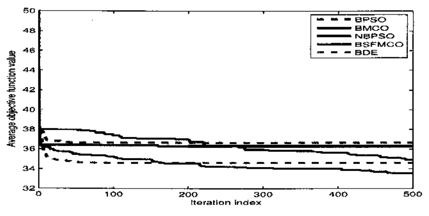


Figure 17: Test function: shifted Sphere

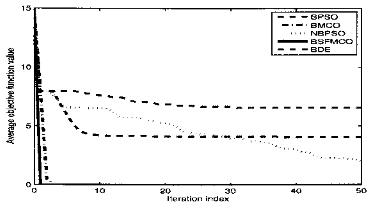


Figure 18: Test function: Rosenbrock

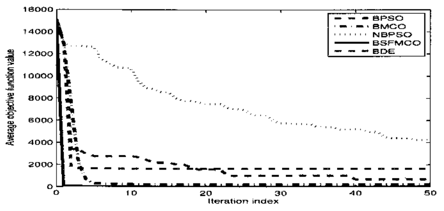


Figure 19: Test function: Rastrigin

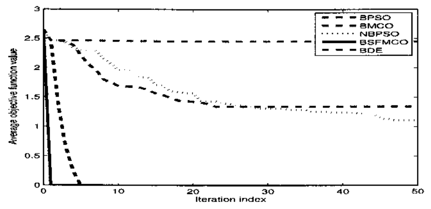


Figure 20: Test function: Ackley

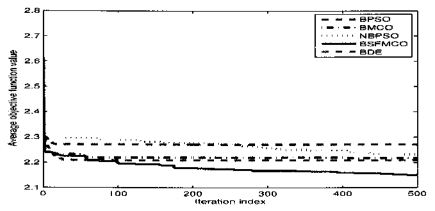


Figure 21: Test function: shifted Ackley

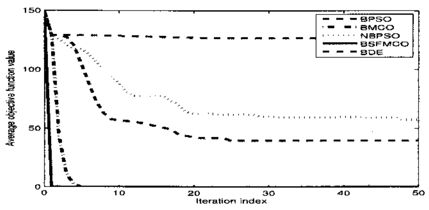


Figure 22: Test function: De Jone F4

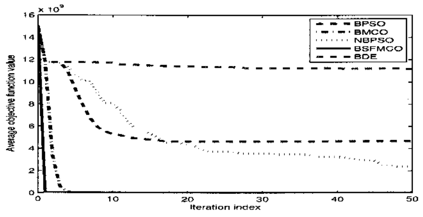


Figure 23: Test function: Zakharov

following directed topological heterogeneous formation control protocol

$$\begin{aligned} u_i &= -\sum_{j=1,j\neq i}^{M} L^{\mathcal{G}_1}(x_i(t)-x_j(t)-l_{i,j}) \\ &\quad -\sum_{j=1,j\neq i}^{M} L^{\mathcal{G}_2}(v_i(t)-v_j(t)) \end{aligned} \tag{41}$$

Or the vector form is

$$\dot{X} = \Phi X + \Psi, \quad X(0) = X_0 \tag{42}$$

where $X = [x_1, \cdots x_M, v_1, \cdots, v_M]^{\mathrm{T}}$, $\Phi = \begin{bmatrix} \mathbf{0} & I \\ -L^{\mathcal{G}_1} & -L^{\mathcal{G}_2} \end{bmatrix}$ and $\Psi = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \otimes L^{\mathcal{G}_1} \times \begin{bmatrix} 0 & \cdots & 0 & 0 & l_{12} & \cdots & l_{1q} \end{bmatrix}^{\mathrm{T}}$, where $\otimes$ denotes the Kronecker product, $L^{\mathcal{G}_1}$ and $L^{\mathcal{G}_2}$ are the connectivity matrices for the position and velocity communication topologies, respectively, $L^{\mathcal{G}_1}$ and $L^{\mathcal{G}_2}$ are directed but not necessarily the *same*, or even do not have a spanning tree topology, and $l_{i,j}$ is the desired distance between agent i and agent j. At this moment, one dimensional distance control is considered.

Theorem 4.5. *Suppose that the topology for position has a directed spanning tree, the multiagent system achieves the desired formation under the formation control protocol (41) if and only if Φ has no eigenvalues lying on the imaginary axis, i.e., for each agent i*

$$d_{i,j}(t) \to l_{i,j}, \quad v_i(t) \to \frac{1}{M}\mathbf{1}^{\mathrm{T}}\mathbf{1}v(0), \quad t \to \infty \tag{43}$$

where $d_{i,j}(t) = x_i(t) - x_j(t)$ is the distance between agent i and agent j at time t, $\mathbf{1} = [1, \ldots, 1]^{\mathrm{T}}$, and $v(0) = [v_1(0), v_2(0), \cdots, v_M(0)]^{\mathrm{T}}$ is the initial velocity vector for the multiagent system.

Remark 4.3. *The main difference between the undirected and directed cases lies in the necessary and sufficient condition that matrix Φ has eigenvalues lying on the imaginary axis. In the following part of the work, we will show how to eliminate this constraint when we formate the optimal topology design problem.*

Based on Theorem 4.5 , the formation control protocol can be extended from the undirected case to the directed case. Since the convergence rate and communication cost are two main concerns when designing and evaluating a formation control protocol. Hence in this work, we formulate the topology optimization problem for multiagent

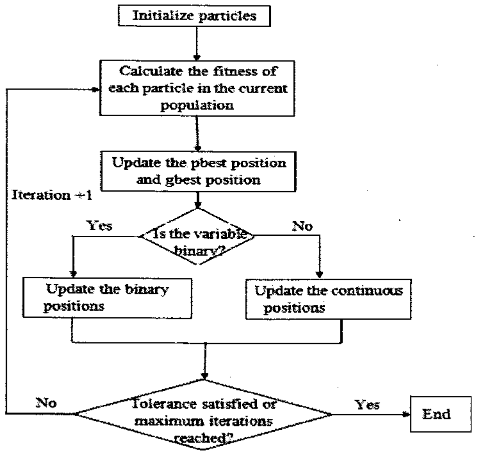

Initialize particles
Calculate the fitness of each particle in the current population
Update the pbest position and gbest position
Iteration +1
Is the variable binary?
Yes
No
Update the binary positions
Update the continuous positions
Tolerance satisfied or maximum iterations reached?
No
Yes
End

Figure 24: Flow diagram for the CSFMCO algorithm addressing MBNP problems

systems underlying directed heterogeneous topologies by considering the tradeoff between the convergence rate and communication cost.

To measure the communication and interagent sensing cost of the multiagent system under a graph $\mathcal{G}$, first the distance-based communication/sensing cost $C(\mathcal{G})$ is defined as $C(\mathcal{G}) = \sum_{i,j=1}^{q} d_{i,j} L_{i,j}^{\mathcal{G}}$ where $d_{i,j}$ is the geospatial distance between agents i and j. Next, the convergence time for the multiagent system is another factor that we need to consider. To this end, let $I(\mathcal{G})$ denote the iteration number of numerical algorithms for the multiagent system to reach the desired formation under $\mathcal{G}$ within allowed numerical error bounds. Now together with the communication/sensing cost, we introduce the following optimization problem given by

$$\begin{aligned} &\text{minimize} && \frac{w_1 C(L_2)}{C(L_t)} + \frac{w_2 I(L_2)}{I(L_t)} \\ &\text{subject to} && \lambda(\Phi) \notin \{j\omega | j^2 = -1, \omega \in \mathbb{R}\} \end{aligned} \tag{44}$$

where $L_2 = L^{\mathcal{G}_2}$, L_t denotes the connectivity matrix for the complete graph topology, $I(\mathcal{G}_c)$ is the iteration number when $\mathcal{G}_2$ is a complete graph topology, $\lambda(\Phi)$ denotes the eigenvalues of Φ, and w_1, w_2 are two positive weight constants.

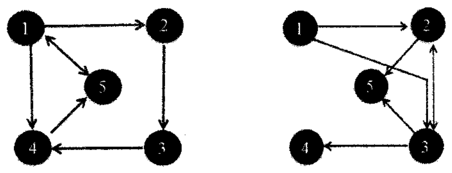
Theorem 4.6. *The constrained optimization problem (44) can be converted into an unconstrained optimization problem:*

$$\text{minimize} \quad \frac{w_1 C(L_2)}{C(L_t)} + \frac{w_2 I(L_2)}{I(L_t)} \tag{45}$$

In the following presentation, we provide some simulation results of binary CSFMCO addressing the optimal topology design problem induced from Theorem 4.6 for multiagent formation. To begin with, we first verify Theorem 4.5 by considering a group of 5 robots keeping distance "1" with each other. The position topology and velocity topology are shown in Fig. 25, specifically, the position topology has a spanning tree while the velocity topology does not. The simulation results are shown in Fig. 26, which is shown that the robots go into the desired formation underlying the heterogeneous topologies.

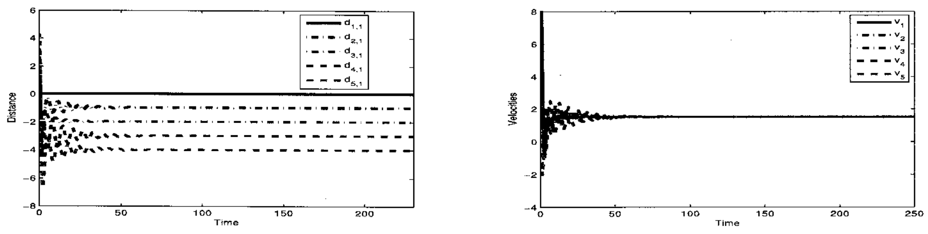
In the second part of the simulation, we provide a numerical approach to solve the optimal topology design problem in Theorem 4.6 by means of the binary CSFMCO algorithm. The distance matrix is given by

$$D(i,j) = \begin{bmatrix} 0 & 775 & 422 & 329 & 758 \\ 775 & 0 & 444 & 1046 & 890 \\ 422 & 444 & 0 & 218 & 313 \\ 329 & 1046 & 218 & 0 & 104 \\ 758 & 890 & 313 & 104 & 0 \end{bmatrix} \tag{46}$$



(a) Position topology (b) Velocity topology

Figure 25: Position and velocity topologies for multiagent formation



(a) Distance $d_{j,1}$ versus time (b) Velocities versus time

Figure 26: Position and velocity topologies for multiagent formation

By running the four different binary algorithms 30 times, the maximum, minimum, medina and average values are compared and shown in Table 5. It can be concluded that the CSFMCO algorithm achieves the best solution among all the other algorithms. The best topology found is shown in Fig. 27 and the trajectories of the distance between agents and velocity are shown in Fig. 28.

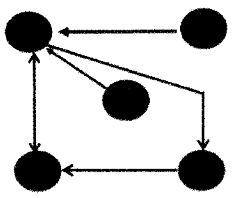
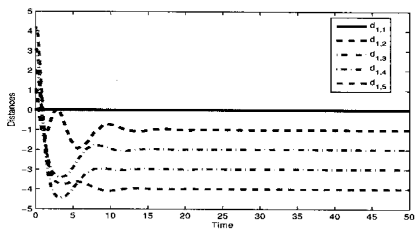


Figure 27: Best topology $L^{\mathcal{G}^*}$ found by binary CSFMCO

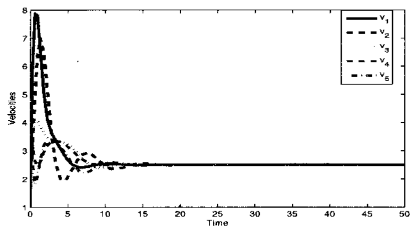
Balanced coordination for damage mitigation and resource allocation in network systems enhances rapid dissemination of network resources and self-healing of the network, which lead to significant reduction of the threats imposed by weapons of mass destruction (WMD) attacks. A fix-structured resource allocation problem has been investigated in [15]. However, network structure plays a significant role when communicating between different nodes to allocate the resource, and thus affects the convergence rate for the allocation algorithms. The independent study of network structure and network parameters cannot give a comprehensive understanding of the resource allocation algorithms. Therefore, to better investigate the relationship between network structure and optimal balanced coordination algorithms over the network, in this part we formulate the corresponding design problem as an MBNP problem and numerically solve it by using the mix-CSFMCO algorithm.

Table 5: Comparison Between BPSO, BMCO, NBPSO and BCSFMCO

	BPSO	BMCO	NBPSO	BCSFMCO
Min	3.0850E4	3.1639E4	3.0922E4	2.7893E4
Max	1.0697E7	1.0697E7	1.0697E7	1.0697E7
Average	4.5029E5	4.5091E5	4.5024E5	4.4860E5
Median	3.0850E4	3.1639E4	3.0922E4	2.8601E4



(a) Distance $d_{j,1}$ versus time by the optimal topology



(b) Velocities versus time

Figure 28: System trajectory underlying topology $L^{\mathcal{G}^*}$

The iterative algorithm design for coordinated resource allocation in network systems is given by

$$\begin{aligned} x_i(k+1) = a_{ii}x_i(k) - &\sum_{j=1,j\neq i}^{n} L_{(j,i)}a_{ji}[x_i(k) + d_iw_i(k)] \\ &+ \sum_{j=1,j\neq i}^{n} L_{(i,j)}a_{ij}[x_j(k) + d_jw_j(k)], \\ i = 1,\ldots,n, \quad & t = 0,1,2,\ldots, \quad x_i(0) \geq 0, \end{aligned} \tag{47}$$

or in the vector form,

$$\begin{aligned} x(k+1) &= Ex(k) + (E-\Lambda)Dw(k) \\ &= \Lambda x(k) + (E-\Lambda)x(k) + (E-\Lambda)Dw(k), \end{aligned}$$

where $x(k) = [x_1(k),\ldots,x_n(k)]^{\mathrm{T}}$,$D = \mathrm{diag}\,[d_1,\ldots,d_n]$, for $i,j = 1,\ldots,n$, $w(k) = [w_1(k),\ldots,w_n(k)]^{\mathrm{T}} \in \mathbb{R}^n$ denotes the standard white noise vector, $\Lambda = \mathrm{diag}\,[a_{11},\ldots,a_{nn}] \in \mathbb{R}^{n\times n}$, , $L_{(i,j)}$ represents the (i,j)th element of the Laplacian matrix L for a certain graph $\mathbb{G}$ (not necessarily the same $\mathcal{G}$ for MCO) defined by $L_{(i,j)} = 1$ if $i \in \mathcal{N}_j$, $L_{(i,j)} = 0$ if $i \notin \mathcal{N}_j$ and $i \neq j$, and $L_{(i,i)} = 1$,

$$E_{(i,j)} = \begin{cases} a_{ii} - \sum_{l=1,l\neq i}^{n} L_{(l,i)}a_{li}, & i = j, \\ L_{(i,j)}a_{ij}, & i \neq j, \end{cases} \tag{48}$$

a_{ij} is the parameter that we need to design, $a_{ij} \geq 0$ and $\sum_{l=1,l\neq i}^{n} a_{li} \leq a_{ii}$. This network system is a stochastic compartmental model representing a mass balance equation physically in which x_i denotes the mass (and hence a nonnegative quantity) of the ith subsystem of the compartmental system. Note that due to the time-average mass balance principle, the maximum amount of expected mass that can be transported cannot exceed the expected mass in a compartment. Then it follows that $a_{ii} \geq \sum_{l=1,l\neq i}^{n} a_{li}$, which interprets this time-average constraint in physics. The term $x_i(k) + d_iw_i(k)$ represents the imperfect information transmission between the $i$the subsystem and jth subsystem, resulting from noisy communication channels between subsystems.

To start with our discussion, we consider the discrete-time linear controlled system with stochastic noise given by

$$\begin{aligned} x(k+1) = Ax(k) + Bu(k) + D_1w(k), \\ x(0) = x_0, \quad k = 0,1,2,\ldots, \end{aligned} \tag{49}$$

where $x(k) = [x_1(k), \dots, x_n(k)] \in \mathbb{R}^n$ is the system state vector, $u(k) = [u_1(k), \dots, u_m(k)] \in \mathbb{R}^m$ is the control input, and $w(k) = [w_1(k), \dots, w_q(k)] \in \mathbb{R}^q$ is the q-dimensional standard Gaussian white noise vector. Here in general $x(0)$ is also a random variable.

The following definition is needed for our problem formulation. To proceed, let spec(A) denote the set of distinct eigenvalues for A and $\mathbb{C}$ denote the set of complex numbers.

Definition 4.2. *Let $A \in \mathbb{R}^{n\times n}$. A is called discrete-time semistable if* spec$(A) \subseteq \{s \in \mathbb{C} : |s| < 1\} \cup \{1\}$, *and if* $1 \in$ spec(A), *then 1 is semisimple, i.e., 1 is semisimple for A if and only if the algebraic multiplicity of 1 equals the geometric multiplicity of 1 for A [27, p. 322].*

The control aim here is to design a state feedback controller given by $u(k) = K[x(k) + D_2w(k)]$, such that the following design criteria are satisfied:

i) The closed-loop system (49) without noise is discrete-time semistable, i.e., $\tilde{A} = A + BK$ is discrete-time semistable.

ii) The performance functional

$$J(K) = \lim_{N\to\infty} \frac{1}{N+1}\mathbb{E}\Big\{\sum_{k=0}^{N}\big[(x(k)-x_\infty)^{\mathrm{T}}R_1(x(k)-x_\infty) + (u(k)-u_\infty)^{\mathrm{T}}R_2(u(k)-u_\infty)\big]\Big\} \tag{50}$$

is minimized, where $R_1 = E_1^{\mathrm{T}}E_1$, $R_2 = E_2^{\mathrm{T}}E_2$, $E_1^{\mathrm{T}}E_2 = 0$, $E_1 \in \mathbb{R}^{r\times n}$, $E_2 \in \mathbb{R}^{r\times m}$, $x_\infty = \lim_{k\to\infty}\mathbb{E}[x(k)]$, and $u_\infty = Kx_\infty$, where $\mathbb{E}$ denotes the expectation operator.

iii) $x_\infty = \frac{1}{\mathbf{u}^{\mathrm{T}}\mathbf{v}}\mathbf{u}\mathbf{v}^{\mathrm{T}}\mathbb{E}[x(0)]$ for some unit vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ satisfying $\mathbf{u}^{\mathrm{T}}\mathbf{v} \neq 0$.

Assuming that $x(0)$ is a random variable having a covariance matrix V, that is, $\mathbb{E}[x(0)x^{\mathrm{T}}(0)] - \mathbb{E}[x(0)]\mathbb{E}[x^{\mathrm{T}}(0)] = V$, $x(0)$ and $w(k)$ are independent for all $k \in \overline{\mathbb{Z}}_+$ and $\mathbf{1}^{\mathrm{T}}E = \mathbf{1}^{\mathrm{T}}$, the original optimal algorithm design problem can be converted into the following equivalent optimization problem.

Theorem 4.7. *Consider the coordinated resource allocation algorithm given by (48). Assume that D is invertible and $\Lambda = I_n$. Then solving the minimization problem*

$$\begin{aligned}&\min_E \big\{\operatorname{tr} S(E-I_n)^{\mathrm{T}}[R_1 + (E-I_n)^{\mathrm{T}}R_2(E-I_n)](E-I_n)\\ &S = S^{\mathrm{T}} > 0, S = ESE^{\mathrm{T}} + (E-I_n)DD^{\mathrm{T}}((E-I_n)^{\mathrm{T}}),\\ &\operatorname{rank}(E-I_n) = n-1, \quad \boldsymbol{v}^{\mathrm{T}}E = \boldsymbol{v}^{\mathrm{T}}, E\boldsymbol{u} = \boldsymbol{u}\big\}\end{aligned}$$

gives an optimal solution satisfying i)–iii), where rank(A) *denotes the rank of matrix A.*

Next, we penalize the graph structure in the optimization problem (51) and formulate the following mixed-binary nonlinear programming problem by introducing the communication cost into consideration.

$$\mathrm{Min}_{L,D}: \quad J \tag{51}$$

$$\mathrm{Sub}: \quad S = S^{\mathrm{T}} > 0, \quad \operatorname{rank}(L\circ D - I_n) = n-1, \tag{52}$$

$$S = L\circ DSL\circ D^{\mathrm{T}} + (L\circ D - I_n)DD^{\mathrm{T}}((L\circ D - I_n)^{\mathrm{T}}), \tag{53}$$

$$\mathbf{v}^{\mathrm{T}}L\circ D = \mathbf{v}^{\mathrm{T}}, \quad L\circ D\mathbf{u} = \mathbf{u}. \tag{54}$$

where $\circ$ denote the *Hadamard product* operation and $J = \lambda_1 \operatorname{tr} S(L\circ D - I_n)^{\mathrm{T}}[R_1 + (L\circ D - I_n)^{\mathrm{T}}R_2(L\circ D - I_n)](L\circ D - I_n) + \lambda_2\mathbf{1}^{\mathrm{T}}(L\circ D)\mathbf{1}$.

In the first part of J, the network parameters are optimized and in the second part of J, the network structure is optimized as well. Therefore, the new mixed-binary nonlinear programming problem will provide an novel approach < a
to investigate the interaction between network parameters and network structure simultaneously, and thus, significantly enhances the effectiveness of the network to handle damage mitigation and resource allocation under WMD attacks.

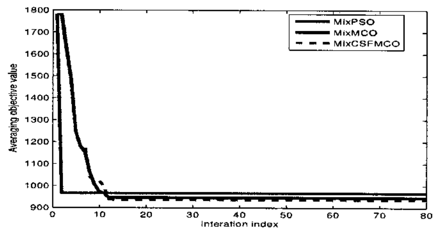
In the following, the mix-CSFMCO algorithm is used to solve the network design problem given by (51)–(54). Let $n = 10$, $\Lambda = I_{10}$, $D = I_{10}$, $R_1 = 8 \times I_{10} \times 10^4$, $R_2 = 5 \times I_{10} \times 10^4$, $V = 0$, $\lambda_1 = 1$, $\lambda_2 = 1000$, $x(0) =$

Table 6: Comparison Between PSO, MCO, and CSFMCO

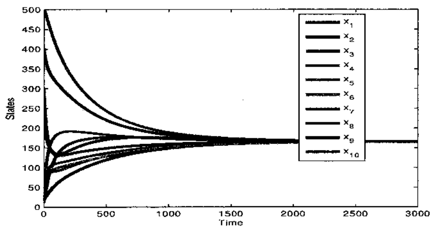
	Mix-PSO	Mix-MCO	Mix-CSFMCO
Min	2.8159E4	2.8088E4	2.8004E4
Max	3.1747E4	3.0990E4	3.0988E4
Average	2.9670E4	2.9381E4	2.9388E4
Median	1.0610E4	2.9459E4	2.9404E4

$[10, 20, 30, 40, 50, 100, 200, 300, 400, 500]^{\mathrm{T}}$, $\underline{v} = 0$, $\overline{v} = 1$, and $\mathbf{u} = \frac{1}{\sqrt{10}}[1, \ldots, 1]^{\mathrm{T}} \in \mathbb{R}^{10}$. Furthermore, we let $E = E^{\mathrm{T}}$, $E\mathbf{u} = \mathbf{u}$, and $\mathrm{rank}(E - I_{10}) = 9$.

$$E_{best} = \begin{bmatrix} 0.9822 & 0 & 0.0089 & 0 & 0 & 0 & 0.0089 & 0 & 0 & 0 \\ 0 & 0.9911 & 0 & 0 & 0.0089 & 0 & 0 & 0 & 0 & 0 \\ 0.0089 & 0 & 0.9733 & 0 & 0 & 0 & 0 & 0.0089 & 0.0089 & 0 \\ 0 & 0 & 0 & 0.9822 & 0 & 0.0089 & 0 & 0.0089 & 0 & 0 \\ 0 & 0.0089 & 0 & 0 & 0.9822 & 0 & 0 & 0.0089 & 0 & 0 \\ 0 & 0 & 0 & 0.0089 & 0 & 0.9911 & 0 & 0 & 0 & 0 \\ 0.0089 & 0 & 0 & 0 & 0 & 0 & 0.9911 & 0 & 0 & 0 \\ 0 & 0 & 0.0089 & 0.0089 & 0.0089 & 0 & 0 & 0.9733 & 0 & 0 \\ 0 & 0 & 0.0089 & 0 & 0 & 0 & 0 & 0 & 0.9822 & 0.0089 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0089 & 0.9911 \end{bmatrix}$$



(a) Comparison between mix-PSO, mix-MCO, and mix-CSFMCO for the network design problem (51)–(54)



(b) Velocities versus time

Figure 29: State trajectories versus time under the best solution obtained by mix-CSFMCO

4.4 A Speed-Up and Speed-Down Strategy for Swarm Optimization

Particle swarm optimization (PSO), firstly proposed in [53], has been widely used in various disciplines. As a swarm intelligence based optimization algorithm, a particle in PSO updates its velocity by comparing the difference between current position and local best position found by itself and the difference between current position and the global best position among all the particles, and then moves into the next position, after which the local best position and the global position will be updated simultaneously.

The update formulas for the PSO algorithm are shown in the following equation.

$$\begin{aligned} \mathbf{v}_{k+1}^{i,j} &= w\mathbf{v}_k^{i,j} + b_1 r_1(\mathbf{g}_{loc,i}^j - \mathbf{x}_k^{i,j}) + b_2 r_2(\mathbf{g}_k^j - \mathbf{x}_k^{i,j}) \\ \mathbf{x}_{k+1}^{i,j} &= \mathbf{x}_k^{i,j} + \mathbf{v}_{k+1}^{i,j}, \quad i = 1, \ldots, q, \quad j = 1, \ldots, n \end{aligned} \tag{55}$$

where $\mathbf{x}_k^{i,j}$ and $\mathbf{v}_k^{i,j}$ are the jth position and velocity element of particle i at iteration k, respectively, $\mathbf{g}_k^j$ is the jth element of the global best solution among all the particles and $\mathbf{g}_{loc,i}^j$ is the jth element of local best solution found by particle i, and w, b_1, b_2, r_1 and r_2 are positive constants.

Due to the great success of PSO, lots of research efforts have been conducted to improve the performance in both accuracy and efficiency. In this research, observed from the fish swarm behavior [61], a speed-up and speed-down (SUSD) mechanism is proposed for PSO as an extra force for the velocity towards both local and global best solutions. Moreover, this SUSD mechanism can be easily integrated into other PSO variants for performance enhancement.

As an example, multiagent coordination optimization (MCO) [14] is introduced and illustrated by using the SUSD mechanism.

Based on the observation that a fish in a group speeds up when the light intensity at its current position is relatively high and slows down as the light intensity decreases [61], a SUSD mechanism is proposed as follows.

$$SUSD = \lambda_1(f_j(\mathbf{g}_k^j) - f_j(\mathbf{x}_k^{i,j}))\mathtt{sign}(\mathbf{g}_k^j - \mathbf{x}_k^{i,j})$$

where $f_j(z) = f(x_1, \ldots, x_n)|_{x_j=z}$, $i = 1, \ldots, q$, $j = 1, \ldots, n$, q is the number of particles, n is the dimension of the objective function $f : \mathbb{R}^n \to \mathbb{R}$, $\lambda_1 < 0$ is a constant, $\mathtt{sign}(x) = 1$ if $x > 0$, $\mathtt{sign}(x) = -1$ if $x < 0$, and $\mathtt{sign}(0) = 0$.

Now this SUSD mechanism is applied to the PSO algorithm, and the update formulas for position and velocity are proposed in Equations (56–57).

$$\begin{aligned}
\mathbf{v}_{k+1}^{i,j} &= \mathbf{v}_k^{i,j} + b_1 r_1(\mathbf{g}_{loc,i}^j - \mathbf{x}_k^{i,j}) + b_2 r_2(\mathbf{g}_k^j - \mathbf{x}_k^{i,j}) \\
&\quad + \lambda_1(f_j(\mathbf{g}_k^j) - f_j(\mathbf{x}_k^{i,j}))\mathtt{sign}(\mathbf{g}_k^j - \mathbf{x}_k^{i,j}) && (56)\\
\mathbf{x}_{k+1}^{i,j} &= \mathbf{x}_k^{i,j} + \mathbf{v}_{k+1}^{i,j} && (57)
\end{aligned}$$

Multiagent coordination optimization (MCO) [14] is a novel swarm optimization algorithm by introducing the velocity consensus protocol and communication topology between particles into a PSO-like algorithm. The update formulas for position and velocity in MCO are shown in Equations (58–60).

$$\begin{aligned}
\mathbf{v}_{k+1}^{i,l} &= w\mathbf{v}_k^{i,j} + \check{\mu}\sum_{j\in\mathcal{N}_i}(\mathbf{x}_k^{j,l} - \mathbf{x}_k^{i,l}) + \check{\eta}\sum_{j\in\mathcal{N}_i} + (\mathbf{v}_k^{j,l} - \mathbf{v}_k^{i,l}) \\
&\quad \check{\kappa}(\mathbf{g}_k^l - \mathbf{x}_k^{i,l}) && (58)\\
\mathbf{x}_{k+1}^i &= \mathbf{x}_k^i + \mathbf{v}_{k+1}^i && (59)\\
\mathbf{g}_{k+1}^l &= \mathbf{g}_k^l + \check{\kappa}(\mathbf{g}_{loc,i}^l - \mathbf{g}_k^l) && (60)
\end{aligned}$$

where $i = 1, \ldots, q$, $l = 1, \ldots, n$, $\check{\eta} \sim U(0, 2\eta)$, $\check{\mu} \sim (0, 2\mu)$, $\check{\kappa} \sim (0, 2\kappa)$, $\eta, \mu, \kappa > 0$ are constants, $U(\cdot, \cdot)$ is the uniform distribution, $\mathcal{N}_i = \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}$ is the set of neighbors of node i and $\mathcal{E} \subseteq \mathcal{V}\times\mathcal{V}$ denotes the set of edges, which is the communication links between two particles, and the set of nodes $\mathcal{V} = \{1, \ldots, q\}$ denotes the index of the particles. Inspired by the SUSD mechanism, the SUSD-MCO algorithm is proposed in Equations (61-63).

$$\begin{aligned}
\mathbf{v}_{k+1}^{i,l} &= w\mathbf{v}_k^{i,j} + \check{\mu}\sum_{j\in\mathcal{N}_i}(\mathbf{v}_k^{j,l} - \mathbf{v}_k^{i,l}) \\
&\quad + \check{\mu}\sum_{j\in\mathcal{N}_i}(\mathbf{x}_k^{j,l} - \mathbf{x}_k^{i,l}) + \check{\kappa}(\mathbf{g}_k^l - \mathbf{x}_k^{i,l}) \\
&\quad + \lambda_1(f_l(\mathbf{g}_k^l) - f_l(\mathbf{x}_k^{i,l}))\mathtt{sign}(\mathbf{g}_k^l - \mathbf{x}_k^{i,l}) && (61)\\
\mathbf{x}_{k+1}^i &= \mathbf{x}_k^i + \mathbf{v}_{k+1}^i && (62)\\
\mathbf{g}_{k+1}^l &= \mathbf{g}_k^l + \check{\kappa}(\mathbf{g}_{loc,i}^l - \mathbf{g}_k^l) && (63)
\end{aligned}$$

where $l = 1, \cdots, n$ and $i = 1, \ldots, q$.

For numerical evaluation, 1000 particles are used to solve the 30-dimension benchmark functions in [62]. Specifically, we use the shifted sphere function, rotated Rosenbrock function, and rotated Griewank function to test the proposed SUSD mechanism. The shifted optimal solution is shown as X^*, where X^*=[98.7900 17.0400 25.7800 39.6800 7.4000 68.4100 40.2400 98.2800 40.2200 62.0700 15.4400 38.1300 16.1100 75.8100 87.1100 35.0800 68.5500 29.4100 53.0600 83.2400 59.7500 33.5300 29.9200 45.2600 42.2600 35.9600 55.8300 74.2500 42.4300 42.9400]. 20 executions of both PSO and SUSD-PSO algorithms solving the three benchmark functions have been conducted, and the results are shown in Table 7. It follows from the simulation results that the SUSD mechanism can largely improve the accuracy of PSO. Moveover, based on the results in Table 8 , the SUSD-MCO algorithm also improves the accuracy of MCO. Due to the page limitation, only the searching trajectories of PSO and SUSD-PSO algorithms solving the shifted sphere function is provided in Figure 30.

Table 7: Comparison between PSO and SUSD-PSO

Function	Min		Max	
	PSO	SUSD-PSO	PSO	SUSD-PSO
Sphere	1.64E-2	5.9263E-6	2.266	7.49E-2
Rosenbrock	2.10E-2	4.5727E-7	8.429E-1	3.1E-3
Griewank	2.0379E2	2.306E-1	2.3292E3	5.4249
Function	Median		Average	
	PSO	SUSD-PSO	PSO	SUSD-PSO
Sphere	2.135E-1	2.1E-3	4.543E-1	1.07E-2
Rosenbrock	1.0811E-1	1.7272E-4	1.597E-1	4.1827E-4
Griewank	8.5715E2	1.2871	1.0002E3	2.1007

Table 8: Comparison between MCO and SUSD-MCO

Function	Min		Max	
	MCO	SUSD-MCO	MCO	SUSD-MCO
Sphere	2.739E-1	8.6E-3	7.845E-1	2.56E-2
Rosenbrock	6.5E-3	4.0371E-4	7.32E-2	6.3E-3
Griewank	8.0663E3	9.2577E2	1.4405E4	1.0827E4
Function	Median		Average	
	MCO	SUSD-MCO	MCO	SUSD-MCO
Sphere	6.334E-1	1.55E-2	5.857E-1	1.45E-2
Rosenbrock	1.62E-2	2.3E-3	2.09E-2	2.8E-3
Griewank	1.0690E4	6.8156E3	1.1243E4	8.5082E3

4.5 Multiagent Coordination Optimization Based Model Predictive Control Strategy with Application to Balanced Resource Allocation

Model predictive control (MPC) [63–65], is a heuristic control strategy to find a consequence of best controllers during a finite-horizon regarding to certain performance functions of a dynamic system. Due to high efficiency of the strategy, MPC has been widely applied in various science and engineering disciplines, such as electrical engineering [66–68], mechanical engineering [69], and chemical engineering [70]. In general, MPC involves two main operations: estimation and optimization. Specifically, the current states and output of the system will be estimated and governed by system constraints, and the output states for the following prediction horizon will also be estimated. To move to the next controller, a constrained optimization problem regarding the system's performance will be solved, and the next control input will be calculated. As the iteration goes on, the system output tracks the reference output.

The most common cost function regarding the system performance is the tradeoff function between control input and convergence rate. To guarantee the stability of the closed-loop system, the terminal state cost functions are also introduced [71–75] into the cost function, and the asymptotic stability is assured by the standard Lyapunov theory. In addition, lots of research efforts have been conducted to improve the efficacy, accuracy, and robustness of the MPC

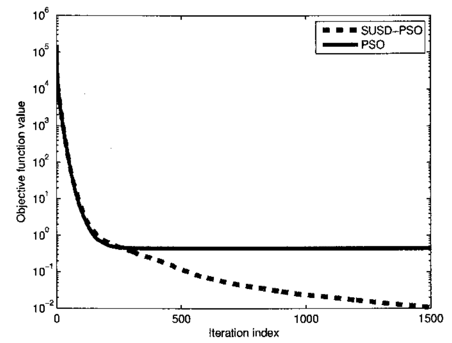


Figure 30: Test function: Shifted sphere function

strategy in this case [65, 76–78]. However, due to the distributed nature of information transmission and computation in network systems, this type of performance functions is not enough to measure the overall performance of the network system. For instance, the network structure is of vital importance when designing the system performance cost functions. First of all, the network structure effects the convergence rate of the whole network system. Secondly, have
due to the inherent limitation of the physical communication network, the constraints in the network structure has to be considered when designing the MPC strategies. Therefore, a new type of cost functions is proposed in this work, where the sparsity of the feedback control matrix is taken into consideration. The new MPC strategy will provide a new perspective for designing real-time optimal controllers. Moreover, the stability analysis is conducted by using the multiple Lyapunov function method, and the sufficient condition for guaranteeing the asymptotic stability of the closed-loop system is derived.

To solve the various forms of the cost functions, a general, computationally efficient and accurate optimization algorithm to handle non-convex or even mixed-integer nonlinear programming problems is highly needed. The vast majority of the optimizers developed in the literature uses the gradient based techniques and most of them can only address convex cost functions. Thus, they cannot be used for solving non-convex, nonlinear, mixed-integer complex optimization problems in real time. However, a recent advance in combining swarm intelligence and control of multiagent networks together gives a hope for solving such problems in real time. In particular, multiagent coordination optimization (MCO) [14] is a new heuristic, fast optimization algorithm inspired by particle swarm optimization (PSO) [79] and consensus protocols for multiagent coordination in [19, 32, 34]. Since the MCO algorithm requires only elementary mathematical operation and is computationally efficient in terms of both memory requirements and speed, it solves many nonlinear optimization problems quite efficiently [80]. Another promising advantage of the MCO algorithm is that the parallel techniques can be easily implemented for MCO to shorten the operation time in the hardware level [15]. Moreover, with the recently developed results on the MCO algorithm, different types of optimization problems can be efficiently solved by using the different types of the MCO optimizers, such as constrained MCO optimizer, binary MCO optimizer [81], and mixed-binary MCO optimizer [80]. Therefore, in this work, we will propose an MCO based MPC strategy to handle non-convex, large-scale nonlinear cost functions regarding the system performance in a time-efficient way.

As an application, balanced coordination for damage mitigation and resource allocation in network systems is considered in this work [15]. LQR based design approaches to address those problems have been extensively studied for both fixed-structure and dynamic-structure cases by converting the original design problem into a non-convex constrained optimization problem via semistability tools [57]. Since the process of resource allocation is highly dependent — fragment
on the scenarios for different time intervals. For instance, the attacked network communication links may be different at different time intervals, or the control input may be limited in certain periods due to the overall control budget. Therefore, how to design a sequence of structure-based controllers to provide the best resource allocation algorithm for different scenarios in different time intervals is very challenging. In this work, the MCO based MPC strategy provides this heuristic design approach to address both time-dependent or scenario-dependent controller design problems.

MPC [63–65] is to hold the system output at a reference value by adjusting the control signal during a finite-horizon meanwhile keeping the value of the performance function as low as possible. The block diagram of the MPC is shown in Fig. 31, where x_{ref}, u, ω, y, and $\bar{y}$ are the reference, control, measurement noise, measured output, and the true value of the output, respectively. MPC involves two main operations: estimation and optimization. More specifically, the current states and output of the system at time t will be estimated and governed by system constraints, and the output states for the following prediction horizon p are estimated: $t = \{kp, kp+1, \cdots, kp+p-1\}$, where $k = 0, 1, \cdots$, is the kth optimization algorithm execution index for the MPC strategy. To move to the next controller, a constrained optimization problem regarding the system's performance will be solved, and the sequence of inputs $u_{kp}, \cdots, u_M$ will be calculated, where $kp \leq M \leq kp+p-1$. The optimization problem for MPC is formulated as below.

$$\min: \sum_{t=kp}^{kp+p-1} x^{\mathrm{T}}(t)Qx(t) + u^{\mathrm{T}}(t)Ru(t) \tag{64}$$

$$\text{s.t.}: \quad C_1(u(t)) \leq U^b \tag{65}$$

$$C_2(x(t)) \leq S^b \tag{66}$$

$$x(t+1) = Ax(t) + Bu(t) + \omega(t) \tag{67}$$

where Q and R are positive definite matrices, C_1 and C_2 are the constraint functions for control input and states with the boundary U^b and S^b, A is the system dynamics matrix, B is the gain matrix for the control input, and $\omega(\cdot)$ denotes the exogenous disturbance.

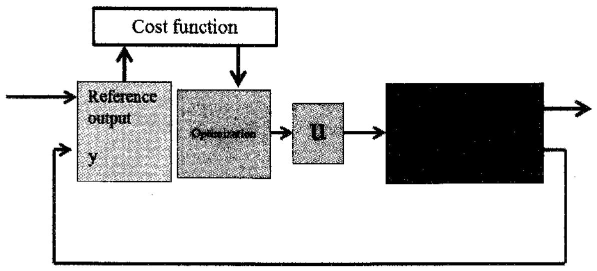


Figure 31: MPC strategy

In this work, undirected and/or directed graphs are used to represent a topology of communication networks. Specifically, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a *directed graph* (or digraph) denoting the communication network with the set of *nodes* (or vertices) $\mathcal{V} = \{1, \ldots, q\}$ involving a finite nonempty set denoting the agents, and the set of *edges* $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ involving a set of ordered pairs (i, j) denoting the direction of communication. A *graph* or *undirected graph* $\mathcal{G}$ is a directed graph for which the set of edges is symmetric. For graphs, we use unordered pairs $\{i, j\}$ for edges. The set of neighbors of node i is thus defined by $\mathcal{N}_i = \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}$.

Based on a synthetic composition of swarm intelligence [17] which simulates the bio-inspired behavior, and co-operative control of multiple agents [18, 32–34], a powerful, fast-convergence optimization solver called multiagent coordination optimization (MCO) [14–16] has been proposed. MCO starts with a set of random solutions for agents that can communicate with each other. The agents then move through the solution space based on the evaluation of their cost and neighbor-to-neighbor rules inspired by multiagent consensus protocols [18, 19, 32–34]. As the algorithm progresses, the agents will accelerate towards individuals with better cost values. The detail description of the MCO algorithm which minimizes a cost function $f(x)$ is shown in Algorithm 3.

Algorithm 3 The MCO Algorithm

```
for each agent $i = 1, \ldots, q$ do
  Initialize the agent's position with a uniformly distributed random vector: $x_i \sim U(\underline{x}, \overline{x})$, where $\underline{x}$ and $\overline{x}$ are the lower and upper boundaries of the search space;
  Initialize the agent's velocity: $v_i \sim U(\underline{v}, \overline{v})$, where $\underline{v}$ and $\overline{v}$ are the lower and upper boundaries of the search speed;
  Update the agent's best known position to its initial position: $p_i \leftarrow x_i$;
  If $f(p_i) < f(p)$ update the multiagent network's best known position: $p \leftarrow p_i$.
end for
repeat
  $k \leftarrow k + 1$;
  for each agent $i = 1, \ldots, q$ do
    Choose random parameters: $\check{\eta} \sim U(0, 2\eta), \check{\mu} \sim U(0, 2\mu), \check{\kappa} \sim U(0, 2\kappa), \eta, \mu, \kappa > 0$;
    Update the agent's velocity: $v_i \leftarrow v_i + \check{\eta} \sum_{j \in \mathcal{N}_i}(v_j - v_i) + \check{\mu} \sum_{j \in \mathcal{N}_i}(x_j - x_i) + \check{\kappa}(p - x_i)$;
    Update the agent's position: $x_i \leftarrow x_i + v_i$;
    for $f(x_i) < f(p_i)$ do
      Update the agent's best known position: $p_i \leftarrow x_i$;
      Update the multiagent network's best known position: $p \leftarrow p + \check{\kappa}(p_i - p)$;
      If $f(p_i) < f(p)$ update the multiagent network's best known position: $p \leftarrow p_i$;
    end for
  end for
until $k$ is large enough or the value of $f$ has small change
return $p$
```

Besides the continuous MCO algorithm, the binary MCO algorithm can also be developed. Similar to the continuous MCO algorithm, the binary MCO algorithm uses the same update formula for velocity, but takes the following formula for the position [81].

$$x_{i,j} \leftarrow \begin{cases} 1 & \text{if } r_{i,j} < \mathtt{sig}(v_{i,j}) \\ 0 & \text{otherwise} \end{cases} \tag{68}$$

where

$$\mathtt{sig}(x) = \frac{1}{1+e^{-x}} \tag{69}$$

is the Logistic function and $r_{i,j}$ is random number in $(0, 1)$.

Here we present a theoretic result on mean-value convergence of the iterative process described in Algorithm 3. In particular, we view the MCO algorithm as a switched linear system and then give some sufficient conditions for its mean-value convergence. To begin with, we define the following notation.

Definition 4.3. *Define a series of matrices $A_k^{[j]}$, $A^{[j]}$, $B_k^{[j]}$, and $B^{[j]}$ as (70),*

$$A_k^{[j]} = \begin{bmatrix} -\check{\mu}[k]L - \check{\kappa}[k]I_q & I_{nq} - \check{\eta}[k]L & \check{\kappa}[k]\boldsymbol{1}_{q\times 1} \\ -\check{\mu}[k]L - \check{\kappa}[k]I_q & -\check{\eta}[k]L & \check{\kappa}[k]\boldsymbol{1}_{q\times 1} \\ \check{\kappa}[k]E^{[j]}_{n\times nq} & \boldsymbol{0}_{n\times nq} & -\check{\kappa}[k]I_n \end{bmatrix}, \quad A^{[j]} = \begin{bmatrix} -\mu L - \kappa I_q & I_{nq} - \eta L & \kappa\boldsymbol{1}_{q\times 1} \\ -\mu L - \kappa I_q & -\eta L & \kappa\boldsymbol{1}_{q\times 1} \\ \kappa E^{[j]}_{n\times nq} & \boldsymbol{0}_{n\times nq} & -\kappa I_n \end{bmatrix}, \tag{70}$$

$B_k^{[j]} = A_k^{[j]} + \begin{bmatrix} \boldsymbol{0}_{2nq\times nq} & \boldsymbol{0}_{2nq\times nq} & \boldsymbol{0}_{2nq\times n} \\ (1-\check{\kappa}[k])E^{[j]}_{n\times nq} & \boldsymbol{0}_{n\times nq} & -(1-\check{\kappa}[k])I_n \end{bmatrix}$, *and* $B^{[j]} = A^{[j]} + \begin{bmatrix} \boldsymbol{0}_{2nq\times nq} & \boldsymbol{0}_{2nq\times nq} & \boldsymbol{0}_{2nq\times n} \\ (1-\kappa)E^{[j]}_{n\times nq} & \boldsymbol{0}_{n\times nq} & -(1-\kappa)I_n \end{bmatrix}$, *respectively, where $j = 1, \dots, q$, $k = 0, 1, 2, \dots$, $\check{\mu}[\cdot] \sim U(0, 2\mu)$, $\check{\eta}[\cdot] \sim U(0, 2\eta)$, $\check{\kappa}[\cdot] \sim U(0, 2\kappa)$, $\mu, \eta, \kappa > 0$, $\boldsymbol{0}_{m\times n}$ denotes the $m \times n$ zero matrix, $\boldsymbol{1}_{m\times n}$ denotes the $m \times n$ matrix whose entries are all ones, L denotes the Laplacian matrix of certain graph topology $\mathcal{G}$, and $E^{[j]}_{n\times nq} \in \mathbb{R}^{n\times nq}$ denotes a block-matrix whose jth block-column is I_n and the rest block-elements are all zero matrices.*

Next, consider the following discrete-time switched linear model to describe the iterative process in Algorithm 3:

$$x_i[k+1] = x_i[k] + v_i[k+1], \quad x_i[0] = x_{i0}, \tag{71}$$

$$\begin{aligned} v_i[k+1] = {} & v_i[k] + \check{\eta}[k]\sum_{j\in\mathcal{N}_i}(v_j[k] - v_i[k]) \\ & + \check{\mu}[k]\sum_{j\in\mathcal{N}_i}(x_j[k] - x_i[k]) \\ & + \check{\kappa}[k](p[k] - x_i[k]), \quad v_i[0] = v_{i0}, \end{aligned} \tag{72}$$

$$p[k+1] = p[k], \quad p[k] \notin \mathcal{Z}_p, \quad p[0] = p_0, \tag{73}$$

$$p[k+1] = x_j[k], \quad p[k] \in \mathcal{Z}_p, \quad k = 0, 1, 2, \dots, \tag{74}$$

where $i = 1, \dots, q$, $x_i \in \mathbb{R}^n$, $v_i \in \mathbb{R}^n$, $p \in \mathbb{R}^n$, $\check{\mu}[\cdot] \sim U(0, 2\mu)$, $\check{\eta}[\cdot] \sim U(0, 2\eta)$, $\check{\kappa}[\cdot] \sim U(0, 2\kappa)$, $\mu, \eta, \kappa > 0$, $\mathcal{Z}_p = \{p \in \mathbb{R}^n : f(x_j) < f(p)\}$, and $x_j = \arg\min_{1\le i\le q} f(x_i)$. Let $Z[k] = [x_1^{\mathrm{T}}[k], \dots, x_q^{\mathrm{T}}[k], v_1^{\mathrm{T}}[k], \dots, v_q^{\mathrm{T}}[k], p^{\mathrm{T}}[k]]^{\mathrm{T}} \in \mathbb{R}^{2nq+n}$.

Note that (71)–(74) can be rewritten as the compact form

$$Z[k+1] = (I_{2nq+n} + A_k^{[j_k]})Z[k], \quad Z[k] \notin \mathcal{S}, \tag{75}$$

$$Z[k+1] = (I_{2nq+n} + B_k^{[j_k]})Z[k], \quad Z[k] \in \mathcal{S}, \tag{76}$$

where $j_k \in \{1, \dots, q\}$ is selected based on $\mathcal{Z}_p$ and $\mathcal{S}$ is defined in terms of $\mathcal{Z}_p$. An important property of MCO in Algorithm 3 is that it always generates a monotonically decreasing sequence $\{f(p[k])\}_{k=0}^{\infty}$, i.e., $f(p[k+1]) \le f(p[k])$ for every $k = 0, 1, 2, \dots$.

Definition 4.4. *The iterative process (71)–(74) is called convergent in the mean if $\lim_{k\to\infty}\mathbb{E}[Z[k]]$ exists for every $Z[0] \in \mathbb{R}^{2nq+n}$, where $\mathbb{E}[\cdot]$ denotes the expectation operator.*

Now we have the main result for the mean-value convergence of the iterative process described in Algorithm 3.

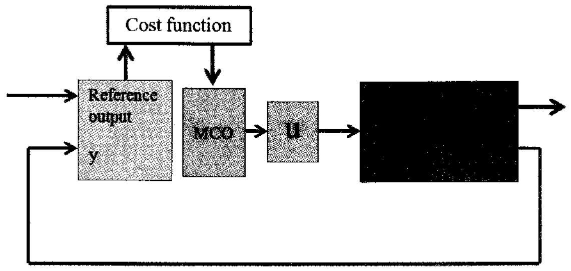


Figure 32: MCO based MPC strategy

Table 9: Comparison between MCO and PSO

MPC	$p = 1$	$p = 5$
PSO-based MPC	2.2202E4	2.0482E5
MCO-based MPC	1.7857E4	1.4628E5

Theorem 4.8. *Consider the iterative process (71)–(74). Let* $h_1^\dagger = \min\Big\{-\frac{\lambda+\bar{\lambda}}{|\lambda|^2} : \lambda \in \{-\kappa, -\kappa \pm \sqrt{\kappa^2-\kappa}, -\frac{\kappa}{2} \pm \frac{1}{2}\sqrt{\kappa^2-4\kappa}, \lambda \in \mathbb{C} : \forall \frac{\lambda^2+\kappa\lambda+\kappa}{\eta\lambda+\mu\lambda+\mu} \in \operatorname{spec}(-L)\backslash\{0\}\}\Big\}$ *and* $h_2^\dagger = \min\Big\{-\frac{\lambda+\bar{\lambda}}{|\lambda|^2} : \lambda \in \{-\frac{\kappa}{2} \pm \frac{1}{2}\sqrt{\kappa^2-4\kappa}, \lambda_1, \lambda_2 \in \mathbb{C} : \forall \frac{\lambda_1^2+\kappa\lambda_1+\kappa}{\eta\lambda_1+\mu\lambda_1+\mu} \in \operatorname{spec}(-L)\backslash\{0\}, \lambda_2^3 + (1+\kappa)\lambda_2^2 + \kappa\lambda_2 + \kappa = 0\}\Big\}$. *If* $\min\{h_1^\dagger, h_2^\dagger\} > 1$, $\|I_{2nq+n} + A^{[j]}\| \leq 1$, $\|I_{2nq+n} + B^{[j]}\| \leq 1$, $\ker((A^{[j]})^{\mathrm{T}}A^{[j]} + (A^{[j]})^{\mathrm{T}} + A^{[j]}) = \ker((A^{[j]})^{\mathrm{T}}A^{[j]} + (A^{[j]})^2)$, *and* $\ker((B^{[j]})^{\mathrm{T}}B^{[j]} + (B^{[j]})^{\mathrm{T}} + B^{[j]}) = \ker((B^{[j]})^{\mathrm{T}}B^{[j]} + (B^{[j]})^2)$ *for every* $j = 1, \ldots, q$, *where* $\ker(A)$ *denotes the kernel of A, then the iterative process (71)–(74) is convergent in the mean. Furthermore,* $\lim_{k\to\infty}\mathbb{E}[x_i[k]] = p^\dagger$, $\lim_{k\to\infty}\mathbb{E}[v_i[k]] = \boldsymbol{0}_{n\times1}$, *and* $\lim_{k\to\infty}\mathbb{E}[p[k]] = p^\dagger$ *for every* $x_{i0} \in \mathbb{R}^n$, $v_{i0} \in \mathbb{R}^n$, $p_0 \in \mathbb{R}^n$, *and every* $i = 1, \ldots, q$, *where* $p^\dagger \in \mathbb{R}^n$.

Although Theorem 4.8 focuses on unconstrained optimization problems, it also holds for constrained optimization problems by assuming that $Z[k]$ is in the feasible region.

In this part, we propose the MCO based MPC strategy for dynamical systems. Here the MCO algorithm is used as an optimization solver to handle large-scale complex performance functions regarding the dynamical system. The flow diagram is shown in Fig. 32. In this strategy, the MCO algorithm is equipped as the optimization solver to solve the non-convex nonlinear optimization problem regarding the system's performance. First of all, we will numerically illustrate the proposed MCO based MPC strategy for a particular system. The system we consider here is given by

$$y(k+1) = -0.3y(k) + 0.5\sin(0.6\pi u(k))u(k) \tag{77}$$

with the cost function

$$J = \sum_{n=1}^{p} r_1(\hat{y}(n+1) - y(n+1))^2 + r_2(u(n))^2 \tag{78}$$

where $\hat{y}(\cdot)$ is the reference output, and r_1 and r_2 are positive constants.

Both PSO and MCO based MPC strategies are compared to solve this unconstrained MPC problem and the results are shown in Fig. 33 and Fig. 34 with different prediction horizons. In particular, the green line is the reference output for the system, and the red line is the output for the system under the MCO based MPC strategy while the blue line is the output for the system under the PSO based MPC strategy. Moreover, the overall values of the cost functions for all iterations are added and compared in Table 9. It can be concluded that the MCO based MPC strategy shows a better performance with lower transient oscillation than the PSO based MPC strategy.

To guarantee the stability of the proposed MCO based MPC algorithm, the terminal cost is also added into the system performance function shown in (79) [72]. The system we consider here is $x(t+1) = Ax(t) + Bu(t) + \omega(t)$, where

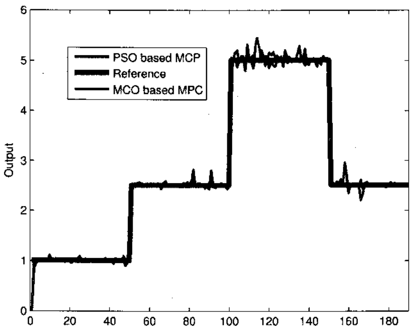


Figure 33: The case when prediction horizon is 2

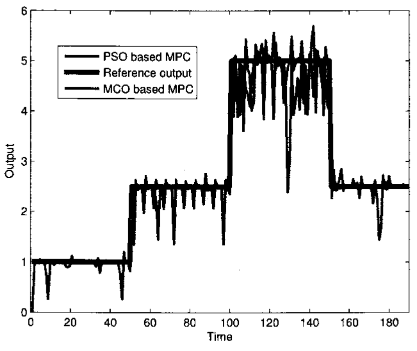


Figure 34: The case when prediction horizon is 5

$x \in \mathbb{R}^n$ is the state variable, $u \in \mathbb{R}^m$ is the control input, $\omega \in \mathbb{R}^n$ is the exogenous disturbance, and $t = 0, 1, 2, \ldots$. It is assumed that (A, B) is stabilizable and the origin is the equilibrium point, if the equilibrium point $x_e \neq \mathbf{0}$, one can always shift the origin of the system to x_e.

$$\begin{aligned}
\min: \quad & \sum_{i=t}^{t+p-1} x^{\mathrm{T}}(i)Qx(i) + u^{\mathrm{T}}(i)Ru(i) \\
& \quad + x^{\mathrm{T}}(t+p)Px(t+p) && (79) \\
\text{s.t.}: \quad & C_1(u(i)) \leq U^b && (80) \\
& C_2(x(i)) \leq S^b && (81) \\
& C_3(x(t+p)) \leq X^b && (82) \\
& x(i+1) = Ax(i) + Bu(i) + \omega(i) && (83)
\end{aligned}$$

where $x^{\mathrm{T}}(t+p)Px(t+p)$ is the terminal cost function with terminal state constraint $C_3(x(t+p))$ and the boundary X^b.

Definition 4.5. *Consider the MCO based MPC algorithm. This algorithm is called averagely semistable if*

i) the closed-loop system (83) is Lyapunov stable;

ii) the iterative process (71)–(74) of MCO to solve the corresponding optimization problem in MPC is convergent in the mean.

Theorem 4.9. *Consider the MCO based MPC algorithm with the input form $u(t) = -K(t)x(t)$, where $K : \{0, 1, 2, \ldots\} \to \mathbb{R}^{m\times n}$. Assume that the initial state of the system is a feasible solution to the problem (79) at $t = 0$. Furthermore, assume that the conditions in Theorem 4.8 hold for (79). If there exist an optimal solution of the finite horizon optimal control problem and a positive definite matrix $P = P^{\mathrm{T}}$ satisfying $(A - BK(t))^{\mathrm{T}}P(A - BK(t)) \leq P$ for every $t = 0, 1, 2, \ldots$, then the MCO based MPC algorithm is averagely semistable.*

Due to the wide applications of the networked system, for instance, power system and sensor network system, the communication topology is of vital importance, therefore, network structure is becoming a main concern regarding to the networked system performance. In the following, we propose a network structure based performance function to handle the interaction between network structure and network parameters.

$$\begin{aligned}
\min: \quad & J(K_k) = \sum_{t=kp}^{kp+p-1} x^{\mathrm{T}}(t)Qx(t) + u^{\mathrm{T}}(t)Ru(t) \\
& + x^{\mathrm{T}}(kp+p)P_k x(kp+p) + \mathtt{F}(K_k) && (84) \\
\text{s.t.}: \quad & C_1(u(t)) \leq U^b && (85) \\
& C_2(x(t)) \leq S^b && (86) \\
& C_3(x(kp+p)) \leq X^b && (87) \\
& C_4(\mathtt{F}(K_k)) \leq F^b && (88) \\
& x(t+1) = Ax(t) + Bu(t) + \omega(t) && (89) \\
& u(t) = -K_k x(t), && (90) \\
& \forall t \in \{kp, kp+1, \cdots, kp+p-1\} && (91)
\end{aligned}$$

where K_k is the state feedback gain matrix at the kth optimization algorithm execution index in MPC for $\forall t \in \{kp, kp + 1, \cdots, kp + p - 1\}$ and K_k satisfies $(A - BK_k)^{\mathrm{T}}P_k(A - BK_k) - P_k \leq -(Q + K_k^{\mathrm{T}}RK_k)$ for some positive definite matrix $P_k = P_k^{\mathrm{T}}$, and $\mathtt{F} : \mathbb{R}^n \to [0, \infty)$ is a function of the sparsity pattern for feedback matrix K, such that $\mathtt{F}(K) = \sum_{i,j} |K(i, j)|$ and $K_{i,j}$ denotes the $(i, j)$the entry of matrix K, and C_4 is the constraint function for $\mathtt{F}$ with the boundary F^b.

Theorem 4.10. *Assume that the initial state of the system (89) is a feasible solution of the problem (84) at $t = 0$ and $\omega = 0$. If there exists an optimal solution of the finite horizon optimal control problem (84) and $P_0 \geq P_1 \geq P_2 \geq \cdots \geq P_k$ for every $k = 0, 1, 2, \ldots$, where P_k is the terminal penalty matrix at the kth optimization algorithm execution index in MPC, then $\lim_{t\to\infty} x(t) = \mathbf{0}_{n\times 1}$ when $\omega(t) \equiv \mathbf{0}_{n\times 1}$.*

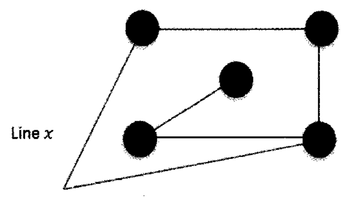


Figure 35: Topology in first scenario

Balanced coordination for damage mitigation and resource allocation in network systems enhances rapid dissemination of network resources and self-healing of the network, which leads to significant reduction of the threats imposed by malicious attacks. LQR based design approaches to address those problems have been extensively studied for both fixed-structure and dynamic-structure cases by converting the original design problem into a non-convex constrained optimization problem via semistability tools [57]. Since the process of resource allocation is highly dependent on scenarios for different time intervals. Phrase For instance, the attacked network communication links may be different at different time intervals, or the control input may be limited in certain periods due to the overall control budget. Therefore, how to design a sequence of structure-based controllers to provide the best resource allocation algorithm for different scenarios in different time intervals is highly challenging. However, the LQR based approaches cannot handle this design problem directly. Fortunately, the MPC strategy can provide a heuristic design approach to address both time-dependent or scenario-dependent controller design problems. Thus, in this part, we will illustrate how to use our proposed MCO based MPC strategy to address this complex design problem regarding different time intervals or scenarios.

The iterative algorithm design for coordinated resource allocation in network systems is given by the equation (47) or in the vector form, (48), where $x(k) = [x_1(k), \dots, x_n(k)]^{\mathrm{T}}$,$D = \mathrm{diag}\,[d_1, \dots, d_n]$, for $i, j = 1, \dots, n$, $w(k) = [w_1(k), \dots, w_n(k)]^{\mathrm{T}} \in \mathbb{R}^n$ denotes the standard white noise vector, $L_{(i,j)}$ represents the (i, j)th element of the Laplacian matrix L and $\Lambda = \mathrm{diag}\,[a_{11}, \dots, a_{nn}] \in \mathbb{R}^{n \times n}$,.

$$E_{(i,j)} = \begin{cases} a_{ii} - \sum_{l=1, l\neq i}^{n} L_{(l,i)} a_{li}, & i = j, \\ L_{(i,j)} a_{ij}, & i \neq j, \end{cases} \tag{92}$$

a_{ij} is the parameter that we need to design, $a_{ij} \geq 0$ and $\sum_{l=1,l\neq i}^{n} a_{li} \leq a_{ii}$.

Let $A = \Lambda$ and $B = I$ in (89), and $\omega(t) = (E - \Lambda)Dw(t)$ where I denotes the identity matrix. The MCO based MPC strategy for network balanced resource allocation for $\forall t \in \{kp, kp+1, \cdots, kp+p-1\}$ can be formulated as follows.

$$\begin{aligned} \min : \quad & \sum_{t=kp}^{kp+p-1} (x(t) - x_\infty)^{\mathrm{T}} Q (x(t) - x_\infty) + u^{\mathrm{T}}(t) R u(t) \\ & + x^{\mathrm{T}}(kp+p) P x(kp+p) \\ \text{s.t.} : \quad & C_1(u(t)) \leq U^b && (93) \\ & C_2(x(t)) \leq S^b && (94) \\ & C_3(x(kp+p)) \leq X^b && (95) \\ & x(t+1) = \Lambda x(t) + (E - \Lambda)x(t) + (E - \Lambda)Dw(t) && (96) \\ & x_\infty = \frac{1}{\mathbf{u}^{\mathrm{T}}\mathbf{v}} \mathbf{u}\mathbf{v}^{\mathrm{T}} \mathbb{E}[x(0)] && (97) \end{aligned}$$

where $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ are unite vectors and $\mathbf{u}^{\mathrm{T}}\mathbf{v} \neq 0$.

In the first scenario, we consider a five-node network system with $\mathbf{u} = \mathbf{v} = \mathbf{1} \in \mathbb{R}^5$, and the network topology is shown in Fig. 35. The topological link for the system is destroyed at time $t = 2$, and the topological link is recovered at time $t = 15$. $R = Q = I \times 10^4$. The MCO based MPC strategy in this scenario is shown in Fig. 36.

In the second scenario, we consider the case with a dynamic structure for the controllers. Specifically, we penalize the network structure in the cost function, shown in (98). Let $A = \Lambda$ and $B = I$ in (89), $\omega(t) = (E_k - \Lambda)Dw(t)$, and $E_k - \Lambda$ is the feedback matrix for $\forall t \in \{kp, kp+1, \cdots, kp+p-1\}$ at the kth optimization algorithm execution index

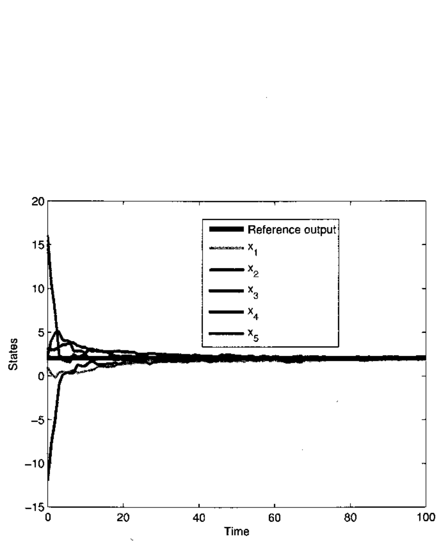


Figure 36: Scenario 1 with $p = 2$

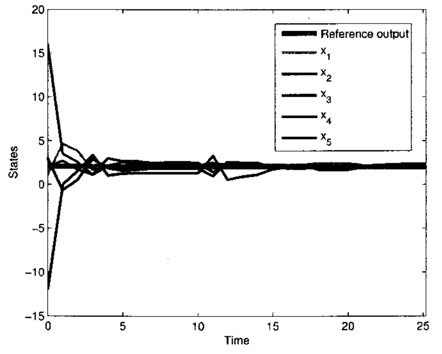


Figure 37: The first case in scenario 2

in MPC. Hence, the optimization problem becomes a new type optimization problem called "mixed binary nonlinear optimization" (MBNO) problem. Since the most MPC strategies are developed to address quadratic optimization problems, they fail to apply in this case. However, based on the recently developed results on the MCO-based MBNO solver [80], the MCO based MPC can provide an efficient way to handle this scenario.

$$\begin{aligned}
\min: \ & \sum_{t=kp}^{kp+p-1} (x(t) - x_\infty)^{\mathrm{T}} Q(x(t) - x_\infty) + u^{\mathrm{T}}(t) R u(t) \\
& + x^{\mathrm{T}}(kp+p) P_k x(kp+p) + \mathrm{F}_s(E_k - \Lambda) && (98) \\
\text{s.t.}: \ & C_1(u(t)) \le U^b && (99) \\
& C_2(x(t)) \le S^b && (100) \\
& C_3(x(kp+p)) \le X^b && (101) \\
& C_4(\mathrm{F}((E_k - \Lambda))) \le F^b && (102) \\
& x(t+1) = \Lambda x(t) + (E_k - \Lambda) x(t) + (E_k - \Lambda) D w(t) && (103) \\
& x_\infty = \frac{1}{\mathbf{u}^{\mathrm{T}}\mathbf{v}} \mathbf{u}\mathbf{v}^{\mathrm{T}} \mathbb{E}[x(0)] && (104) \\
& \forall t \in \{kp, kp+1, \cdots, kp+p-1\} && (105)
\end{aligned}$$

where $\mathbf{u}^{\mathrm{T}}\mathbf{v} \neq 0$. The first case we consider here is $\mathrm{F}_s = 0$, which means there is no constraint for the network structure. By using the MCO based MPC strategy, the system output is shown in Fig. 37. Throughout the control process, the topology for the network is always a fully connected topology shown in Fig. 38.

The second case we consider here is to assess the impact of the topology by letting $\mathrm{F}_s(K) = \sum_{i,j} |\mathtt{sign}(K_{i,j})|$, which

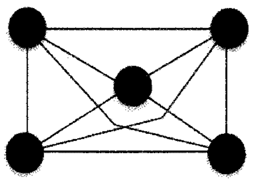


Figure 38: Topology for case 1 for $t = 1, 2, \cdots, 25$.

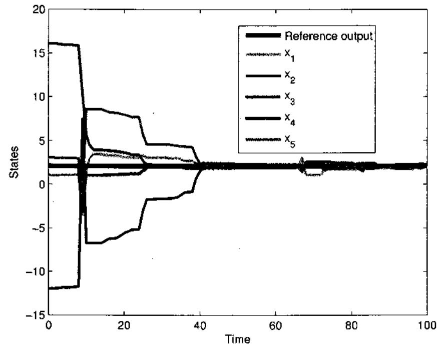
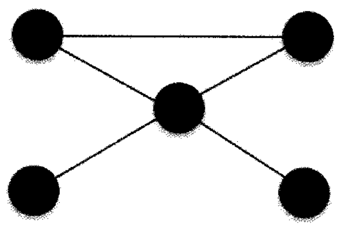


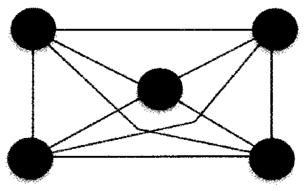
Figure 39: The second case in scenario 2

stands for the constraint for the conjunction links, where `sign` is the sign function and $K_{i,j}$ denotes the (i, j)th entry for matrix K. By using the MCO based MPC strategy, the system output is shown in Fig. 39. Throughout the control process, the topology for the network is always changing. We pick the topologies at $t = 1$, $t = 50$, and $t = 90$ as an example, which are shown in Fig. 40. When $t = 50$, the topology is a fully connected topology which means that the convergence rate of the dynamic system is of high priority compared with the network structure constraint, and when the time goes to 90, the network structure constraint plays a more significant role than the convergence rate.

4.6 Power System Vulnerability Analysis Using High Performance Swarm Computing

For the last several months we have been developing code to address the network vulnerability problem related to electrical power networks. To identify small groups of lines, whose removal would cause a server blackout, is an important issue for secure operation of the electric power grid. Therefore, an MBNLP problem is formulated in [82] by optimizing the total volume of load shed varying from line-cut variables and the phase angle variables. A lossless power system network with m bus and n lines is considered, and the voltages at the buses are assumed to be fixed and the dependence of real power injection at buses on the phase angle variables can be fully described by active power constraints. The power system model is developed as $A^{\mathrm{T}} B \sin(A\theta) - P = 0$, where P is a vector of power injections, A is the node-arc incidence matrix of the topology of the power system, B is a diagonal matrix whose diagonal entries correspond to line admittances, θ is a vector of phase angle variables, and $\sin(A\theta)$ denotes a vector whose ith component is $\sin((A\theta)_i)$. To further study the changes in the topology, a binary-valued line parameter γ_i, is introduced to indicate whether the ith line is in service, specifically, $\gamma_i = 1$ if the line is out of service and $\gamma_i = 0$ if the line is in service. Matrix Γ is defined as a diagonal matrix whose diagonal entries corresponding to $(1 - \gamma_i)$, so the power flow model can


(a) $t = 1$


(b) $t = 50$.

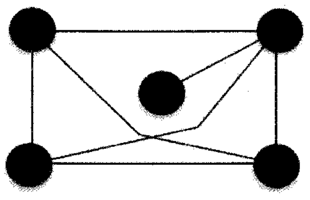

(c) $t = 90$.

Figure 40: Topologies at different time in the second case

be rewritten as $A^{\mathrm{T}}B\Gamma\sin(A\theta) - P = 0$. Hence, the optimal load shedding strategy is formulated in [82] as

$$
\begin{aligned}
\min_{\theta,Z} \quad & -e^{\mathrm{T}}Z^{g} && (106)\\
\text{s.t.} \quad & A^{\mathrm{T}}B\Gamma\sin(A\theta) - (P + Z) = 0, && (107)\\
& P^{l} \le P^{l} + Z^{l} \le 0, && (108)\\
& 0 \le P^{g} + Z^{g} \le P^{g}, && (109)\\
& -\frac{\pi}{2} \le A\theta \le \frac{\pi}{2} && (110)
\end{aligned}
$$

where e denotes a column vector whose elements are all ones, P^g denotes power injections in generations, P^l denotes power injections in loads, Z^g denotes changes in generations, Z^l denotes changes in loads, $P = [(P^g)^{\mathrm{T}}, (P^l)^{\mathrm{T}}]^{\mathrm{T}}$, and $Z = [(Z^g)^{\mathrm{T}}, (Z^l)^{\mathrm{T}}]^{\mathrm{T}}$.

Next, the power network vulnerability analysis problem is formulated as a bilevel mixed-binary nonlinear optimization problem [82], where in the outer level we look for the critical lines, which corresponds to the combinatorial part of the problem, and in the inner level we measure the blackout severity by solving the load shedding problem, which corresponds to the nonlinear part of the problem. To formally state this problem, let $\mathcal{LS}(A, B, \Gamma, \theta, P)$ denote an instance of the load shedding problem in (106)–(110), and let $\arg\min \mathcal{LS}(A, B, \Gamma, \theta, P)$ denote an optimal solution to this problem. Then the power system vulnerability problem can be defined as follows.

$$
\begin{aligned}
\min_{\theta,Z,\gamma} \quad & -e^{\mathrm{T}}\gamma\\
\text{s.t.} \quad & Z = \arg\min \mathcal{LS}(A, B, \Gamma, \theta, P),\\
& -e^{\mathrm{T}}Z^{g} \ge S,\\
& \gamma_i \in \{0, 1\} \quad \text{for} \quad i = 1, 2, \cdots, m
\end{aligned}
$$

where $S > 0$ is a specified severity.

We have coded this problem as a penalized optimization problem with the first four bullets used as the penalty term. We have been experimenting with a relaxation of the problem with the integer constraints being relaxed to $0 \leq \gamma_i \leq 1$. While we have limited experience with this problem we have found that optimal solutions that we have computed are nearly integer. Many times they can be rounded to integer solutions. Our optimization code is based on the particle swarm algorithm. We have been able to solve problems with roughly 30 nodes and 50 edges. We have developed an openMP version of the code (in C), which yields nearly linear scaling (Table 10).

Table 10: Shown is a small parallel speed up test using 1000 particles. The number of iterations is the rows and the thread counts are the columns. The results are in seconds.

iterations/threads	1	4	8	12	16
100	3.957	1.192	0.706	0.549	2.025
500	19.647	5.757	3.358	2.586	1.733
1000	39.277	11.532	6.61	5.049	3.703

In addition we are working with graphical software that will show the connection graph and we expect to be able to annotate this graph based on optimization results (Figure 41). Furthermore, we have solved the power system vulnerability problem with 39 nodes and are currently tuning our code for even larger problems. We have developed an openMP version of the code (in C) and used some graphical software to illustrate our result in Figure 42. The HPCC at TTU has installed several nodes with dual Xeon Phi. We have experimented with porting our code to this environment.

Finally, all the relevant algorithms developed above have been converted into pseudo code or C code in our developed software package to DTRA. The prototype of this software package has been tested for a preliminary version.

5 Accomplishments for Year 2 (07/01/2014-08/29/2015)

The following research accomplishments were achieved over the second year duration of this project.

5.1 ε Constrained CCPSO with Different Improvement Detection Techniques for Large-Scale Constrained Optimization

Nature-inspired meta-huristics such as evolutionary algorithms and swarm intelligence algorithms have been shown to be effective optimization techniques [83, 84], especially for complicated problems such as nonconvex nonlinear optimizations with an unknown objective function.

In recent years, there have been many studies on nature-inspired algorithms for large-scale (e.g., 100–1000D) unconstrained optimization [85–90], e.g., the MLCC method [90], CCPSO2 algorithm [88], and methods based on variable interactions [85, 89]. There has been also a lot of attention on algorithms for constrained optimization at smaller scale (e.g., 10–30D) [91–93]. However, large-scale constrained optimization using nature-inspired algorithms is still a new and under-explored area, and to the best of the authors' knowledge, by far there is no nature-inspired algorithm known to be capable of solving general large-scale real-valued constrained optimization problems. Clearly, for solving general large-scale constrained optimization problems, we need to face both the difficulties in searching the minimum of the large-dimensional objective function and in locating the feasible region defined by the large-dimensional constraint functions. Many practical applications, however, require such optimization techniques. In power grid systems, identifying highly important lines are crucial for system security, and such vulnerability analysis usually involves solving optimization problems of a large number of variables [94]. For example, the IEEE Three-Area RTS-96 system has a total of 73 buses and 185 transmission lines, and the IEEE 118-bus system has a total of 185 lines and 19 generation buses [94, 95].

Here we combine an algorithm in large-scale unconstrained optimization area, i.e., CCPSO2, with an effective constrained optimization technique, i.e., the ε constrained method, and propose a new framework, i.e., εCCPSO, for solving large-scale constrained optimization problems. On one hand, CCPSO2 is a cooperative coevolutionary particle swarm optimization (CCPSO) algorithm using a random grouping technique, based on the cooperative coevolution (CC) strategy [96, 97], and has shown promising solution capability for large-scale unconstrained optimization. On the other hand, in [91] for 10–30D constrained optimization, the ε constrained method is adopted by a differential

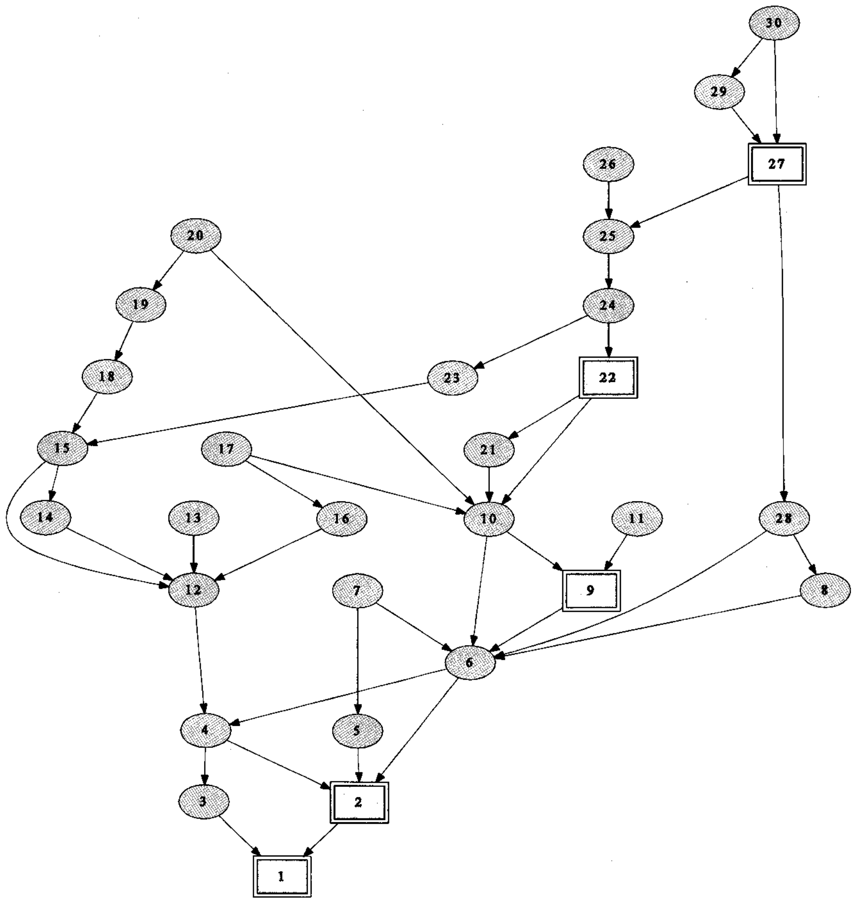


Figure 41: A connection graph topology created from a 30 bus power system. The line from 5 to 2 has been cut and is marked in red.

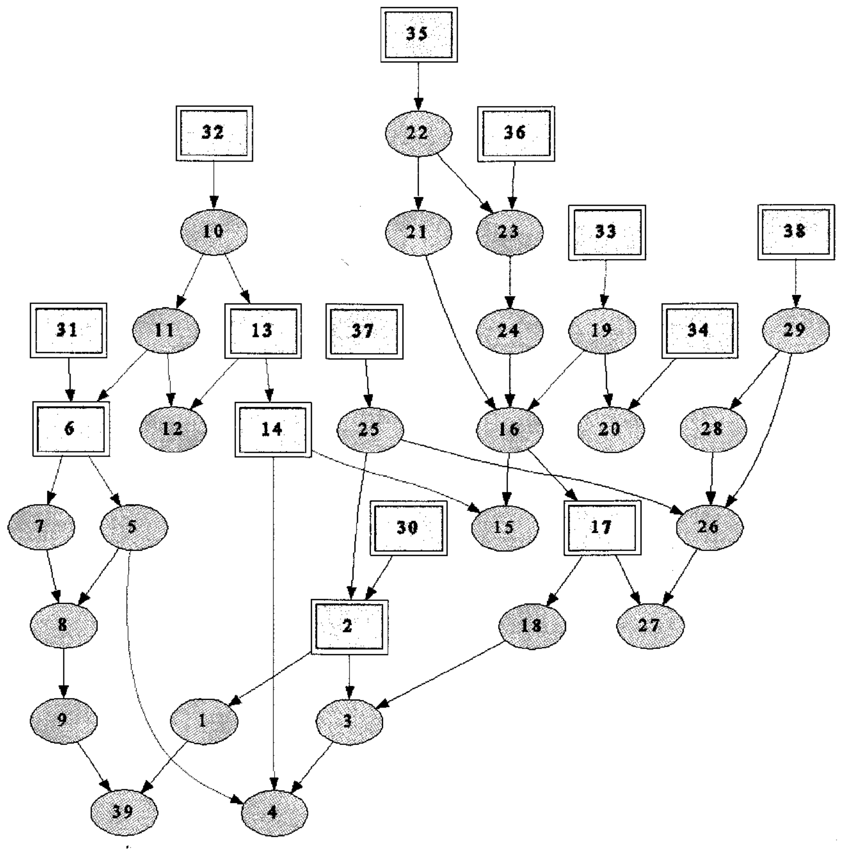


Figure 42: A connection graph topology created from a 39 bus power system. The lines that have been cut are marked in red.

evolutionary (DE) algorithm, i.e., εDEag, which shows competitive results in the CEC'10 constrained real-parameter optimization competition [98, 99]. Experimental results in later sections will show that our proposed algorithms compare favorably against the εDEag algorithm on large-scale constrained optimization problems.

In summary, the contributions of this report are listed as follows.

1. This report is, to the best of the authors' knowledge, the first published work on large-scale constrained optimization using nature-inspired algorithms (e.g., evolutionary algorithms and PSO) examined with a comprehensive benchmark problem set.
2. A hybridized optimization algorithm framework, i.e., εCCPSO, is proposed for large-scale constrained optimization, by combining the CCPSO2 algorithm and ε constrained method.
3. Three different algorithms based on the εCCPSO framework are proposed for large-scale constrained optimization, i.e., εCCPSOd with ε-criteria based improvement detection, εCCPSOw with fixed-size improvement detection window, and εCCPSOw2 with adaptive improvement detection window.
4. Eleven benchmark problems proposed for the CEC'10 special session on constrained real-parameter optimization (i.e., CEC'10CRPO) are extended from a maximum of 30D to a maximum of 1000D for our computer experiments.
5. A comprehensive study is carried out comparing the proposed algorithms with the state-of-the-art constrained optimization algorithm εDEag on the eleven large-scale benchmark problems, with an emphasis on εCCPSOw2 among the other three εCCPSO algorithms.

Many industrial applications are essentially constrained optimization problems, with the physical or environmental constraints formulated as equality or inequality relationships. For example, the balanced coordinated resource allocation design of a network system [100], the vulnerability analysis of power grids [94], and the optimization of cascading failure protection in complex networks [101] can all be formulated as constrained nonlinear optimization problems. Moreover, the above mentioned applications are all potentially large-scale problems, since a network can have a large number of nodes.

Generally, a constrained optimization problem can be expressed in the following formulation, i.e.,

$$\begin{aligned} \text{minimize:} \quad & f(\mathbf{x}), \\ \text{subject to:} \quad & g_i(\mathbf{x}) \le 0, \ i = 1, \dots M_g \\ & h_j(\mathbf{x}) = 0, \ j = 1, \dots M_h \\ & x_k^{\min} \le x_k \le x_k^{\max}, \ k = 1, \dots, D, \end{aligned} \tag{111}$$

where $\mathbf{x} = (x_1, x_2, \dots, x_D) \in R^D$ is a D dimensional vector, and $f(\mathbf{x}) : R^D \to R$ is called an objective function. The problem has M_g inequality constraints $g_i(\mathbf{x}) \le 0$ and M_h equality constraints $h_j(\mathbf{x}) = 0$. Each element $x_k \in R$ of the vector $\mathbf{x}$ is also called a variable, and has a lower bound $x_k^{\min}$ and an upper bound $x_k^{\max}$. Usually the equality constraints are relaxed into inequality constraints of the form

$$\tilde{h}_j(\cdot) = |h_j(\cdot)| - h_\delta \le 0,$$

where the tolerance level h_δ specifies how much the equality constraints are relaxed.

To solve large-scale constrained optimization problems, our εCCPSO framework combines a cooperative coevolutionary PSO algorithm [88] with the ε constrained method [91]. The update law of a classical PSO algorithm is given as follows,

$$\begin{aligned} v_{i,d}(t+1) &= \omega v_{i,d}(t) + c_1 r_1^{(i)}(t)(p_{i,d}(t) - x_{i,d}(t)) + \\ &\quad c_2 r_2^{(i)}(t)(g_d(t) - x_{i,d}(t)), \\ x_{i,d}(t+1) &= x_{i,d}(t) + v_{i,d}(t+1), \end{aligned} \tag{112}$$

where $x_{i,d}$, $v_{i,d}$ and $p_{i,d}$ denotes the dth dimension of the position $\mathbf{x}_i$, velocity $\mathbf{v}_i$ and personal best $\mathbf{p}_i$ of particle i, respectively; and g_d denotes the dth dimension of the global best $\mathbf{g}$ of the swarm. The parameters ω, c_1 and c_2 are called inertia weight, cognitive attraction and social attraction, respectively. Besides, $r_1^{(i)}$ and $r_2^{(i)}$ are random variables independently and uniformly sampled from $[0, 1]$ for particle i in each iteration. In the terminology for many nature-inspired algorithms, an iteration t as in (112) is also called a generation.

The idea of cooperative coevolution is to partition the dimensions/variables space into certain groups and evolve the current group's variables using the best solutions found in the other groups [96,97]. To maximize the performance of CC algorithms, correlated variables should be partitioned into the same group, and uncorrelated variables into different groups. To this end, different grouping strategies have been proposed and investigated [85,86,88–90]. For example, in multilevel cooperative coevolution (MLCC) [90], in each generation a group size s is probabilistically chosen from a set S according to the recorded performance of each group size in S, then the variables are divided into $K = n/s$ groups. In CCPSO2 [88], this procedure is simplified by choosing a new group size s uniformly at random from S only if there is no improvement to the global best $\mathbf{g}$ in the last generation. Competitive results have shown the effectiveness of CCPSO2 for large-scale unconstrained continuous optimization. Instead of using the classical update law (112), Ref. [88] shows that a Cauchy and Gaussian-based update law with ring topology works better with the random grouping cooperative coevolutionary PSO. The update law is shown as follows,

$$x_{i,d}(t+1) = \begin{cases} p_{i,d}(t) + C_i(t)|p_{i,d}(t) - l_{i,d}(t)|, & \text{if rand} < \sigma, \\ l_{i,d}(t) + \mathcal{N}_i(t)|p_{i,d}(t) - l_{i,d}(t)|, & \text{otherwise}, \end{cases} \tag{113}$$

where C_i and $\mathcal{N}_i$ are random variables following the standard Cauchy distribution and the standard Gaussian distribution for particle i, independently sampled at each generation t. Besides, $l_{i,d}$ is the dth dimension of the neighborhood's best $\mathbf{l}_i$ of particle i. The probability of choosing Cauchy update is specified by a user-defined parameter $\sigma \in [0,1]$, and the probability of choosing the Gaussian update is $1-\sigma$.

There are many constraints handling methods for nature-inspired-algorithms, as introduced in [93]. Some of the common ones include penalty method, stochastic ranking, ε-constrained method, multi-objective approaches, ensemble of constraints-handling techniques, etc. In this report, the ε-constrained method [92] is chosen to be the constraints handling component of our algorithm, mainly because of its simplicity and competitive performance, as already demonstrated in many works [91,92,102,103]. Besides, in the CEC'10 competition & special session on constrained real-parameter optimization, an ε-constrained differential evolution algorithm with an archive and gradient-based mutation (eDEa) demonstrated highly competitive results [91], and was the winner of the competition.

In the ε-constrained method, for any $\varepsilon \geq 0$, the ε level comparison $<_\varepsilon$ between two pairs of particles' objective values and constraints violations (f_1, ϕ_1) and (f_2, ϕ_2) is defined as follows,

$$(f_1, \phi_1) <_\varepsilon (f_2, \phi_2) \iff \begin{cases} f_1 < f_2, \text{ if } \phi_1, \phi_2 \leq \varepsilon; \\ f_1 < f_2, \text{ if } \phi_1 = \phi_2; \\ \phi_1 < \phi_2, \text{ otherwise}, \end{cases}$$

where the constraints violation can be calculated in various ways. Here we use the following constraint violation function

$$\phi(\mathbf{x}) = \begin{cases} \sum_{i=1}^{M_g} (\max\{0, g_i(\mathbf{x})\})^2 + \sum_{j=1}^{M_h} (\max\{0, \tilde{h}_j(\mathbf{x})\})^2, \\ \quad \text{if } x_k \in [x_k^{\min}, x_k^{\max}],\ k = 1, \ldots, D, \\ \infty, \quad \text{otherwise}, \end{cases}$$

where the inequality constraint $\tilde{h}_j(\cdot) = |h_j(\cdot)| - h_\delta \leq 0$ is a relaxation for the equality constraint $h_j(\cdot)$ with tolerance h_δ.

The definition of the ε level comparison $<_\varepsilon$ allows transforming a constrained optimization problem into an unconstrained problem. In case of $\varepsilon = \infty$, $<_\varepsilon$ is equivalent to the comparison $<$ of objective function values. In case of $\varepsilon = 0$, $<_\varepsilon$ is equivalent to the lexicographic order comparison where constraint violation precedes objective function values. Furthermore, it can be proved that for a certain ε value, the order of the particles is well-defined, i.e., if $(f_1, \phi_1) <_\varepsilon (f_2, \phi_2)$ and $(f_2, \phi_2) <_\varepsilon (f_3, \phi_3)$, then $(f_1, \phi_1) <_\varepsilon (f_3, \phi_3)$. More properties of the ε level comparison can be found in [91,92].

Next, we introduce the εCCPSO framework and the novel ε level control method in this framework. Furthermore, for different new group size selection methods, three different algorithms based on the εCCPSO framework are proposed. As in CCPSO2, the dimensions of the solutions are divided into K groups, with each having size s ($d = Ks$). Here, the jth group part of the dimensions of a vector $\mathbf{z}$ is denoted as $\mathbf{z}.\mathbf{d}_j$, $j \in [1 \ldots K]$, and $\mathbf{b}(j, \mathbf{z})$ returns the vector $(\mathbf{g}.\mathbf{d}_1, \mathbf{g}.\mathbf{d}_2, \ldots, \mathbf{g}.\mathbf{d}_{j-1}, \mathbf{z}, \mathbf{g}.\mathbf{d}_{j+1}, \ldots, \mathbf{g}.\mathbf{d}_K)$. In εCCPSO, the comparisons between objective function values in CCPSO2 are replaced by the ε-comparisons between the pairs of objective function values and constraints violations. Furthermore, using ε-comparison, the notion of ε-minimizations, i.e., $\min_\varepsilon$ and $\arg\min_\varepsilon$, can be introduced. For example, $\min_\varepsilon[f(\mathbf{z}_i), \phi(\mathbf{z}_i)]$ denotes the minimal pair $[f(\mathbf{z}_i^*), \phi(\mathbf{z}_i^*)]$ by the criteria of ε-comparison. Let $\mathcal{N}(i)$ denote the neighborhood particles of particle i. Since we use a ring topology, we have $\mathcal{N}(1) = \{n, 1, 2\}$, $\mathcal{N}(n) = \{n-1, n, 1\}$ and $\mathcal{N}(i) = \{i-1, i, i+1\}$ for $i = 2, \ldots n-1$. Furthermore, for the clarity of representation, let us define $\mathcal{P}(\mathbf{x}) = [f(\mathbf{x}), \phi(\mathbf{x})]$ as the pair of objective value and constraints violation corresponding to solution $\mathbf{x}$. Thus,

$$i^* = \underset{i' \in \mathcal{N}(i)}{\arg\min_\varepsilon} [f(\mathbf{z}_{i'}), \phi(\mathbf{z}_{i'})] = \underset{i' \in \mathcal{N}(i)}{\arg\min_\varepsilon} \mathcal{P}(\mathbf{z}_{i'})$$

denotes the neighborhood best of particle $\mathbf{z}_i$ by means of ε-comparison.

In an algorithm utilizing the ε constrained method, the ε level needs to be controlled such that it gradually reduces from a large value (corresponding to large tolerance of constraints violations) to zero (corresponding to zero tolerance of constraints violations). Here, based on the ε level control method used in [91], we adopt a modified version for large-scale constrained optimization, which is shown as follows,

$$\begin{aligned}
\varepsilon(0) &= \frac{1}{2}\left(\frac{1}{N}\sum_{i=1}^{N}\phi(x_i(0)) + \min_{i=1,\ldots,N}\phi(x_i(0))\right) \\
\varepsilon(\tilde{t}) &= \begin{cases} \varepsilon(0)\left(1 - \frac{\tilde{t}}{T_c}\right)^{c_p(\tilde{t})}, & 0 < \tilde{t} \le T_c, \\ 0, & \tilde{t} > T_c, \end{cases} \\
c_p(\tilde{t}) &= \begin{cases} \max\left\{c_p^{\min}, \frac{\log \varepsilon_\lambda - \log \varepsilon(0)}{\log(1 - T_\lambda / T_c)}\right\}, & 0 \le \tilde{t} \le T_\lambda, \\ c_p^{\min} - k_{c_p}(T_c - \tilde{t}), & \tilde{t} > T_\lambda, \end{cases} \\
k_{c_p} &= \frac{c_p^{\min} - c_p(T_\lambda)}{T_c - T_\lambda}.
\end{aligned}$$

where $\tilde{t}$ is the current fitness evaluations, i.e., FES, and the initial ε level $\varepsilon(0)$ is chosen to be the middle of the best value and average value of initial constraints violations of the particles. Besides, $T_c = 0.95$Max_FES is the fitness evaluations when ε is decreased to zero, i.e., $\varepsilon(T_c) = 0$, where Max_FES is the maximum number of fitness evaluations, and T_λ is the fitness evaluations when the ε level is decreased to ε_λ, i.e., $\varepsilon_\lambda = \varepsilon(T_\lambda)$. The control parameter c_p is set fixed for $0 \le \tilde{t} \le T_\lambda$, and is gradually reduced to its specified minimum value $c_p^{\min}$ when $\tilde{t} > T_\lambda$, with $c_p(T_c) = 0$.

In unconstrained CCPSO2, a new group size s is selected from S if the global best has not been improved in the last generation. Note that since in unconstrained optimization, the global best can only be changed to improve the objective function, we can consider any update of the global best as an improvement. However, when combining CCPSO2 with the ε constrained method, a problem of detecting the improvements arises. In εCCPSO for constrained optimization, new global best solution can also be chosen for the improvement of constraints violations.

Now we consider several different ways of determining whether the global best has been improved or not in the last generation. First, let εCCPSOd be the εCCPSO algorithm that detects the improvement of global best using ε-comparison, i.e., the global best is considered improved if $\mathcal{P}(\mathbf{g}_{\text{new}}) <_\varepsilon \mathcal{P}(\mathbf{g}_{\text{old}})$, where $\mathbf{g}_{\text{new}}$ denotes the new global best and $\mathbf{g}_{\text{old}}$ denotes the old global best. This is reasonable based on the new ε-criteria. Thus, we know that any update of the global best is considered as an improvement in εCCPSOd. This is directly derived from and is similar to the original CCPSO2 algorithm.

Secondly, let εCCPSOw be the εCCPSO algorithm that detects the improvement of the global best using the ordinal comparison of objective function values (i.e., fitness values) with an improvement detection window of a specified size. In εCCPSOw, the new global best is considered to have been improved if the new global best objective value is smaller than the minimum of the global best fitness values in the last w generations, i.e., $f(\mathbf{g}_{\text{new}}) < \min\{f(\mathbf{g}) : \mathbf{g} \in S_g(w)\}$, where S_g denotes the set of global best in the last w generations, and w is called the size of the improvement detection window. The rationality behind the design of εCCPSOw is that in a constrained optimization, both the global best fitness value and constraints violation should be reduced in order for an update of global best to be called an improvement. However, an update based on ε-comparison with a new ε level does not necessarily reduce the fitness value, thus it is reasonable to compare the current global best fitness value with a certain number of previous global best fitness values to decide whether the update is an improvement. Note that different improvement detection window size has different effect. For some problems, εCCPSOw with $w = 1$ yields better performance than using $w = 10$, while sometimes using $w = 10$ yields better performance.

To automatically select an appropriate improvement detection window size, finally we propose an εCCPSO method which uses an adaptive detection window, i.e., εCCPSOw2. We adopt the adaptive weighting scheme used in [90], where a performance record list $\mathbf{R} = \{r_1, r_2\}$ is used and is updated according to

$$r_i = \frac{f(\mathbf{g}_{\text{old}}) - f(\mathbf{g}_{\text{new}})}{f(\mathbf{g}_{\text{old}})}.$$

The probability p_1 and p_2 of selecting window size 1 and 10 respectively in each generation is computed as

$$p_i = \frac{e^{7r_i}}{e^{7r_1} + e^{7r_2}},$$

where constant 7 and the natural exponential constant e are empirical values.

In this part, we show the experimental results of the proposed three different algorithms, i.e., εCCPSOd, εCCPSOw and εCCPSOw2, compared with the state-of-the-art constrained optimization evolutionary algorithm εDEag. First, the eleven benchmark problems and the parameter settings for our algorithms are described. Second, the results of the εCCPSOw algorithm with $w = 1$ and $w = 10$ are compared to εCCPSOw2 with adaptive improvement detection technique. Then, εCCPSOw2 and εCCPSOd are compared to εDEag on the benchmarks of 100D. Finally, we compare the results of εCCPSOw2 and εCCPSOd on the benchmarks of higher order dimensions, i.e., 500/1000D, and show that εCCPSOw2 is more favorable for a general application. All results here are the average of 25 runs of the corresponding algorithms.

Table 11: Parameter settings

Name	Value
N	30
D	100/500/1000
σ	0.5
Max_FES	20000*D
S	{2,5,10,50,100} for D<250; {2,5,10,50,100,250} otherwise
h_δ	0.001
ε_λ	0.34
$c_{\text{p}}^{\text{min}}$	3
T_{c}	0.95 Max_FES
T_λ	0.8 Max_FES

The parameter settings for the different εCCPSO algorithms (i.e., εCCPSOd, εCCPSOw and εCCPSOw2) are shown in Table 11.

We adopt and extend eleven benchmark problems proposed for the CEC'10 special session on constrained real-parameter optimization (i.e., CEC'10CRPO) [99], i.e., C01, C02, C03, C04, C05, C09, C12, C14, C16, C17 and C18. The properties of these problems are shown in Table 12, where S is for "separable" and N for "Non-separable". For example, C04 has a separable objective function, 2 non-separable and 2 separable equality constraints, and no inequality constraint. Detailed problem definitions can be found in [99].

Roughly speaking, these problems are defined in the formulation of (111). Moreover, all the variables of a problem have the same lower bounds and upper bounds, i.e., $x_k^{\text{min}} = x^{\text{min}}$ and $x_k^{\text{max}} = x^{\text{max}}$. Besides, a random translation vector $o \in [o^{\text{min}}, o^{\text{max}}]^D$ is used, where o^{min} and o^{max} are dependent on the problem. We extend the original CEC'10CRPO problems from a maximum of 30D to a maximum of 1000D. For example, in C02, we have $x^{\text{min}} = -5.12$ and $x^{\text{max}} = 5.12$. From the code of the original C02 problem, $\min(o) = -0.4966$ and $\max(o) = 0.4934$, thus we let $o^{\text{min}} = -0.5$ and $o^{\text{max}} = 0.5$, and generate o uniformly at random in the region $[o^{\text{min}}, o^{\text{max}}]^{1000}$. After the translation vectors o for the problems are generated, they are fixed for all experiments.

There are cases in which different improvement detection window sizes make huge differences to the performance of the εCCPSOw algorithm. As shown in Fig. 43, when solving problem C02 with εCCPSOw, using $w = 1$ results in better performance than using $w = 10$. However, when solving problem C05, $w = 10$ yields better performance

Table 12: Properties of the eleven CEC'10 problems

Problem	Search range	Objective	Number of constraints	
			Equalities	Inequalities
C01	$[0, 10]^D$	N	0	2 N
C02	$[-5.12, 5.12]^D$	S	1 S	2 S
C03	$[-1000, 1000]^D$	N	1N	0
C04	$[-50, 50]^D$	S	2 N 2 S	0
C05	$[-600, 600]^D$	S	2 S	0
C09	$[-500, 500]^D$	N	1 S	0
C12	$[-1000, 1000]^D$	S	1 N	1 S
C14	$[-1000, 1000]^D$	N	0	3 S
C16	$[-10, 10]^D$	N	2 S	1 S 1 N
C17	$[-10, 10]^D$	N	1 S	2 N
C18	$[-50, 50]^D$	N	1 S	1 S

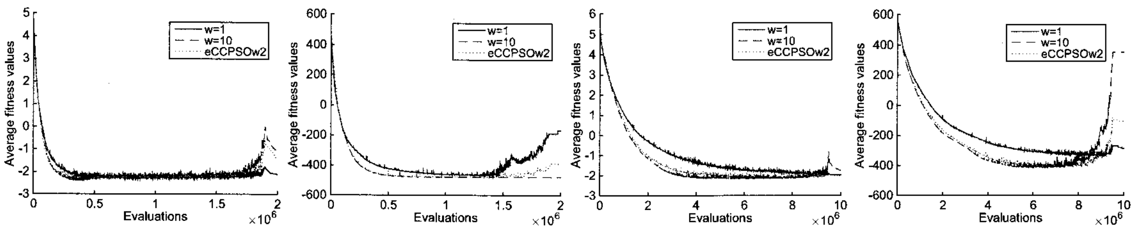
Table 13: εDEag experiments at different scales

C04	70D	0.789630467	0.402137082
	90D	2.970161086	2013.070712
	97D	5.536830321	11457.7614
	100D	Unsolvable	
C12	35D	-0.215615805	0
	40D	-82.53091592	5.9706E+143
	50D	Unsolvable	
	100D	Unsolvable	

(note that in ε constrained optimization, the global best fitness values are not always decreasing, since the particles are compared using new ε-criteria continually). Because of the adaptive improvement detection window, for both C02 and C05, the performance of εCCPSOw2 is in between εCCPSOw with $w = 1$ and with $w = 10$. For other problems which are not shown in the figure, it is either that using $w = 1$ and $w = 10$ in εCCPSOw do not make much difference, or that the average fitness value of εCCPSOw2 almost coincide to one of the two cases of εCCPSOw (i.e., $w = 1$ or $w = 10$).

Thus, the εCCPSOw2 algorithm can roughly represent the class of εCCPSO algorithms with improvement detection window technique in terms of the average fitness value performance. For the rest of the experimental studies, we thus use the εCCPSOw2 algorithm to compare with other types of algorithms on large-scale constrained optimization. Although there is no nature-inspired algorithm known to be capable of solving general large-scale real-valued constrained optimization problems, constrained optimization algorithms of smaller scales are abundant, and many of them are well studied [93]. The εDEag algorithm is an ε constrained differential evolutionary algorithm which makes use of an archive and gradient-based mutation. It has been shown to be very efficient for 10/30D real-valued constrained optimization [91]. The code of the algorithm is downloadable from [104], which is also used in this research.

However, as the number of dimensions increases, the efficiency decreases, as shown in Table 13 for problems C04 and C12. Furthermore, when the number of dimensions is greater than a certain limit (i.e., 97D for C04 and 40D for C12), overflows occur during the execution of εDEag for these two problems. In Table 14, we compare the average fitness values (Avg. Fitness) and the maximum constraints violations (Max. Const.) of the final global best solution of the 25 runs of the εCCPSOw2 and εCCPSOd algorithms with those of the solutions of the εDEag algorithm. As shown in the table, both εCCPSOw2 and εCCPSOd surpass εDEag for all problems except C01, where εCCPSOw2 and εCCPSOd have not found a final feasible solution. Note that the global best of εCCPSOw2 and εCCPSOd have better fitness values than that of εDEag for all problems; and except for C01, the maximum constraints violations of



(a) C02 of 100D (b) C05 of 100D (c) C02 of 500D (d) C05 of 500D

Figure 43: Fitness values of εCCPSOw with $w = 1$ and $w = 10$ compared to εCCPSOw2 solving two benchmark problems of 100/500 dimensions

Table 14: 100D experiments

	εCCPSOw2		εCCPSOd		εDEag	
	Avg. Fitness	Max. Const.	Avg. Fitness	Max. Const.	Avg. Fitness	Max. Const.
C01	-3.7166E+01	5.6250E-01	-3.7166E+01	5.6250E-01	-4.6248E-01	0.0000E+00
C02	-1.4476E+00	0.0000E+00	-2.1663E+00	0.0000E+00	1.5347E-02	0.0000E+00
C03	1.8205E+02	4.4568E-02	4.0366E+04	9.7032E+00	1.2688E+08	1.0447E+05
C04	2.8649E+00	3.2839E+00	2.1011E+00	3.5152E+00	Unsolvable	
C05	-3.9359E+02	0.0000E+00	-2.9817E+02	1.1518E-02	1.0681E+01	2.6775E+02
C09	3.0431E+03	0.0000E+00	2.3324E+05	0.0000E+00	7.6256E+07	0.0000E+00
C12	-7.6703E+02	4.7877E+04	-7.9284E+02	1.0397E+03	Unsolvable	
C14	1.0349E+04	0.0000E+00	1.9022E+11	0.0000E+00	7.0077E+12	0.0000E+00
C16	1.4223E-03	0.0000E+00	1.4212E-02	0.0000E+00	1.0791E+00	0.0000E+00
C17	8.6879E+01	0.0000E+00	5.4661E+01	0.0000E+00	4.6463E+02	0.0000E+00
C18	4.7323E+00	0.0000E+00	1.7498E+01	0.0000E+00	7.7350E+03	0.0000E+00

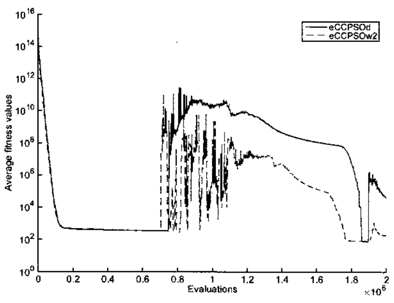
both εCCPSOw2 and εCCPSOd are all smaller than that of εDEag.

The final average global best fitness values (Avg. Fitness) and maximum constraints violation (Max. Const.) of εCCPSOw2 and εCCPSOd applied on the eleven benchmark problems of 100/500/1000D are shown in Table 14, 15 and 16, respectively. Besides, the evolution process of the average fitness values of εCCPSOw2 and εCCPSOd solving six of the benchmarks (i.e., C03, C04, C09, C14, C16 and C18) of 100/500D are shown in Fig. 44 and Fig. 45. From the results we can see that for five of the benchmarks, i.e., C01, C02, C04, C12, C16 and C17, in the cases of 100/500D, the performance of the two algorithms are too close to tell which is better. However, for the set of problems that εCCPSOw2 outperforms εCCPSOd, i.e., C03, C05, C09, C14 and C18, the performance gaps are clear. We would thus conclude that although εCCPSOd is better in some situations, εCCPSOw2 is more favorable for general optimization problems.

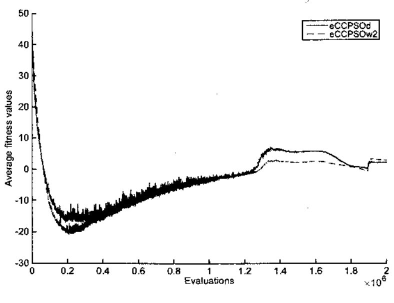
5.2 B-spline Membership Function Based FCM to Maximize Overlap Areas for Interconnected Power Systems

State estimation of multi-area power systems is a challenging problem due to its large size. Traditionally it was performed in regional control centers with limited interaction. Firstly we should divide the whole power system in several subsystems whose control center can perform its own state estimation. Dividing systems is a crucial and essential step to estimate state of power systems. The quality of the divided system can significantly affect the state estimation we will finally obtain.

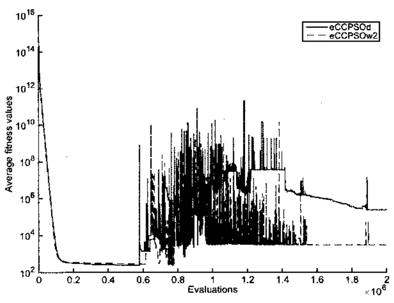
Some of the division methods have been proposed to address this problem. Fuzzy C-means (FCM) is one of them and its success is primarily due to the introduction of fuzziness index [105–107, 107]. By doing so we can easily figure out the overlap areas between the subsystems. And, in order to enhance the communication between different areas, we should classify more areas to be the overlap areas which can communicate with the subsystems they belong to.



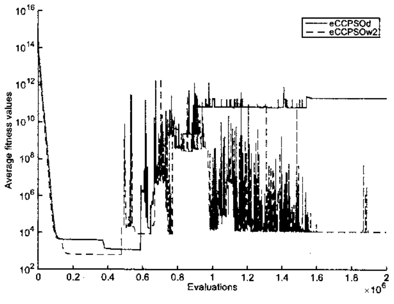
(a) C03



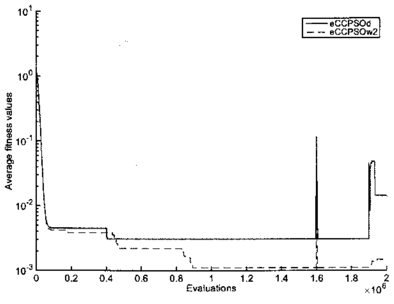
(b) C04



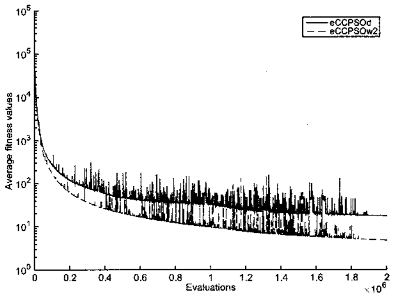
(c) C09



(d) C14



(e) C16



(f) C18

Figure 44: Fitness values of εCCPSOd and εCCPSOw2 solving six benchmark problems of 100 dimensions

By introducing a bridgeness term into the objective function, a novel FCM algorithm can be proposed to maximize the overlap areas in the power system. This algorithm will make the membership degree of one bus more fuzziness. However, we should consider to incorporate a regularized term to avoid classifying the whole power system into only one area.

Even though this modified FCM can acquire more overlap areas between neighborhood subsystems, one of its disadvantages is that it needs to take more iteration steps. However, we notice that when the Euclidean distance between a bus and a center of area is small, the membership degree that bus belongs to this cluster is high and vice versa. Hence we try to use B-spline membership functions instead of the fuzzy membership functions to represent this relationship. By doing so we can accelerate the convergence speed of our proposed algorithm.

The standard FCM algorithm is an unsupervised cluster algorithm which based on Picard iterations. And it assigns buses to each areas by using their fuzzy membership. Let us denote the power system with N buses which should be partitioned into c areas by $X = \{x_i, i = 1, 2, \ldots, N | x_i \in R^d\}$ and the membership degree of bus x_i to areas $j \in \{1, \ldots, c\}$ by $u_{ji} \in [0, 1]$. Also, we can denote the set of area prototypes by $v_j \in \{v_1, v_2, \ldots, v_c\}$. FCM algorithm is formulated as the minimization of the objective function J with respect to the membership matrix $U = u_{ji}$:

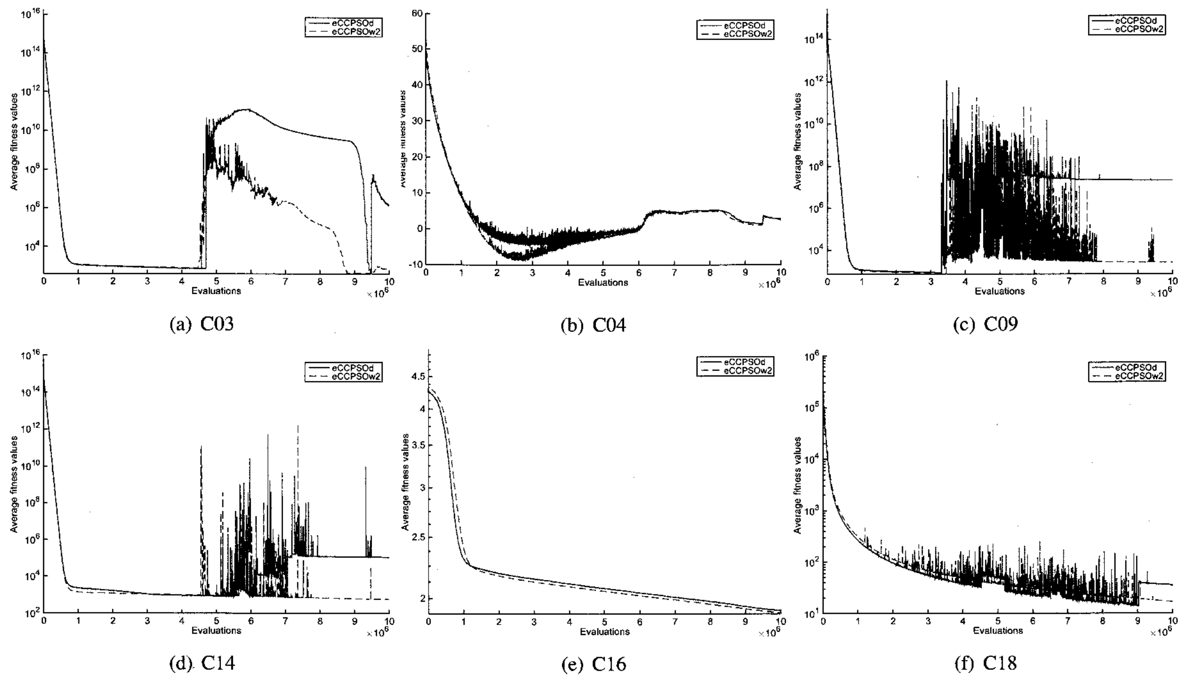
$$J = \sum_{j=1}^{c} \sum_{i=1}^{N} u_{ji}^{m} \|x_i - v_j\|^2 \tag{114}$$

15 ?

with the following constraints:

$$\sum_{j=1}^{c} u_{ji} = 1, \quad \forall i; \qquad 0 \le u_{ji} \le 1, \quad \forall j, i; \\ \sum_{i=1}^{N} u_{ji} > 0, \quad \forall j \tag{115}$$

where m is the fuzzy index which is chosen in advance and can influence the fuzziness of the final partition. Generally, we choose m to be 2. In each iteration step, minimization with respect to u_{ji} and v_j is done separately. The fuzzy



(a) C03 (b) C04 (c) C09

(d) C14 (e) C16 (f) C18

Figure 45: Fitness values of εCCPSOd and εCCPSOw2 solving six benchmark problems of 500 dimensions

membership update equation is:

$$u_{ji} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|x_i - v_k\|}{\|x_i - v_j\|} \right)^{\frac{1}{m-1}}} \tag{116}$$

and the area prototypes update equation is:

$$v_j = \frac{\sum_{i=1}^{N} u_{ji}^{m} x_i}{\sum_{i=1}^{N} u_{ji}^{m}} \tag{117}$$

Before implementation, the membership degree of each bus is initialized randomly. Then, using (116) and (117), the iteration process is running until obtaining the final result.

In order to maximize the communication between each area, we should assign more buses to the overlap areas or make one bus being shared among more areas. This can be measured by the concept of bridgeness of the bus. We define the bridgeness of a bus as the distance of its membership vector $u_i = [u_{1i}, u_{2i}, \ldots, u_{ci}]$ from the reference vector $[\frac{1}{c}, \frac{1}{c}, \ldots, \frac{1}{c}]$ in the Euclidean vector norm. Therefore, if a bus belongs to only one area, then the bridgeness should be large, and if otherwise, then the bridgeness will become small. The bridgeness term is defined as follows:

$$b_i = \sum_{j=1}^{c} \|\frac{1}{c} - u_{ji}\|^2 \tag{118}$$

We notice that b_i attains its minimum when v_i belongs to all of the clusters exactly with the same membership degree. ? { Then, if a bus has the small b_i, its membership to different areas will have less difference then it will have more chance to be in the overlap area to strength the communication between each area.

From the definition of bridgeness, we can find out that the fuzziness index is 2. In order to contain fuzziness index m in the iteration process, the fuzziness index should be generalized. Hence a new definition of bridgeness is given by:

$$b_i = \sum_{j=1}^{c} u_{ji} \|\frac{1}{c} - u_{ji}^{m-1}\| \tag{119}$$

Table 15: 500D experiments

	εCCPSOw2		εCCPSOd	
	Avg. Fitness	Max. Const.	Avg. Fitness	Max. Const.
C01	-8.2194E-01	0.0000E+00	-8.3171E-01	0.0000E+00
C02	-2.0467E+00	0.0000E+00	-1.9519E+00	0.0000E+00
C03	7.6050E+02	6.0284E-01	1.3366E+06	4.5627E+02
C04	2.4587E+00	7.6006E+01	2.8211E+00	1.1994E+02
C05	-1.1141E+02	0.0000E+00	-2.5524E+02	0.0000E+00
C09	2.6721E+03	0.0000E+00	2.0820E+07	0.0000E+00
C12	-5.7557E+02	1.5743E+03	-7.3178E+02	1.5694E+03
C14	5.5099E+02	0.0000E+00	1.0691E+05	0.0000E+00
C16	1.8885E+00	0.0000E+00	1.9077E+00	0.0000E+00
C17	1.3603E+04	0.0000E+00	1.4905E+04	0.0000E+00
C18	1.6546E+01	0.0000E+00	3.4631E+01	0.0000E+00

Table 16: 1000D experiments

	εCCPSOw2		εCCPSOd	
	Avg. Fitness	Max. Const.	Avg. Fitness	Max. Const.
C01	-8.3683E-01	0.0000E+00	-8.2553E-01	0.0000E+00
C02	-1.9279E+00	0.0000E+00	-1.3352E+00	0.0000E+00
C03	4.6553E+03	1.8839E+00	3.5834E+05	1.3029E+02
C04	3.4268E+00	9.0832E+01	4.6200E+00	2.0689E+02
C05	-1.8222E+02	0.0000E+00	-2.5260E+02	0.0000E+00
C09	1.2166E+03	0.0000E+00	6.4330E+09	0.0000E+00
C12	-3.2676E+02	2.2523E+03	-5.6378E+02	2.5163E+03
C14	9.2093E+02	0.0000E+00	8.4167E+02	0.0000E+00
C16	2.8188E+00	0.0000E+00	2.8280E+00	0.0000E+00
C17	3.1576E+04	0.0000E+00	3.3193E+04	0.0000E+00
C18	2.7503E+01	0.0000E+00	2.4988E+01	0.0000E+00

Consider the extreme value of b_i with constraint $\sum_{j=1}^{c} u_{ji} = 1$, where $0 \le u_{ji} \le 1$. According to the Lagrange multiplier procedure, we can obtain:

$$F = \sum_{j=1}^{c} u_{ji} \| \frac{1}{c} - u_{ji}^{m-1} \| - \lambda (\sum_{j=1}^{c} u_{ji} - 1) \tag{120}$$

where λ is the Lagrangian multiplier. By letting $(\partial F/\partial u_{1i}) = 0$, $(\partial F/\partial u_{2i}) = 0, \cdots, (\partial F/\partial u_{ci}) = 0$, we can have $u_{1i} = u_{2i} = \cdots = u_{ci} = \frac{1}{c}$, and $\lambda = \frac{1}{c} - mc^{1-m}$. At this moment, the value of bridgeness is $b_i = 0$. Because b_i is continuous in $[0, 1]^c$, b_i must have its maximum and minimum in the point $u_{1i} = u_{2i} = \cdots = u_{ci} = \frac{1}{c}$ or the boundary points. For the boundary points of $[0, 1]^c$, we can assume that there exist p zeros and q ones in the set $\{u_{1i}, u_{2i}, \ldots, u_{ci}\}$, where $0 \le p, q \le 1$. Thus if $p + q = c$, we can obtain $\sum_{j=1}^{c} u_{ji} \| \frac{1}{c} - u_{ji}^{m-1} \| = q(\frac{c-1}{c})$, which is greater than 0. And if $p + q < c$, without loss of generality, we can substitute the first $(p + q)$ elements from the b_i by $q(\frac{c-1}{c})$, then b_i can become the $\sum_{j=p+q+1}^{c} u_{ji} \| \frac{1}{c} - u_{ji}^{m-1} \| + q(\frac{c-1}{c})$. We can easily see $\sum_{j=p+q+1}^{c} u_{ji} \| \frac{1}{c} - u_{ji}^{m-1} \| + q(\frac{c-1}{c}) > 0$, which means the bridgeness will achieve its minimum when the membership vector of each bus equals $[\frac{1}{c}, \frac{1}{c}, \ldots, \frac{1}{c}]$. Hence this can explain that (118) and (119) have the same effect in the objective function [108].

However, if the bridgeness is too small, then it is possible that the whole buses will be classified as only one area.

In order to avoid this situation, we should define the between-cluster variation as follows:

$$V_b = \sum_{j=1}^{c} \sum_{i=1}^{N} u_{ji} \|v_j - \overline{x}\|^2 \tag{121}$$

where $\overline{x}$ is the mean of Euclidean norm of all the buses in the power system, and other symbols are similar to (114). In the process of cluster, we hope to minimize the bridgeness of buses and simultaneously maximize this between-cluster function. In order to incorporate this term into the objective function, we need to replace the u_{ji} with the u_{ji}^m in the between-cluster term, which is consistent with the standard FCM algorithm.

Now we introduce a new algorithm to cluster these buses based on FCM, the basic idea is that when clustering, we want to minimize the bridgeness term and maximize the between-cluster function at the same time. Henceforth the objective function is formulated as follows:

$$\begin{aligned} J_m =& \sum_{j=1}^{c} \sum_{i=1}^{N} u_{ji}^m \|x_i - v_j\|^2 + \sum_{j=1}^{c} \alpha \sum_{j=1}^{c} u_{ji} \|\frac{1}{c} - u_{ji}^{m-1}\| \\ & - \sum_{j=1}^{c} \eta \sum_{i=1}^{N} u_{ji}^m \|v_j - \overline{x}\|^2 \end{aligned} \tag{122}$$

where,

$$\begin{aligned} & \sum_{j=1}^{c} u_{ji} = 1, \quad \forall i; \qquad 0 \le u_{ji} \le 1, \quad \forall j, i; \\ & \sum_{i=1}^{N} u_{ji} > 0, \quad \forall j \end{aligned} \tag{123}$$

In order to acquire iteration equations, we use the Lagrange multiplier method, then the function J_m can be transformed into an unconstrained minimization problem as follows:

$$\begin{aligned} J_m =& \sum_{j=1}^{c} \sum_{i=1}^{N} u_{ji}^m \|x_i - v_j\|^2 + \sum_{j=1}^{c} \alpha \sum_{j=1}^{c} u_{ji} \|\frac{1}{c} - u_{ji}^{m-1}\| \\ & - \sum_{j=1}^{c} \eta \sum_{i=1}^{N} u_{ji}^m \|v_j - \overline{x}\|^2 + \sum_{i=1}^{N} \lambda_i (\sum_{j=1}^{c} u_{ji} - 1) \end{aligned} \tag{124}$$

By letting $\partial J_m / \partial u_{ji} = 0$ and $\partial J_m / \partial \lambda_i = 0$, we can obtain the following equations:

$$\begin{aligned} \frac{\partial J_m}{\partial u_{ji}} =& m u_{ji}^{m-1} \|x_i - v_j\|^2 + \alpha(1/c - m u_{ji}^{m-1}) \\ & - \eta m u_{ji}^{m-1} \|v_j - \overline{x}\|^2 + \lambda_i = 0 \end{aligned} \tag{125}$$

$$\frac{\partial J_m}{\partial \lambda_i} = \sum_{j=1}^{c} u_{ji} - 1 = 0 \tag{126}$$

Combining above two equations, we can obtain the following iteration equation:

$$u_{ji} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - v_j\|^2 - \eta \|v_j - \overline{x}\|^2 - \alpha}{\|x_i - v_k\|^2 - \eta \|v_k - \overline{x}\|^2 - \alpha} \right)^{1/(m-1)}} \tag{127}$$

Also, let $\partial J_m / \partial v_j = 0$, we have another iteration equation:

$$v_k = \frac{\sum_{i=1}^{N} u_{ji}^m (x_i - \eta \overline{x})}{\sum_{i=1}^{N} (1 - \eta) u_{ji}^m} \tag{128}$$

Algorithm 4 Maximize overlap FCM

```
Set the number of cluster $c$, the fuzziness index $m$, the parameters $\alpha$ and $\eta$, accuracy parameter $\epsilon$, the maximum number of iterations $t$, and the initialized fuzzy membership degree $u_{ji}$;
repeat
  Computing the cluster prototypes $v_j$ by (128);
  Computing the membership degree $u_{ji}$ by (127);
until $||J_m^{k+1} - J_m^k|| \leq \epsilon$ or reach the maximum number of iterations $t$.
```

5.3 B-Spline Membership Functions

Membership functions obtained by optimizing the objective function are restricted to particular shapes determined by (127). However, in some cases, people want to use their own membership function shapes that are better suitable to the practical problem or they may know the real distribution of the membership degree. Hence they can choose an approximate function to approach the real membership degree. In our cases, when the Euclidean distance between a bus and the prototype of the cluster is small, then the membership degree is large, otherwise it will be small. Consequently we try to represent this relationship by using the B-spline curve. Via B-spline curves we can translate the membership degree obtained by (127) into the form of control points of B-spline curves and construct B-spline membership functions, which can better represent the membership degree than the original one.

The B-spline curve can be constructed as follows: Given $n+1$ control points $\{p_0, p_1, \dots, p_n\}$, the i-th B-spline curve of order k is denoted by $N_{i,k}(t)$. Therefore, the B-spline curve $B(t)$ can be defined as:

$$B(t) = \sum_{i=0}^{n} p_i N_{i,k}(t) \qquad 1 \leq k \leq n \tag{129}$$

where

$$\begin{aligned} N_{i,k}(t) =& \Big(\frac{t - t_i}{t_{i+k-1} - t_i}\Big) N_{i,k-1}(t) \\ &+ \Big(\frac{t_{i+k} - t}{t_{i+k} - t_{i+1}}\Big) N_{i+1,k-1}(t) \\ & t_i \leq t < t_{i+k} \end{aligned}$$

$$N_{i,1}(t) = \begin{cases} 1 & \text{if } t_i \leq t \leq t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{130}$$

for $i = 0, 1, \dots, n$. This is a recursive definition. And using this formula, it is easy to construct the k-th order B-spline curve from two blending curves of order $k-1$. The set of knots $\{t_0, t_1, \dots\}$ is called the knot vector T. The above polynomial will be equal to zero when the denominator is zero. We choose B-spline because it is easy to evaluate when we know the knot vectors and control points. Also it can be implemented easily and is numerically well behaved [109, 110].

In order to represent the membership degree, we should choose knot vectors and control points for the B-spline curve first. The control points can be acquired from the approximate points which can be obtained from (127) and another key point is to specify knot sequence for the B-spline. These knots should satisfy the requirement $t_i \leq t_{i+1}$, which means the sequence is a non-decrease series of real numbers. We choose a uniform knot vector, in which the individual knot valued is evenly spaced. For the smoothness of the curve, we consider the order of the B-spline to be 3. Then the knot vector can be defined as:

$$t_i = \begin{cases} x_0 & \text{if } i < k \\ t_{i-1} + \frac{x_m - x_0}{n-k+2} & \text{if } k \leq i \leq n \\ x_m & \text{if } i > n \end{cases} \tag{131}$$

where x_i is the Euclidean distance between bus i and the prototype of the cluster. We can use the knot vector and the control points calculated according to approximate points by (127) to construct the B-spline Membership Function (BMF) to represent the membership degree.

Now we obtain a set of data from (127) which are treated as approximate points. We want to construct the membership function in term of B-spline curves. Hence the main task is to find out the set of control points for the B-spline curve to approach these approximate points. The error function should be defined as follows [111]:

$$e(x_j) = d(x_j) - u(x_j) = d(x_j) - \sum_{i=0}^{n} p_i N_{i,k}(x_j) \tag{132}$$

where $d(x_j)$ is the membership degree obtained from (127). The mean square error J can be defined as:

$$J = \sum_{j=0}^{n} e^2(x_j) = \sum_{j=0}^{m} \left[d(x_j) - \sum_{i=0}^{n} p_i N_{i,k}(x_j) \right]^2 \tag{133}$$

The error J function is optimized when the control points $\{p_i, i = 0, 1, \ldots, n\}$ can be found out to make the B-spline curve approximate $d(x_j)$. Therefore the partial derivative of J with respect to p_i should be zero, which means:

$$\frac{\partial J}{\partial p_i} = 2 \sum_{j=0}^{m} \left[d(x_j) - \sum_{i=0}^{n} p_i N_{i,k}(x_j) \right] [-N_{i,k}(x_j)] = 0 \tag{134}$$

which yields:

$$\sum_{j=0}^{m} d(x_j) N_{i,k}(x_j) = \sum_{j=0}^{m} \sum_{l=0}^{n} p_l N_{l,k}(x_j) N_{i,k}(x_j) \tag{135}$$

If we let:

$$Q_i = \sum_{j=0}^{m} d(x_j) N_{i,k}(x_j) \tag{136}$$

$$N_{il} = \sum_{j=0}^{m} N_{i,k}(x_j) N_{l,k}(x_j) \tag{137}$$

then we can substitute them into (135) and we can obtain:

$$Q_i = \sum_{i=0}^{n} p_l N_{il} \quad for\ i = 0, 1, 2, \ldots, n \tag{138}$$

We can rewrite it to a matrix form as follows:

$$\mathbf{Q} = \mathbf{NP} \tag{139}$$

where

$$\mathbf{Q} = [Q_0\ Q_1 \cdots Q_n]$$

$$\mathbf{N} = \begin{bmatrix} N_{00} & N_{01} & \ldots & N_{0n} \\ N_{10} & N_{11} & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ N_{n0} & N_{n1} & \ldots & N_{nn} \end{bmatrix}$$

$$\mathbf{P} = [p_0\ p_1 \ldots p_n] \tag{140}$$

Thus from the above equation, we can acquire the control points **P** which will lead the least squared error:

$$\mathbf{P} = \mathbf{N}^{-1}\mathbf{Q} \tag{141}$$

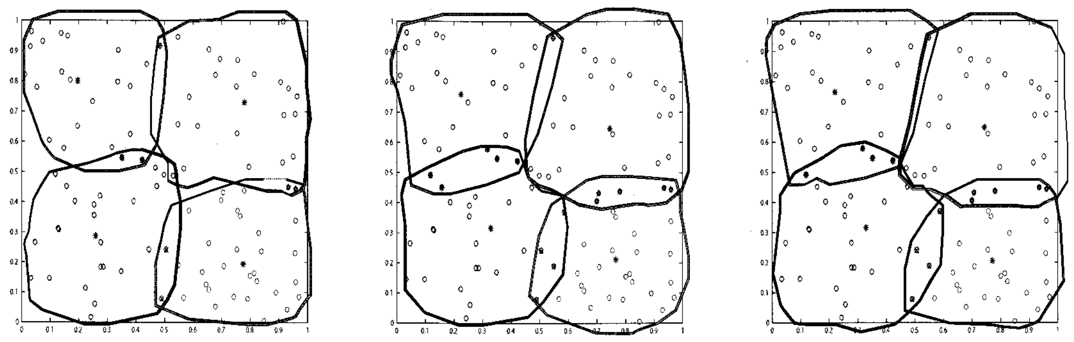
then, the B-spline membership function can be constructed as the following:

$$u(x) = \sum_{i=0}^{n} p_i N_{i,k}(x) \tag{142}$$

Now, the algorithm to evaluate the membership degree is modified as follows:

Algorithm 5 B-spline based maximize overlap FCM

```
Set the number of cluster $c$, the fuzziness index $m$, the parameters $\alpha$ and $\eta$, accuracy parameter $\epsilon$, the maximum number of iterations $t$, and the initialized fuzzy membership degree $u_{ji}$;
repeat
  Computing the control points by (141)
  Computing the BMF of every bus by (142);
  Computing the cluster prototypes $v_j$ by (128);
  Computing the approximate membership degree $u_{ji}by$ (127);
until $||J_m^{k+1} - J_m^k|| \le \epsilon$ or reach the maximum number of iterations $t$.
```



(a) standard FCM (b) maximum overlap FCM (c) B-spline based maximum overlap FCM

Figure 46: The cluster performance of three algorithms (Case 1)

5.4 Example

To verify our proposed B-spline based maximize overlap FCM algorithm, we have conducted two experiments to compare our algorithm with maximize overlap FCM algorithm and standard FCM algorithm. In Case 1, we can use an artificial data set to represent the buses in a power system. The experiment result shows that our proposed algorithm can maximize the overlap area when comparing with the standard FCM algorithm. Suppose the the data can be classified to four subsystems. We can set some parameters as follows: the fuzziness index $m = 2$, $0 < \alpha < 1$. In our experiment, we can set α to be 0.2 and η to be 0.01, accuracy parameter $\epsilon = 10^{-6}$ and the maximal iteration number $t = 200$. When $||J_m^{k+1} - J_m^k|| \le \epsilon$ or $t > 200$, the algorithm will terminate.

We set the defuzziness parameter to be 0.1, which means if the difference of two largest membership degrees is less than this number, then the bus will belong to both corresponding areas, otherwise, the bus will belong to the area with the largest membership degree. Figure 46 shows the result of the standard FCM algorithm, maximum overlap FCM and B-spline based maximum overlap FCM. We can see both maximum overlap FCM and B-spline based maximum overlap FCM have better performance than the standard FCM because they can generate more overlap buses after the cluster process. This will strengthen the communication between the subsystems of the power system.

The standard FCM will have fewer overlap areas when comparing with other two algorithms as shown in Figure 46. The last two algorithms will have the similar performance in which the maximum overlap FCM will generate 15 buses in the overlap areas and B-spline based maximum overlap FCM will generate 14 buses out of 100 buses. The standard FCM will only generate 7 buses in the overlap areas. However, the iteration number of B-spline based maximum overlap FCM is less than the maximum overlap FCM. And we can improve the convergence speed of our algorithm by using B-spline as membership degrees. According to Figure 47, the number of iteration steps of B-spline based maximum overlap FCM is 38, which is less than 84–the number of iteration steps of maximum overlap FCM.

Now in Case 2, we consider an IEEE 39-bus system in which there are 39 buses instead of 100 buses in Case 1. And other parameters are the same as in Case 1. Figure 48 shows the result of these three algorithms. From Figure 48, we can see the maximum overlap FCM generates 13 buses in the overlap areas, and B-spline based maximum overlap FCM generates 12 buses in the overlap areas. Both of them perform better than the standard FCM which generates only 4 buses in the overlap areas. Therefore maximum overlap FCM and B-spline based maximum overlap FCM will

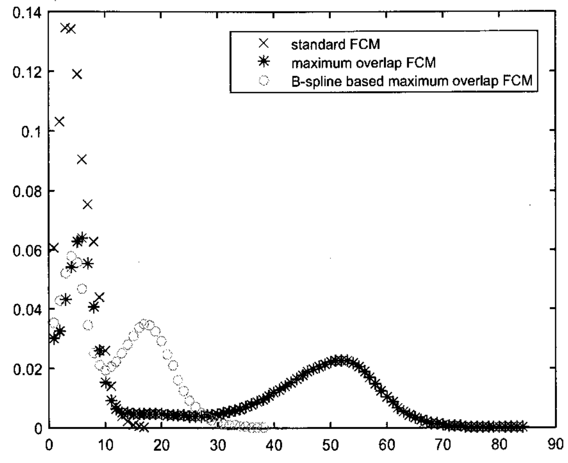
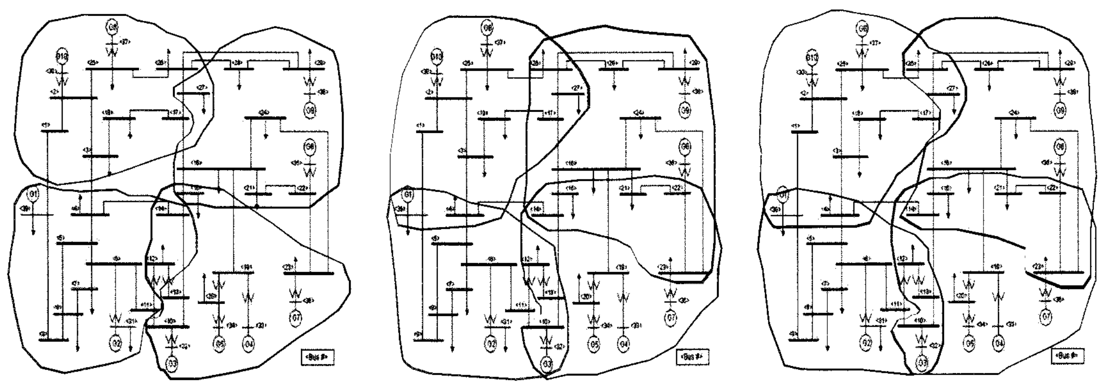


Figure 47: Iteration number of three algorithms (Case 1)



(a) standard FCM (b) maximum overlap FCM (c) B-spline based maximum overlap FCM

Figure 48: The cluster performance of three algorithms (Case 2)

have more strong connection between areas than standard FCM. Figure 49 shows the convergence speed of these three algorithms in Case 2. We can find that the number of iteration steps of B-spline based maximum overlap is 52, which is less than 70–the number of iteration steps of maximum overlap FCM. Also, from Figure 47 and Figure 49, we can see the magnitude of fluctuation of both maximum overlap FCM and B-spline based maximum overlap FCM are less than that of standard FCM.

5.5 Motion Planning for AmigoBot with Line Segment Based Map and Voronoi Diagram

The motion planning problem for robots refers to obtaining a collision-free path in an either known or unknown environment with some obstacles. Also, the path is often required to meet some optimal conditions, such as kinematic constraints and distance length restriction. Generally speaking, the problems related to motion planning in an unknown environment involve mapping, localization, collision avoidance, and trajectory tracking. Among them, mapping of the unknown environment with detective sensors is fundamental to the others [112]. A robot needs to reason geometrically about the unknown real-world environment from sensor readings and to represent it with the configuration space or *C-space* [113], only when the motion of the robot can be planned. Thus, mapping of the unknown environment is a crucial preliminary task for robots to automatically implement other assignments.

Two essentials should be considered in representation of the environment with a map: metric and topology. The metric framework is to represent obstacles in the two-dimensional or three-dimensional space with precise coordinates,

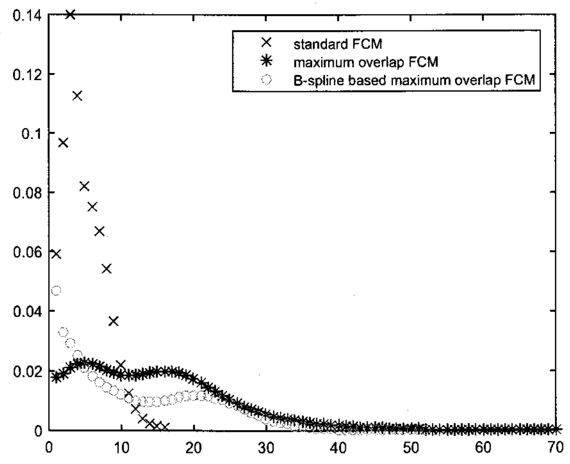


Figure 49: Iteration number of three algorithms (Case 2)

and the topological framework is to represent the relations between the obstacles. In other words, metric maps acquire geometrical features and properties of the environment, while the topological methods are focused on connectivity of points [114]. Thus, if a map with the accurate and complete information of metric and topology about the environment can be retrieved, then it is an easier task for a robot to plan a collision-free path. Unfortunately, the metric and topology cannot easily be integrated into a map, which will dramatically increase the computational effort in map building.

A compromised alternative approach is to build a map only with accurate metric information or with a simple topological information about the environment. Besides, the map can be retracted to a topological one which has the complete information about the environment and can be effectively utilized to plan the path for robots. In this report, a line segment based map is first constructed to represent the environment and then, a Voronoi diagram is retracted from the line segment based map and is applied to path planning of the AmigoBot. Compared to occupancy grid maps, the line segment based map is a good alternative for several reasons. First, it has low-level topological information about the environment, i.e., a set of point obstacles is represented with a line. Second, it is more concise and suitable for this method to represent large range environments and it is known that less storage memory is needed in this case. Moreover, line segments can be represented with much fewer points in the two-dimensional space and this method can greatly reduce the computational complexity when generating the Voronoi diagram due to its fewer points in the map.

As one of the most fundamental data structures in computational geometry, the Voronoi diagram has many applications for a wide range of problems inside and outside computer science, such as associative file searching, cluster analysis, scheduling record access, and collision detection [115]. The Voronoi diagram is also one of the earliest methods applied to the motion planning problem for robots. It is an excellent retraction method to build a map with useful topological information about the environment for motion planning. According to the definition of the Voronoi diagram, given some number of point or line obstacles, their Voronoi diagram divides the space based on the nearest-neighbor rule. As a result, the points in the edges of the Voronoi diagram are with the maximizing clearance between points and obstacles. Thus, the motion planning problem here is to find a path along the edges of the Voronoi diagram with a desired clearance from the initial configuration to the goal.

Many related works have been done for motion planning with the Voronoi diagram. The earliest work is done in [116], where a collision-free path for a disc is planned with a generalized Voronoi diagram in an environment with polygonal obstacles. Later, Fortune proposed a sweep line technique to compute the Voronoi diagram in [117]. This sweep line algorithm is widely used even nowadays for its simplicity and thus has many derivatives and variants [118, 119]. On the other hand, an algorithm based on the Voronoi diagram to compute an optimal path between a source and destination in the presence of simple disjoint polygonal obstacles is addressed in [120]. Meanwhile, Sakahara proposed a Voronoi-based StRRT (Spatiotemporal RRT) algorithm subjected to biasing extraction of sample points toward the border of a generalized Voronoi diagram to plan safe trajectories in [121]. In [122], an approximation method based on Voronoi duals to compute the shortest path in undirected graphs is presented.

To plan the motion with the Voronoi diagram, the robot needs to construct a more concise map to reduce the computational complexity. Compared to polygon road maps and grid-based maps, the line segment based map is a compromised alternative approach to represent the unknown environment, which has lower time complexity and needs less storage in construction. The line segment based map has sprung up a lot of research work in recent decade [123–125]. For instance, some line extraction algorithms on 2D laser scans are presented and experimented in [126, 127]. Also, a Split-and-Merge Fuzzy line extractor is proposed in [128] to construct a line segment based map for robots.

In this report, a line segment based map is constructed from the readings of sonar sensors installed on an AmigoBot. Different from the existing line segment extracting algorithms mentioned above which construct a line from a set of readings detected with laser sensors, in our proposed algorithm the lines have to be constructed incrementally with scatter obstacles detected from eight sonar sensors. Three key algorithms, $Generate - Line$, $Line - Fitting$, and $Merge-Line$, are proposed respectively to construct a concise line segment based map. Algorithm $Clear-Intersection$ is also designed to eliminate the intersecting lines. Then, a Voronoi diagram is generated with Fortune's sweep line algorithm. Next, Dijkstra's algorithm is applied in searching for a shortest path along the edges of the Voronoi diagram. Terminology *Clearance* is defined and calculated for the planned path to keep a safe distance with obstacles. Finally, a tracking control method is designed with $Line - of - Sight$ (LOS) approach to track the reference path. Simulation and experiment results are given to illustrate the efficacy of the proposed motion planning method. This research is part of the larger problem of threat detection and localization using multiple autonomous sensors [129] and can be used to construct a feasible path for mobile autonomous robots to detect and localize some contaminants or radioactive signatures in a geographic area.

An AmigoBot has two different driving wheels and a balance caster as shown in Fig. 50. Each drive axle is attached with a high-resolution optical quadrature shaft encoder for position and speed sensing and also dead-reckoning. The distribution of sonar sensors mounted on an AmigoBot is shown in Fig. 51. The sonar sensors have a sensitivity range from 10 cm to more than 3 meters and can provide a 360-degree sensing coverage [130]. Research on mapping

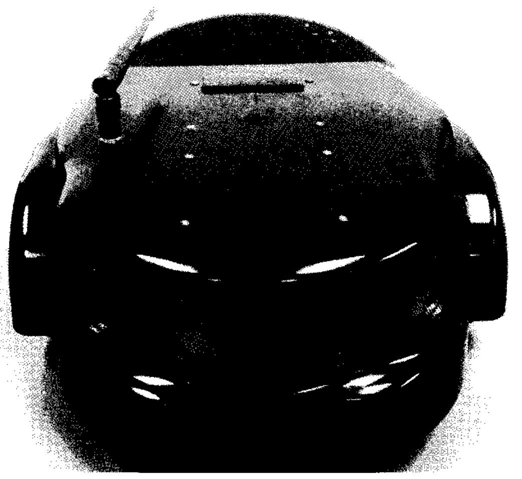


Figure 50: AmigoBot

and localization problems of robots in indoor environments have some important application values for service robots. With only sonar sensors, an AmigoBot is capable of detecting an unknown environment and building up a map with the dead-reckoning approach. Dead-reckoning is a simple but unreliable approach for long-term missions due to the time-increasing drift of estimates of robot location. For an AmigoBot, the locations estimated with the inertial navigation system are not accurate enough when the robot is turning or accelerating, which results in the shift of the positions of obstacles. Therefore, we envision that a two-step process is needed to solve this issue. The work done in this report will be the first step by building up a line segment based map to represent the unknown world and plan the path to navigate an AmigoBot. The next future work will emphasize on the localization of robots which is so called the simultaneous localization and mapping (SLAM) problem.

?

The mechanism of the motion planning system designed in this report is shown in Fig. 52. With perception from sonar sensors, the real-world environment is modeled with a line segment based map. Topological information of the environment is retracted from the line segment based map with a Voronoi diagram and then a shortest reference path

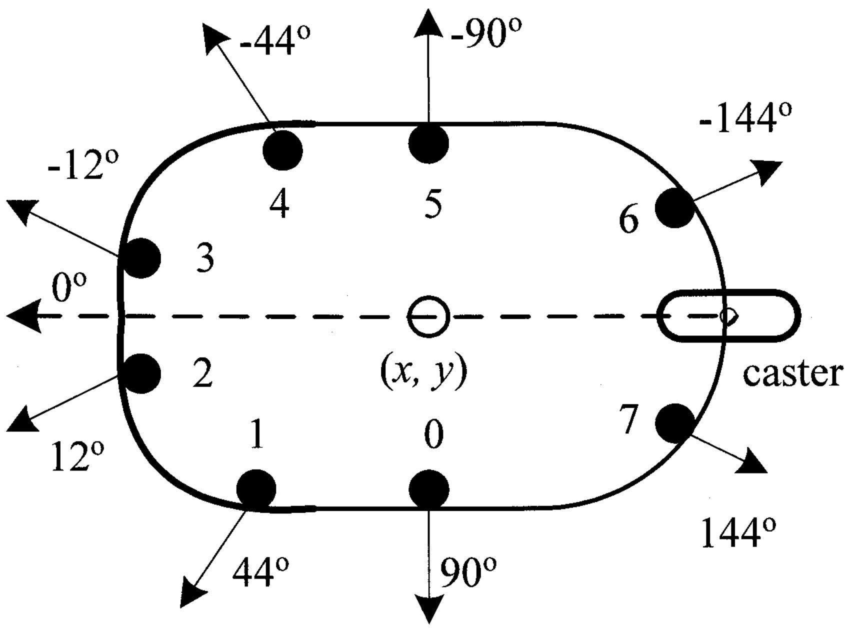


Figure 51: Sonar Distribution

?

with safe clearance is searched from it. With a motion control method, an AmigoBot is to track the reference path in the real-world environment from the initial configuration to the final destination. In the following discussion, the

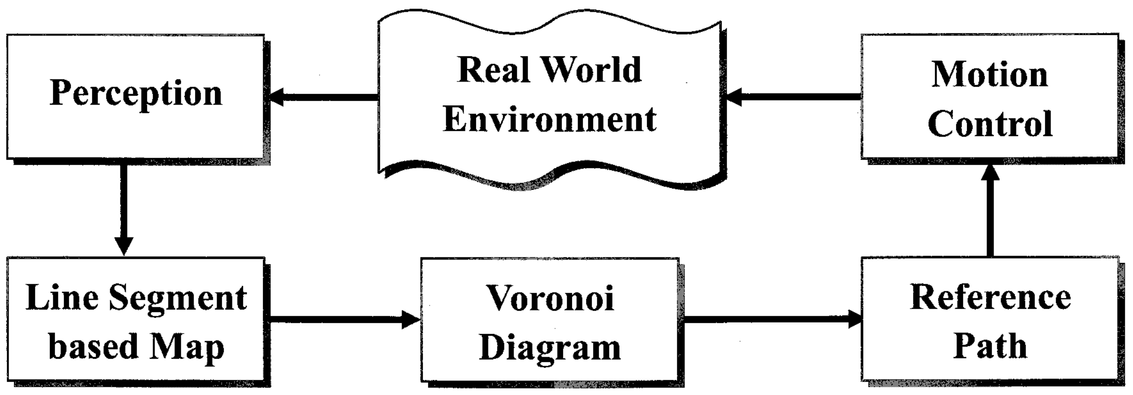


Figure 52: Flow Chart of Motion Planning System

distance between two points $p_i \in R^2$ and $p_j \in R^2$ is defined as the Euclidean distance $d(p_i, p_j) = \|p_i - p_j\|$. The distance between a point p_i and a line segment ℓ is defined as $d(p_i, \ell) = \min_{p_j \in \ell} \|p_i - p_j\|$. Also, the perpendicular distance of a point p_i to a line segment ℓ is defined with $d_P(p_i, \ell) = \min_{p_j \in \ell \cup \ell^+} \|p_i - p_j\|$, where ℓ^+ is the extension of ℓ.

The distance between two line segments ℓ_i and ℓ_j is defined with Hausdorff distance [131]. Hausdorff distance is the maximum distance of a set to the nearest point in the other set [131]. Hausdorff distance from line segment ℓ_i to line segment ℓ_j is a $max - min$ function, defined as

$$d_H(\ell_i, \ell_j) = \max_{p_i \in \ell_i}\{\min_{p_j \in \ell_j} d(p_i, p_j)\} \tag{143}$$

It should be noted that Hausdorff distance is directed and so is asymmetric, which means that most of times $d_H(\ell_i, \ell_j)$ is not equal to $d_H(\ell_j, \ell_i)$. An undirected distance measure of two line segments is defined with the directed Hausdorff distance $d_H(\ell_i, \ell_j)$ and $d_H(\ell_j, \ell_i)$ as

$$d(\ell_i, \ell_j) = max(d_H(\ell_i, \ell_j), d_H(\ell_j, \ell_i)) \tag{144}$$

With only eight sonar sensors, an AmigoBot needs to construct the line segment based map incrementally at each sampling time. Two principles are needed for constructing such algorithms to obtain a concise map with complete and accurate information about the environment: one is to save the information about the environment as much as possible and the other is to extract the primitive information about the environment with lines as accurately as possible. Bearing this in mind, three algorithms are developed respectively and implemented successively. Algorithm *Generate − Line* is the first one to insert the obstacles detected into the map and generates lines with some simple criteria. The second algorithm *Line − Fitting* is to extract a line from the preliminary results generated with Algorithm *Generate − Line*. Finally, Algorithm *Merge − Line* is implemented to eliminate the redundant lines to make a concise map.

The line segment map is stored with a data structure of *tree* and each *branch* includes two nodes which represent the two ending points of P_1 and P_2 of a line segment, or one node which represents one ending point of a line segment that has not been generated. The node of a tree is defined with (x, y), where (x, y) is the position of the point in R^2. There are only two different kinds of obstacles in the line segment based map denoted with M, which are lines ℓ_k and points p_j, where $\ell_k \in M$ and $p_j \in M$. Lines represent a set of obstacles such as wall, closed door and some obstacle with length being big enough in the horizontal place. Points represent a kind of obstacles with small length such as the legs of a table or a desk or a walking stick. When a new obstacle B_i is detected, the building algorithm should decide whether it should be inserted as an ending point of a line or it is a discrete point. Simple criteria are defined with distance as follows.

Definition 1. *An obstacle $B_i(x_i, y_i)$ is considered to be in M, if $d(B_i, p_j) < D_1$ or $d(B_i, \ell_k) < D_1$ for any $\ell_k \in M$ or any $p_j \in M$.*

Definition 2. *An obstacle $B_i(x_i, y_i)$ can be jointed with $p_j \in M$ as one new line if $d(B_i, p_j) \geq D_1$ and $d(B_i, p_j) \leq D_2$.*

Definition 3. *An obstacle $B_i(x_i, y_i)$ is added to M as one discrete point if $d(B_i, p_j) > D_2$, $\forall \ell_k \in M$, and $d(B_i, \ell_j) > D_2$, $\forall p_j \in M$.*

Parameters D_1 and D_2 are two distances designed to evaluate the length of obstacles in the horizontal place. It is assumed that $D_1 > 0$ and $D_2 > D_1$. With the three criteria given above, the algorithm to generate a new line is proposed as shown in Algorithm 1. Each obstacle detected is evaluated either to be ignored as a redundant one according to **Definition 1** or to be connected with a starting point to generate a new line according to **Definition 2**, or to be add to M as new discrete point according to **Definition 3**.

The time complexity of this algorithm is only $O(n)$. Although it is primitive and coarse when generating a line, it saves the information as completely as possible and also extracts some preliminary information about the environment with a fast processing speed. Algorithm *Generate − Line* has a fast response to extract a new line, but it neglects the possibility of generating duplicate lines. The following algorithm *Merge − Line* as shown in Algorithm 2 is an effective and simple way to eliminate the redundant lines. The two lines who are reduplicated to each other will be determined with Hausdorff distance.

Definition 4. *Two lines ℓ_i and ℓ_j are merged if $d(\ell_i, \ell_j) < L_{min}$.*

L_{min} is a distance to evaluate the reduplication of two lines. With this definition, if $d(\ell_i, \ell_j) < L_{min}$, then the shorter one will be merged with the longer one. The time complexity of this algorithm is $O(n^2)$. It is an effective approach to make the map more concise. The next step is to refine the line segment based map to be as accurate as possible with Algorithm *Line − Fitting*.

The map generated with the above algorithms has lots of discrete line segments and points, of which many can be jointed and represented with one longer line segment. In this section, an algorithm *Line − Fitting* is proposed to reduce the number of discrete line segments and points, and to refine the map with a satisfactory accuracy. First, the line segments in M are classified with a *pipe* according to **Definitions 5** and **6**. Each pipe is set with a line segment ℓ_i of M ordered by length. All the discrete line segments in the *pipe* have a similar orientation angle and close distance with ending points.

[0]Operator " > " denotes the order of the iterator for each branch in M.

Definition 5. *A line $\ell_j(p_{j1}, p_{j2})$ is considered to be in the Pipe of line ℓ_i only if it satisfies the following two conditions:*

(i) $d_P(p_{j1}, \ell_i) < d_{min}$ and $d_P(p_{j2}, \ell_i) < d_{min}$

(ii) $d(p_{j1}, \ell_i) < d_{max}$ or $d(p_{j2}, \ell_i) < d_{max}$

Definition 6. *A point p_j is considered to be in the Pipe of line ℓ_i only if it satisfies $d_P(p_j, \ell_i) < d_{min}$ and $d(p_j, \ell_i) < d_{max}$.*

The parameters d_{min} and d_{max} are two distances to adjust *Pipe*. With **Definitions 5** and **6**, the line segments and points in *Pipe* can be chained and fitted with a longer line segment ℓ_f. Next, least squares fitting is applied to calculating the line ℓ_f with minimum sum of squares of offsets of all the points in *Pipe* from line ℓ_f. Define line ℓ_f with $y = kx+b$, $\forall k \in (-\infty, \infty)$. Assume that there are N points in *Pipe*, $P_i(x_i, y_i) \in Pipe$, $i = 1, 2, \ldots, N$. Let $X = \sum_{i=1}^{N} x_i$, $Y = \sum_{i=1}^{N} y_i$, $X_1 = \sum_{i=1}^{N} x_i y_i$, and $X_2 = \sum_{i=1}^{N} x_i^2$. According to the least squares method, if $(NX_2 - X^2) \neq 0$, then the parameters k and b can be calculated by

$$\begin{aligned} k &= (NX_1 - XY)/(NX_2 - X^2) \\ b &= (X_2Y - XX_1)/(NX_2 - X^2) \end{aligned} \tag{145}$$

Otherwise, if $(NX_2 - X^2) = 0$, then the line ℓ_f is defined as $x + b = 0$ and the parameter b is calculated by

$$b = -X/N \tag{146}$$

Since the nature of the least-square method is to minimize the sum of squares of offsets for all points, it does not have the ability to distinguish the points with gross offset. Therefore, gross offset checking with distance d_{min} is added to algorithm *Line – Fitting*. It eliminates the point $p_j \in Pipe$ with maximum offset $d(p_j, \ell_f)$ which is bigger than d_{min} and then re-calculate the line ℓ_f with the left points. This checking will be repeated until the offsets of all point are little than d_{min} .

After the parameter of line ℓ_f is calculated with satisfactory offset d_{min}, the two ending points of line ℓ_f should be determined. In this report, a simple criterion is utilized to select the two ending points. First, we find four points with $max(x_i)$, $min(x_i)$, $max(y_i)$, and $min(y_i)$ in *Pipe*, and calculate their projecting points $Pj_i(x_{pi}, y_{pi})$ on ℓ_f with **Definition 7**. The two projecting points with $min(x_{pi})$ and $max(x_{pi})$ or with $min(y_{pi})$ and $max(y_{pi})$ are then selected as the ending points of ℓ_f.

Definition 7. *The projecting point $p_j \in \ell$ of a point p_i to line segment ℓ is the point with the minimum distance $d(p_i, \ell)$.*

The main pseudo code of the algorithm *Line – Fitting* is shown in Algorithm 3. The time complexity of this algorithm is $O(n^2)$. Algorithm *Line – Fitting* is implemented before generating the Voronoi diagram. It remarkably reduces the scale of the line segment based map and also reduces the complexity of computing the Voronoi diagram. The Line segment based map only contains the information of obstacles about the real-world environment and cannot be directly used for motion planning of robots. The motion of a robot should be planned in the C_{free} space. The Voronoi diagram provides a retraction method to extract the information about the C_{free} space from the line segment based map and retract it as a topological graph.

In this report, the notion of *Configuration Space* is defined with $C = R^2$ and C_{free} is a polygonal limited subset whose boundary ∂C_{free} is entirely made of line segments. In computational geometry, the Voronoi diagram is a partitioning process of a plane into regions based on distance to points in a specific subset of the plane according to the nearest-neighbor rule. The set of points in the generalized Voronoi diagram has the useful property of maximizing the clearance between the points and obstacles. Normally, for $q \in C_{free}$ define

$$Clearance(q) = \min_{p \in \partial C_{free}} \|q - p\| \tag{147}$$

and

$$near(q) = \{\|q - p\|_{p \in \partial C_{free}} = Clearance(q)\} \tag{148}$$

where $near(q)$ is the set of boundary points of C_{free} minimizing the distance to q. Then the generalized Voronoi diagram is given by $\{q \in C_{free}, |near(q)| > 1\}$, that is, the set of points in C_{free} with at least two nearest neighbors in the boundary of C_{free}. Therefore, the generalized Voronoi diagram is a retraction method, which is the locus of points that are equidistant from the closest two or more obstacle boundaries.

As a retraction method, the generalized Voronoi diagram denoted with *Vor*, has the following properties [132]:

(i) The Voronoi diagram is a retraction which has a continuous map $\rho : C_{free} \rightarrow Vor$, $Vor \subset C_{free}$, and $\rho(Vor) = Vor$.

(ii) The Voronoi diagram is a retraction with connectivity preserving, i.e., $\forall x \in C_{free}$, q and $\rho(q)$ belong to the same connected component of C_{free}.

(iii) There exists a path between $q \in C_{free}$ and $p \in C_{free}$ if and only if there exists a path in Vor between $\rho(p)$ and $\rho(q)$.

For example, define the map $\rho(q)$, $q \in C_{free}$, as the nearest edge $e_i \in Vor$ of q. Then the properties listed above can be easily proved and illustrated [133].

In this report, Fortune's sweep line algorithm in [117] is applied to generate a Voronoi diagram from the line segment based map. Due to the fact that the discrete points are mostly eliminated by Algorithms $Merge - Line$ and $Line - Fitting$, only line segments in M are included in generating the Voronoi diagram. In addition, two points of the position of an AmigoBot q_r and the goal q_G are added as obstacles to guarantee that there always exists a nearest edge to q_S (q_G) in Vor and no other obstacles lie between them. Also, the intersecting line segments are disposed by use of Algorithm $Clear - Intersection$ to reduce the complexity of the generating algorithm. Only two types of intersecting line segments are considered as shown in Fig. 53. With Algorithm $Clear - Intersection$, the intersecting line segments

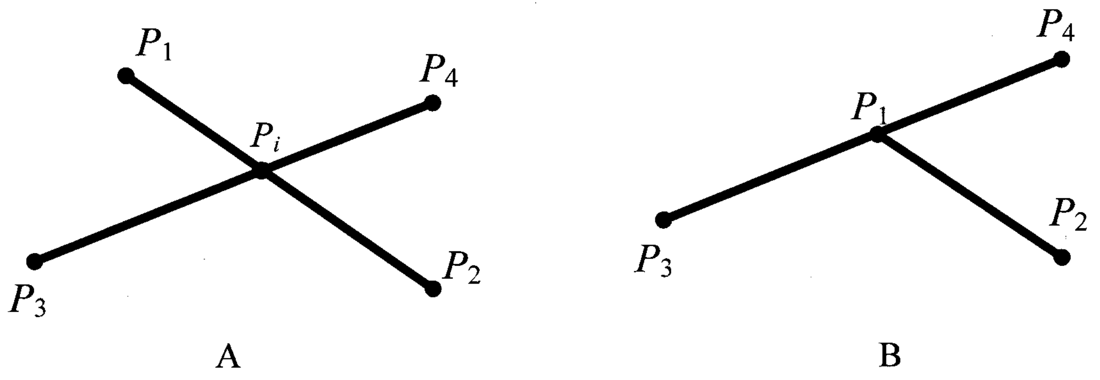


Figure 53: Two Types of Intersection

as (A) in Fig. 53 are divided into four new line segments $\ell_1(P_1, P_i)$, $\ell_2(P_2, P_i)$, $\ell_3(P_3, P_i)$, and $\ell_4(P_4, P_i)$. As in (B) the line $\ell(P_3, P_4)$ is divided into two new lines $\ell_1(P_3, P_1)$ and $\ell_2(P_2, P_i)$.

There are two categories of geometric configuration in the Voronoi diagram: linear line segments and parabolic curves. To facilitate the searching for the shorted path with Dijkstra's algorithm, parabolic curves are discretized as line segments. Thus, in the following $Path - Planning$ algorithm, there are only line segments in the Voronoi diagram Vor. After the Voronoi diagram Vor is built from the line segment based map M, Clearance is checked before planning the path. For each edge $e_i \in Vor$ included in searching the accessible path, the action is only taken if $Clearance(q) > SafeDist$, $\forall q \in e_i$, where $SafeDist$ is a minimum distance between the gap formed by obstacles that can allow the AmigoBot to go through with a safe margin.

The path planning algorithm is shown in Algorithm 4. q_S is the starting point and q_G is the ending point of the path. The map $\rho(q)$, $q \in C_{free}$, is defined as the nearest edge $e_i \in Vor$ of q. Dijkstra's Algorithm is applied to search the shorted path in Vor. The line segment added from q_S to $\rho(q_S)$ is defined as the line segment $\ell(q_S, q_P)$, where q_P is the point associated with $d(q_S, \rho(q_S))$. It is the same for the line segment added from q_G to $\rho(q_G)$. The time complexity of this Dijkstra's searching algorithm is $O(V^2)$ where V is the number of vertices for the Voronoi graph. The path planning algorithm is only implemented when the obstacle newly detected is in collision with the $Path$.

In this report, the LOS approach is used to design the path tracking control law for an AmigBot. With this LOS approach, the path tracking problem is simplified as an angle tracking problem. Specifically, assume that the current line segment to be tracked is $e_i \in Path$. Denote the position of the AmigoBot in R^2 by q_r and define $q_P \in e_i$ as the point associated with $d(q_r, e_i)$ and $q_R \in e_i$ as the point associated with $d(q_R, q_P) = kL$ ahead of q_P along the forward direction, where $k > 0$ is a designed parameter and L is the length of the AmigoBot. Then the LOS angle θ_R to be tracked for the AmigoBot is defined as the direction from point q_r to q_R.

A PD controller is designed to track angle θ_R as in (150). The angle tracking error is defined in (149).

$$\theta_E = \begin{cases} \sin(\theta_r - \theta_R), & \|\theta_r - \theta_R\| < \pi/2 \\ sign(\theta_r - \theta_R), & \|\theta_r - \theta_R\| \geq \pi/2 \end{cases} \tag{149}$$

where $\theta_r, \theta_R \in [-\pi, \pi]$ and $sign(\cdot)$ is the sign function. Then, the PD control law for the right and left wheels of the AmigoBot is designed as in (150).

$$\begin{aligned} V_R &= V_0 \\ V_L &= V_R + k_p\theta_E + k_d\dot{\theta}_E \end{aligned} \tag{150}$$

where V_L and V_R are the velocities of the left and right wheels, V_0 is the desired constant speed, and $k_p > 0$ and $k_d > 0$ are the parameters of the PD controller.

A Simulation in MobileSim environment is conducted to verify the proposed motion planning method for the AmigoBot. MobileSim is a simulator for Mobile Robots/ActivMedia robots based on the Stage robot simulator library [130] and is widely used to testify the motion planning algorithms and control methods. The setting environment is shown in Fig. 54. A box is added in order to enclose the map and avoid the infinite edges while generating the Voronoi diagram. The four points of the box in this simulation are given with $A(-3000, -4000)$, $B(-3000, 4000)$, $C(21000, 4000)$ and $D(21000, -4000)$.

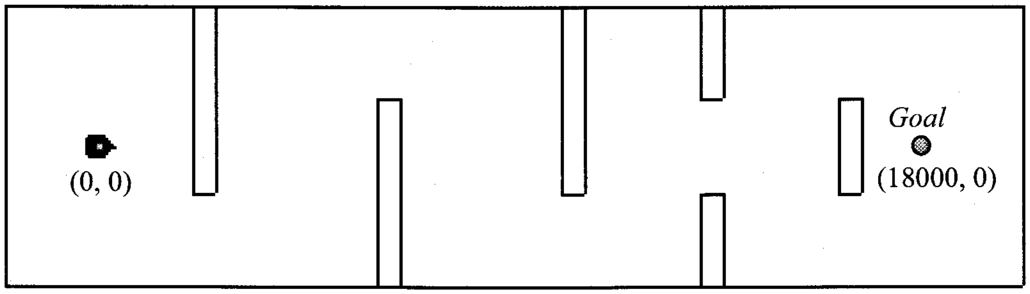


Figure 54: Environment Setting in MobileSim

The parameters in the map building algorithms are designed with $D_1 = 20$ mm, $D_2 = 200$ mm, $L_{min} = 50$ mm, $d_{min} = 50$ mm, and $d_{max} = 200$ mm. The parameters in motion planning are designed with $SafeDist = 300$ mm, $V_0 = 50$ mm/s, $k_p = 60$, and $k_d = 5$. The algorithms are coded with $C++$ and compiled in Visual Studio 2008. All the results are saved in text files and plotted as figures through MATLAB. The results in Fig. 55 and Fig. 56 show the Voronoi diagram and reference path generated at the sampled time instants $t = 534$ s and $t = 704$ s. With *Clearance* in the algorithm, it can be seen that the reference path is safe to obstacles and is also the shortest from the current position of the AmigoBot to the final destination.

The final Voronoi diagram generated by our proposed algorithm is given by Fig. 57. It can be seen that the Voronoi diagram is an excellent approach to extract the topological information from C_{fee}. The trace of the AmogoBot from the initial position to the final destination is shown in Fig. 58. In most of time the robot is tracking the path with the maximum clearance, which is the dominant advantage of the Voronoi diagram. The line segment based map finally constructed in Fig. 58 is concise and has only 139 segments. It successfully represents the environment with the primary information and also with a satisfactory accuracy. From these results, it can be seen that the proposed line segment based map and motion planning method with the Voronoi diagram have achieved a successful application for an AmigoBot to move in an unknown environment.

The experimental environment setting for AmigoBot is shown in Fig. 59. The initial position of AmigoBot is set with $(0, 0)$ and the goal position is given with $(8000, 0)$. The four points of the added box to map in this experiment are given with $A(-3000, 5000)$, $B(-3000, -5000)$, $C(15000, -5000)$ and $D(15000, 5000)$. The parameters in the map building and motion planning algorithms are all set with the same values as that in the simulation above. Experiment results are given in the following figures. Fig. 60 and Fig. 61 give the constructed map M(plotted with black lines), the Voronoi diagram Vor (plotted with blue lines)and the reference $Path$ (plotted with red lines) respectively at the sample times $k = 676$ and at the last sample time . It is obvious that the $path$ is collision-free and is with safe clearance. Also, it can be seen that the constructed line segment map M in Fig. 62 is almost perfect to represent the continuous obstacles such as the wall and the box.

The trace of AmigoBot in Fig. 58 illustrates the biggest advantage of the proposed Voronoi diagram based motion planning method once again, that is, AmigoBot moves from the initial position to the goal with almost maximum clearance. However, some errors caused by the limitations of sonar sensors are also shown in Fig. 62. The obstacles in

/S l.c.

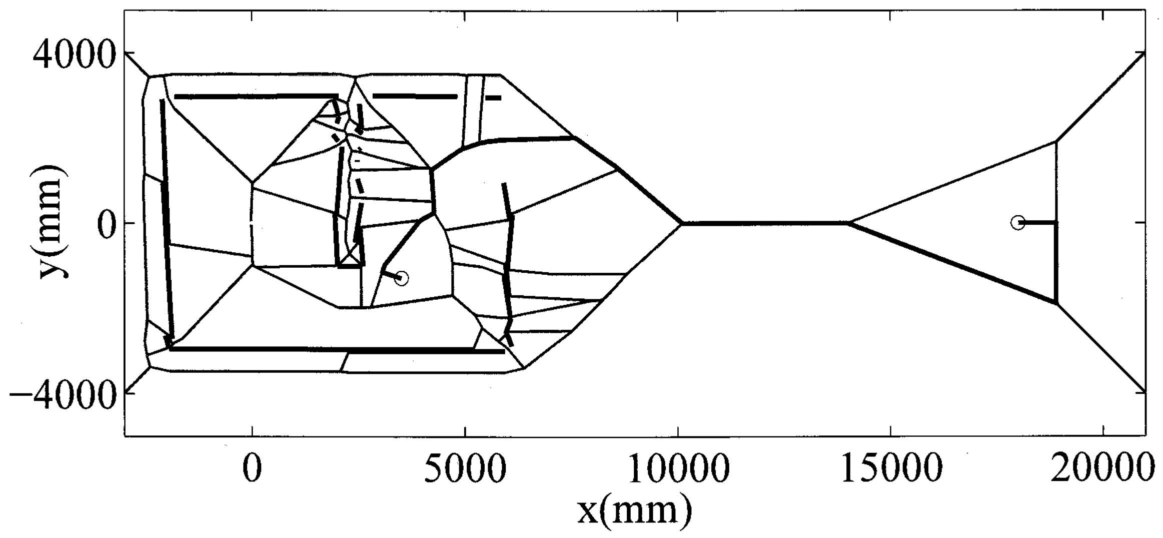


Figure 55: *Vor* and *Path* at Sampled Time Instant t = 534 s

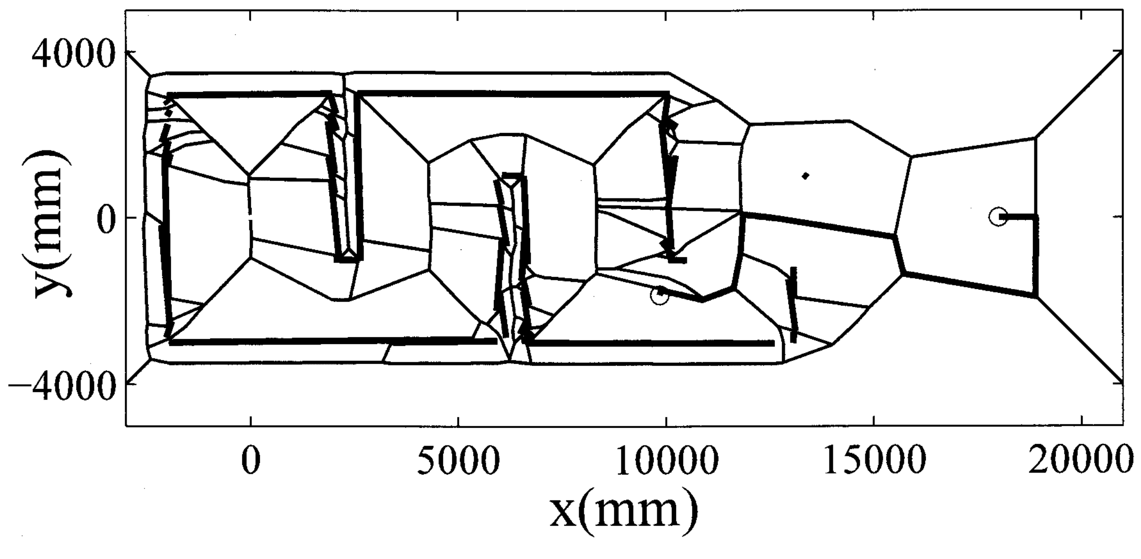


Figure 56: *Vor* and *Path* at Sampled Time Instant t = 704 s

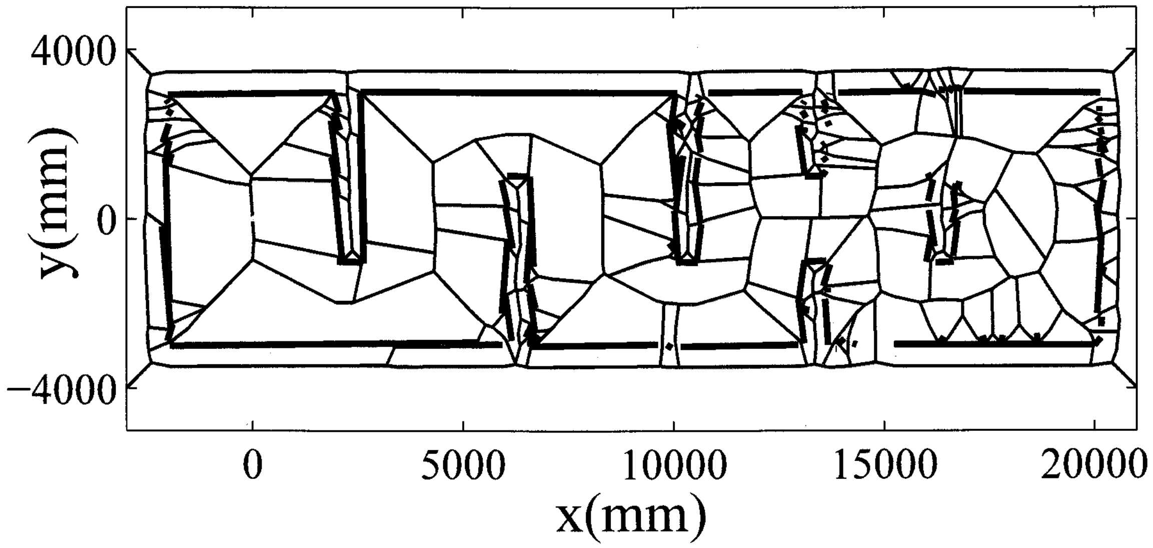


Figure 57: *M* and *Vor* at Final Time

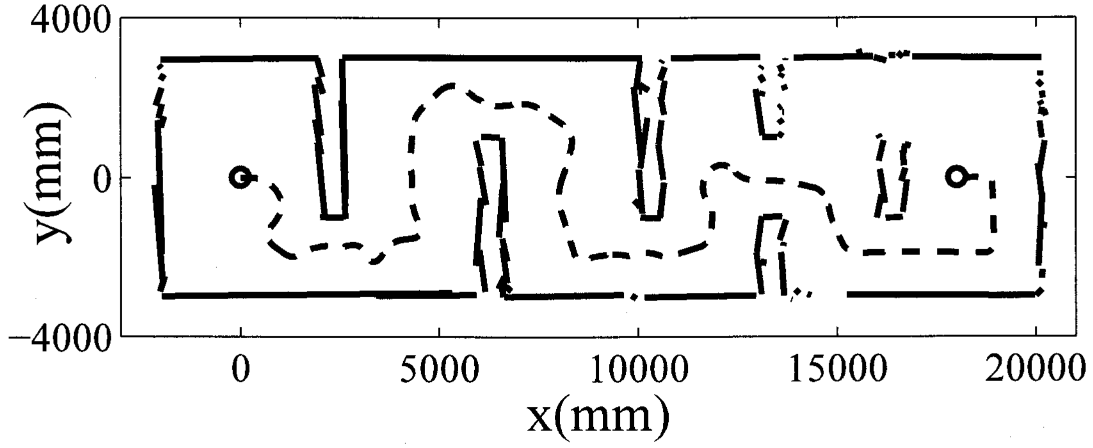


Figure 58: Trace of AmigoBot in *M*

the circle (plotted with green dotted line) are not existent in the real world but are caused by specular reflection. These errors are inevitable in experiment and need to be erased with heuristic knowledge.

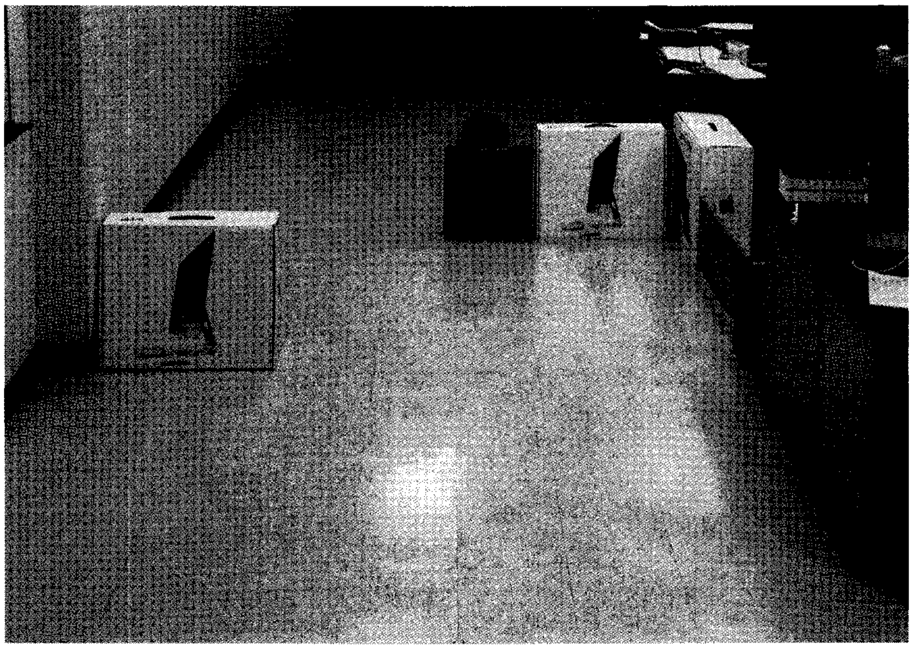


Figure 59: Experimental Setting for AmigoBot

6 Research Personnel Supported

Faculty Supported:

- Qing Hui, Principal Investigator
- Philip Smith, Co-PI
- Jin Cheng, Visiting Scholar

Students Graduated:

- Haopeng Zhang, Ph.D., August 2014
- Zhenyi Liu, Ph.D., August 2014
- Xianlin Zeng, Ph.D., August 2015

Technicians:

- Travis Cotton, Programmer

7 Publications (07/01/2013-08/29/2015)

Published Journal Papers

1. **Q. Hui**, H. Zhang, and Z. Liu, "On Robust and Optimal Imperfect Information State Equipartitioning for Network Systems," *Journal of the Franklin Institute*, vol. 352, no. 9, pp. 3410-3446, 2015.

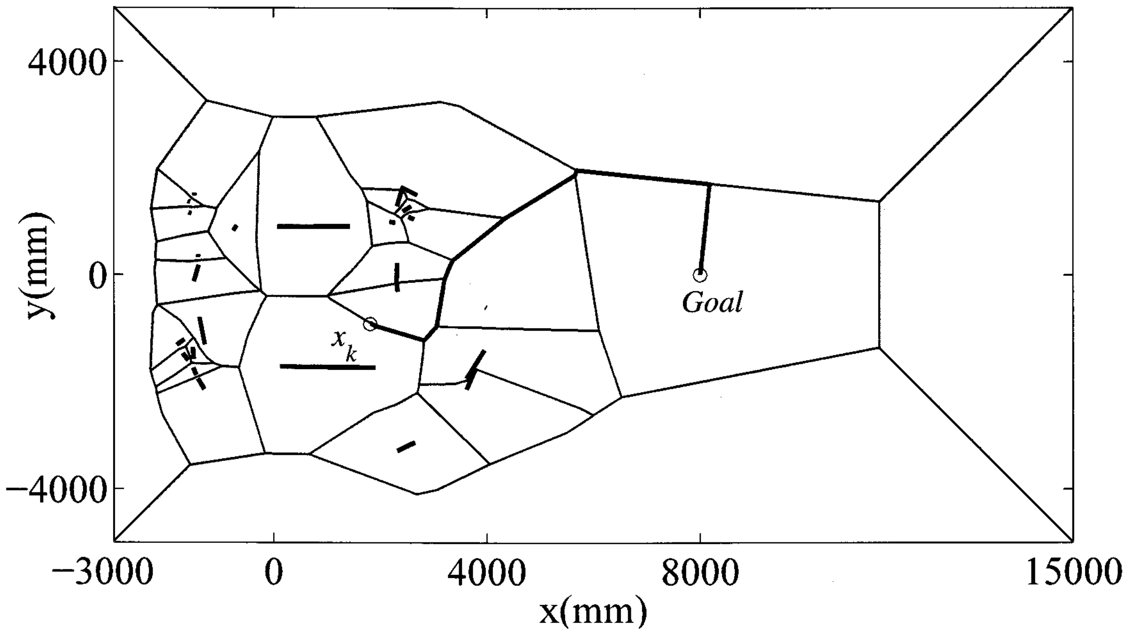


Figure 60: *Vor* and *Path* at k = 676

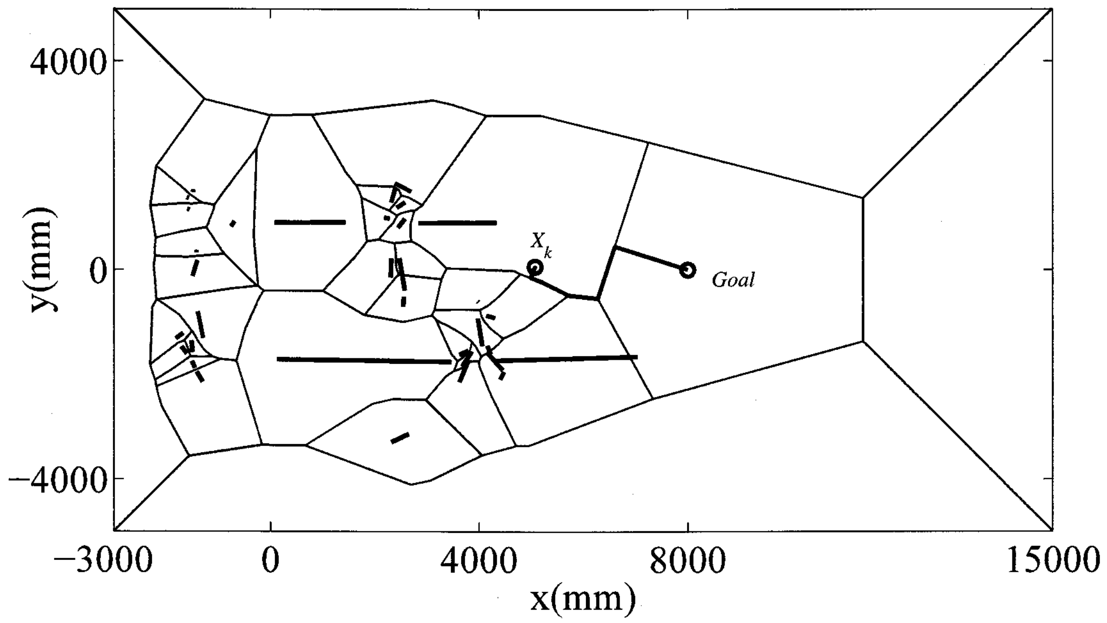


Figure 61: *Vor* and *Path* at Final Time

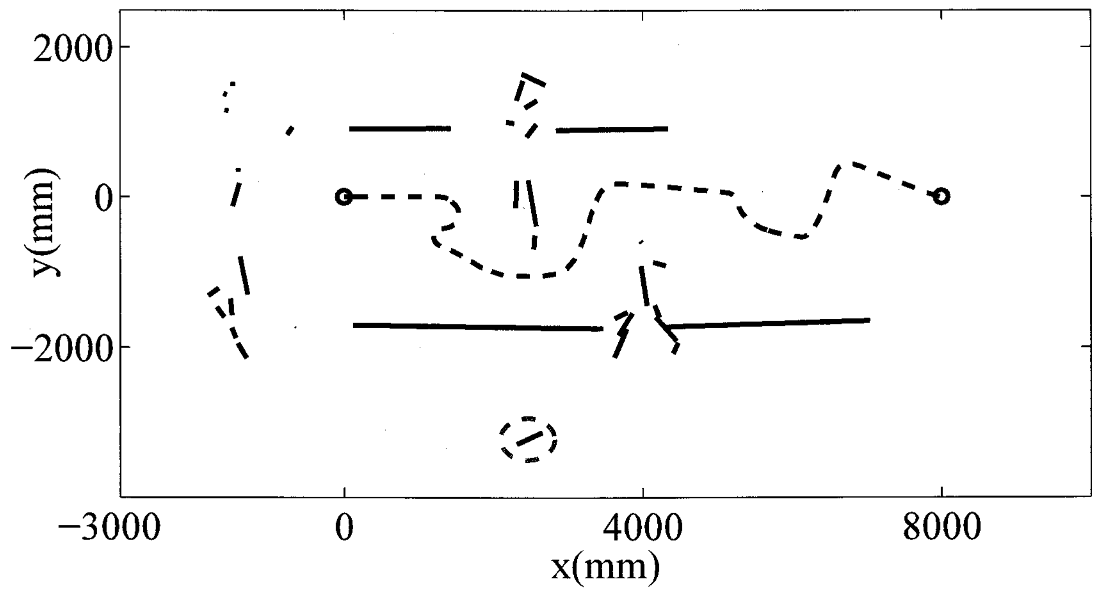


Figure 62: Trace of AmigoBot in M

2. **Q. Hui** and H. Zhang, "Optimal Balanced Coordinated Network Resource Allocation Using Swarm Optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 5, pp. 770-787, 2015.
3. A. L'Afflitto, W. M. Haddad, and **Q. Hui**, "Optimal Control for Linear and Nonlinear Semistabilization," *Journal of the Franklin Institute*, vol. 352, no. 3, pp. 851-881, 2015.
4. X. Zeng, Z. Liu, and **Q. Hui**, "Energy Equipartition Stabilization and Cascading Resilience Optimization for Geospatially Distributed Cyber-Physical Network Systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 25-43, 2015.
5. W. M. Haddad, **Q. Hui**, and J. M. Bailey, "Human Brain Networks: Spiking Neuron Models, Multistability, Synchronization, Thermodynamics, Maximum Entropy Production, and Anesthetic Cascade Mechanisms," *Entropy*, vol. 16, no. 7, pp. 3939-4003, 2014.
6. W. M. Haddad, S. G. Nersesov, **Q. Hui**, and M. Ghasemi, "Formation Control Protocols for Nonlinear Dynamical Systems via Hybrid Stabilization of Sets," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 136, no. 5, Article ID 051020, 13 pages, 2014.
7. Jinglai Shen, Jianghai Hu, and **Qing Hui**, "Semistability of Switched Linear Systems with Application to PageRank Algorithms," *European Journal of Control*, vol. 20, no. 3, pp. 132-140, 2014.
8. **Qing Hui**, Wassim M. Haddad, James M. Bailey, and Tomohisa Hayakawa, "A Stochastic Mean Field Model for an Excitatory and Inhibitory Synaptic Drive Cortical Neuronal Network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 4, pp. 751-763, 2014.
9. Xianlin Zeng, **Qing Hui**, Wassim M. Haddad, Tomohisa Hayakawa, and James M. Bailey, "Synchronization of Biological Neural Network Systems with Stochastic Perturbations and Time Delays," *Journal of the Franklin Institute*, vol. 351, no. 3, pp. 1205-1225, 2014.
10. Yajuan Tang, Xiapu Luo, **Qing Hui**, and Rocky K. C. Chang, "Modeling the Vulnerability of Feedback-Control Based Internet Services to Low-Rate DoS Attacks ," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 3, pp. 339-353, 2014.

Accepted Journal Papers

1. X. Zeng and **Q. Hui**, "Energy-Event-Triggered Hybrid Supervisory Control for Cyber-Physical Network Systems," *IEEE Transactions on Automatic Control*, to appear.

2. **Q. Hui**, "Further Results on Paracontracting Matrices and Correction to 'Optimal Semistable Control in *Ad Hoc* Network Systems: A Sequential Two-Stage Approach'," *IEEE Transactions on Automatic Control*, to appear.

Published Conference Papers

1. H. Zhang and **Q. Hui**, "New Multiagent Coordination Optimization Algorithms for Mixed-Binary Nonlinear Programming with Control Applications," *2014 IEEE Symposium on Computational Intelligence in Control and Automation*, Orlando, FL, December 2014.

2. Z. Liu, P. Smith, T. Park, A. A. Trindade, and **Q. Hui**, "Novel Response Surface Methodologies with Design of Experiment for Source Localization in Unknown Spatial-Temporal Fields," *The 53rd IEEE Conference on Decision and Control*, pp. 5724-5729, Los Angeles, CA, December 2014.

3. Z. Liu, P. Smith, and **Q. Hui**, "Contaminant Detection and Localization with Design of Experiment and Response Surface Methodology," *2014 ASME Dynamic Systems and Control Conference*, San Antonio, TX, October 2014.

4. X. Zeng and **Q. Hui**, "Energy Equipartition Control for Geospatial Cyber-Physical Network Systems," *2014 ASME Dynamic Systems and Control Conference*, San Antonio, TX, October 2014.

5. H. Zhang and **Q. Hui**, "Multiagent Coordination Optimization Based Model Predictive Control Strategy with Application to Balanced Resource Allocation," *2014 ASME Dynamic Systems and Control Conference*, San Antonio, TX, October 2014.

6. **Q. Hui**, "Multistability Analysis of Discontinuous Dynamical Systems via Finite Trajectory Length," *2014 World Automation Congress*, Big Island, HI, August 2014.

7. H. Zhang, F. Zhang, and **Q. Hui**, "A Speed-Up and Speed-Down Strategy for Swarm Optimization," *2014 ACM Genetic and Evolutionary Computation Conference*, pp. 1481-1482, Vancouver, Canada, July 2014.

8. A. L'Afflitto, W. M. Haddad, and **Q. Hui**, "Optimal and Singular Control for Linear and Nonlinear Semistabilization," *2014 American Control Conference*, pp. 5669-5674, Portland, OR, June 2014.

9. **Q. Hui** and H. Zhang, "Paracontracting Multiagent Coordination Optimization: Convergence and Parallelization," *2014 American Control Conference*, pp. 1704-1709, Portland, OR, June 2014.

Accepted Conference Papers

1. C. Peng and **Q. Hui**, "ε Constrained CCPSO with Different Improvement Detection Techniques for Large-Scale Constrained Optimization," *The 49th Hawaii International Conference on System Sciences*, Kauai, HI, January 2016.

2. X. Zeng and **Q. Hui**, "Partial Cluster Stabilization and Partial Cascade Stabilization of Physical Networks," *The 5th IFAC Conference on Analysis and Design of Hybrid Systems*, Atlanta, GA, October 2015.

Conference Presentations

1. **Q. Hui**, H. Zhang, and Z. Liu, "On Robust and Optimal Imperfect Information Consensus Protocols for Network Systems," *SIAM Workshop on Network Science*, Chicago, IL, July 2014.

2. H. Zhang and **Q. Hui**, "Coupled Spring Forced Multiagent Coordination Optimization for Mixed-Binary Nonlinear Programming," *2014 SIAM Annual Meeting*, Chicago, IL, July 2014.

References

[1] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 26, no. 1, pp. 29–41, 1996.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, Australia, 1995, pp. 1942–1948.

[3] H. Zhang and Q. Hui, "A new hybrid swarm optimization algorithm for power system vulnerability analysis and sensor network deployment," in *2013 Int. Joint Conf. Neural Netw.*, Dallas, TX, 2013, pp. 1221–1228.

[4] Y. Tang, X. Luo, Q. Hui, and R. K. C. Chang, "On generalized low-rate denial-of-quality attack against Internet services," in *17th Int. Workshop Quality of Service*, Charleston, SC, 2009.

[5] ——, "Modeling the vulnerability of feedback-control based Internet services to low-rate DoS attacks," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 3, pp. 339–353, 2014.

[6] J. Mercieca and S. G. Fabri, "Particle swarm optimization for nonlinear model predictive control," in *5th Int. Conf. Adv. Eng. Comput. Appl. Sci.*, 2011, pp. 88–93.

[7] X. Wang and J. Xiao, "PSO-based model predictive control for nonlinear processes," in *Advances in Natural Computation*, L. Wang, K. Chen, and Y. S. Ong, Eds. Berlin, Germany: Springer-Verlag, 2005, vol. 3611, pp. 196–203.

[8] M. Clerc, "The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization," in *Proc. 1999 IEEE Congr. Evolut. Comput.*, Washington, DC, 1999, pp. 1951–1957.

[9] C. A. Coello Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. 2002 IEEE Congr. Evolut. Comput.*, Honolulu, HI, 2002, pp. 1051–1056.

[10] X. Hu and R. C. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," in *Proc. 2002 IEEE Congr. Evolut. Comput.*, Honolulu, HI, 2002, pp. 1677–1681.

[11] C. A. Coello Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Trans. Evolut. Comput.*, vol. 8, no. 3, pp. 256–279, 2004.

[12] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *Evolutionary Programming VII*, V. Porto, N. Saravanan, D. Waagen, and A. Eiben, Eds. Berlin, Germany: Springer-Verlag, 1998, vol. 1447, pp. 591–600.

[13] Z. Liu, S. Deshpande, and Q. Hui, "Quantized particle swarm optimization: An improved algorithm based on group effect and its convergence analysis," in *Proc. 2011 ASME Dyna. Syst. Control Conf.*, Arlington, VA, 2011.

[14] H. Zhang and Q. Hui, "Multiagent coordination optimization: A control-theoretic perspective of swarm intelligence algorithms," in *2013 IEEE Congr. Evolut. Comput.*, Cancun, Mexico, 2013, pp. 3339–3346.

[15] ——, "Convergence analysis and parallel computing implementation for the multiagent coordination optimization algorithm with applications," in *2013 IEEE Int. Conf. Automat. Sci. Eng.*, Madison, WI, 2013, pp. 837–842.

[16] Q. Hui and H. Zhang, "Convergence analysis and parallel computing implementation for the multiagent coordination optimization algorithm," Contr. Sci. Eng. Lab., Dept. Mech. Eng., Texas Tech Univ., Lubbock, TX, Tech. Rep. CSEL-06-13, 2013 [Online], Available: http://arxiv.org/abs/1306.0225.

[17] E. Bonabeau, M. Dorigo, and G. Thèraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. Oxford, U.K.: Oxford Univ. Press, 1999.

[18] Q. Hui, "Hybrid consensus protocols: An impulsive dynamical system approach," *Int. J. Control*, vol. 83, pp. 1107–1116, 2010.

[19] ——, "Finite-time rendezvous algorithms for mobile autonomous agents," *IEEE Trans. Autom. Control*, vol. 56, no. 1, pp. 207–211, 2011.

[20] Z. Liu and Q. Hui, "A numerical gradient based technique and directed neighborhood structure for constrained particle swarm optimization," in *2013 Amer. Control Conf.*, Washington, DC, 2013, pp. 4687–4692.

[21] Q. Hui and H. Zhang, "Paracontracting multiagent coordination optimization: Convergence and parallelization," in *Proc. Amer. Control Conf.*, Portland, OR, 2014.

[22] J. von Neumann, *Functional Operators. Vol. II: The Geometry of Orthogonal Spaces*, ser. Ann. Math. Studies. Princeton, NJ: Princeton Univ. Press, 1950, vol. 22.

[23] S. Nelson and M. Neumann, "Generalizations of the projection method with applications to SOR theory for Hermitian positive semidefinite linear systems," *Numer. Math.*, vol. 51, no. 2, pp. 123–141, 1987.

[24] J. Shen, J. Hu, and Q. Hui, "Semistability of switched linear systems with applications to sensor networks: A generating function approach," in *Proc. IEEE Conf. Decision Control*, Orlando, FL, 2011, pp. 8044–8049.

[25] L. Elsner, I. Koltracht, and M. Neumann, "On the convergence of asynchronous paracontractions with applications to tomographic reconstruction from incomplete data," *Linear Alg. Appl.*, vol. 130, pp. 65–82, 1990.

[26] Q. Hui, "Optimal semistable control in *ad hoc* network systems: A sequential two-stage approach," *IEEE Trans. Autom. Control*, vol. 58, no. 3, pp. 779–784, 2013.

[27] D. S. Bernstein, *Matrix Mathematics,* 2nd ed. Princeton, NJ: Princeton Univ. Press, 2009.

[28] X. Wang and Z. Cheng, "The convergence of infinite product of uniformly paracontracting matrices and its application in consensus of the Vicsek model," *J. Sys. Sci. Math. Scis.*, vol. 33, no. 6, pp. 724–731, 2013.

[29] Q. Hui and H. Zhang, "Semistability-based convergence analysis for paracontracting multiagent coordination optimization," Contr. Sci. Eng. Lab., Dept. Mech. Eng., Texas Tech Univ., Lubbock, TX, Tech. Rep. CSEL-08-13, 2013 [Online], Available: http://arxiv.org/abs/1308.2930.

[30] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Info. Process. Lett.*, vol. 85, no. 6, pp. 317–325, 2003.

[31] Q. Hui and H. Zhang, "Multiagent coordination optimization," submitted for publication.

[32] Q. Hui, W. M. Haddad, and S. P. Bhat, "Finite-time semistability and consensus for nonlinear dynamical networks," *IEEE Trans. Autom. Control*, vol. 53, pp. 1887–1900, 2008.

[33] ——, "Semistability, finite-time stability, differential inclusions, and discontinuous dynamical systems having a continuum of equilibria," *IEEE Trans. Autom. Control*, vol. 54, no. 10, pp. 2465–2470, 2009.

[34] Q. Hui and W. M. Haddad, "Distributed nonlinear control algorithms for network consensus," *Automatica*, vol. 44, pp. 2375–2381, 2008.

[35] Q. Hui, "Optimal distributed linear averaging," *Automatica*, vol. 47, no. 12, pp. 2713–2719, 2011.

[36] ——, "Semistability of nonlinear systems having a connected set of equilibria and time-delays," *IEEE Trans. Autom. Control*, vol. 57, no. 10, pp. 2615–2620, 2012.

[37] T. H. Fay and S. D. Graham, "Coupled spring equations," *International Journal of Mathematical Education in Science and Technology*, vol. 34, no. 1, pp. 65–79, 2003.

[38] L. Gurvits, "Stability of discrete linear inclusion," *Linear Alg. Appl.*, vol. 231, pp. 47–85, 1995.

[39] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces.* New York: Springer-Verlag, 2011.

[40] E. Zeidler, *Nonlinear Functional Analysis and Its Applications II/B. Nonlinear Monotone Operators.* New York: Springer-Verlag, 1985.

[41] R. T. Rockafellar, *Convex Analysis.* Princeton, NJ: Princeton Univ. Press, 1970.

[42] X. Li, K. Tang, M. N. Omidvar, Z. Yang, and K. Qin, "Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization," Evolut. Comput. Machine Learning Group, RMIT Univ., Melbourne, Australia, Tech. Rep., 2013 [Online], Available: http://goanna.cs.rmit.edu.au/~xiaodong/cec13-lsgo/competition/cec2013-lsgo-benchmark-tech-report.pdf.

[43] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions," INRIA Saclay, Orsay, France, Tech. Rep. RR-6829, 2009 [Online], Available: http://hal.inria.fr/docs/00/36/26/33/PDF/RR-6829.pdf.

[44] Y. Liu, Z. Qin, Z. Shi, and J. Lu, "Center particle swarm optimization," *Neurocomputing*, vol. 70, no. 4, pp. 672–679, 2007.

[45] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evolut. Comput.*, vol. 10, no. 3, pp. 281–295, 2006.

[46] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," *IEEE Trans. Evolut. Comput.*, vol. 8, no. 3, pp. 204–210, 2004.

[47] F. You and I. E. Grossmann, "Mixed-integer nonlinear programming models and algorithms for large-scale supply chain design with stochastic inventory management," *Industrial and Engineering Chemistry Research*, vol. 47, no. 20, pp. 7802–7817, 2008.

[48] D. Klabjan, E. L. Johnson, G. L. Nemhauser, E. Gelman, and S. Ramaswamy, "Solving large airline crew scheduling problems: Random pairing generation and strong branching," *Computational Optimization and Applications*, vol. 20, no. 1, pp. 73–91, 2001.

[49] A. Federgruen and P. Zipkin, "A combined vehicle routing and inventory allocation problem," *Operations Research*, vol. 32, no. 5, pp. 1019–1037, 1984.

[50] F. Noonan and R. Giglio, "Planning electric power generation: a nonlinear mixed integer model employing benders decomposition," *Management Science*, vol. 23, no. 9, pp. 946–956, 1977.

[51] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya *et al.*, "An algorithmic framework for convex mixed integer nonlinear programs," *Discrete Optimization*, vol. 5, no. 2, pp. 186–204, 2008.

[52] M. A. Duran and I. E. Grossmann, "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical programming*, vol. 36, no. 3, pp. 307–339, 1986.

[53] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Networks*, 1995, pp. 1942–1946.

[54] H. Zhang and Q. Hui, "Coupled spring forced multiagent coordination optimization for mixed-binary nonlinear programming," in *2014 SIAM Annual Meeting*.

[55] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *1997 IEEE International Conference on Systems, Man, and Cybernetics*, vol. 5, 1997, pp. 4104–4108.

[56] H. Zhang and Q. Hui, "Binary multiagent coordination optimization with application to formation control design," in *2013 IEEE Congr. Evolut. Comput.*, Cancun, Mexico, 2013, pp. 1968–1975.

[57] Q. Hui and H. Zhang, "Optimal balanced coordinated network resource allocation using swarm optimization," in *Proc. IEEE Conf. Decision Control*, Maui, HI, 2012, pp. 3936–3941.

[58] M. A. Khanesar, M. Teshnehlab, and M. A. Shoorehdeli, "A novel binary particle swarm optimization," in *2007 Mediterranean Conference on Control and Automation*, 2007, pp. 1–6.

[59] G. Pampara, A. Engelbrecht, and N. Franken, "Binary differential evolution," in *2006 IEEE Congress on Evolutionary Computation*, 2006, pp. 1873–1879.

[60] H. Zhang and Q. Hui, "Topological heterogeneity and optimality analysis for multiagent formation," in *Proc. IEEE Conf. Decision Control*, 2012, pp. 5954–5959.

[61] W. Wu, I. D. Couzin, and F. Zhang, "Bio-inspired source seeking with no explicit gradient estimation," in *Proc. IFAC Workshop on Distributed Estimation and Control in Networked Systems*, 2012, pp. 240–245.

[62] J. Liang, B. Qu, P. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," *Comput. Intell. Lab., Zhengzhou Univ., China, and Nanyang Tech. Univ., Singapore, Technical Report*, vol. 2012-12, 2013.

[63] E. F. Camacho and C. B. Alba, *Model Predictive Control.* Springer, 2013.

[64] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: theory and practicea survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[65] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, 2010.

[66] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright, "Distributed MPC strategies with application to power system automatic generation control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1192–1206, 2008.

[67] J. Wen, Q. Wu, D. Turner, S. Cheng, and J. Fitch, "Optimal coordinated voltage control for power system voltage stability," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 1115–1122, 2004.

[68] A. Ilzhoefer, B. Houska, and M. Diehl, "Nonlinear MPC of kites under varying wind conditions for a new class of large-scale wind power generators," *International Journal of Robust and Nonlinear Control*, vol. 17, no. 17, pp. 1590–1599, 2007.

[69] F. Borrelli, P. Falcone, T. Keviczky, and J. Asgari, "MPC-based approach to active steering for autonomous vehicle systems," *International Journal of Vehicle Autonomous Systems*, vol. 3, no. 2, pp. 265–291, 2005.

[70] D. Dougherty and D. Cooper, "A practical multiple model adaptive strategy for single-loop MPC," *Control engineering practice*, vol. 11, no. 2, pp. 141–159, 2003.

[71] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[72] H. Chen and F. Allgöwer, "A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability," *Automatica*, vol. 34, no. 10, pp. 1205–1217, 1998.

[73] G. De Nicolao, L. Magni, and R. Scattolini, "Stability and robustness of nonlinear receding horizon control," in *Nonlinear model predictive control*, 2000, pp. 3–22.

[74] W.-H. Chen, D. J. Ballance, and J. O'Reilly, "Model predictive control of nonlinear systems: Computational burden and stability," in *IET Control Theory and Applications*, vol. 147, no. 4, 2000, pp. 387–394.

[75] H. Li and W. M. Haddad, "Model predictive control for a multicompartment respiratory system," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 5, pp. 1988–1995, 2013.

[76] M. N. Zeilinger, C. N. Jones, and M. Morari, "Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization," *IEEE Trans. Autom. Control*, vol. 56, no. 7, pp. 1524–1534, 2011.

[77] D. Q. Mayne, M. M. Seron, and S. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219–224, 2005.

[78] A. G. Wills, D. Bates, A. J. Fleming, B. Ninness, and S. R. Moheimani, "Model predictive control applied to constraint handling in active noise and vibration control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 3–12, 2008.

[79] R. Eberhart and Y. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proc. 2001 IEEE Congr. Evolut. Comput.*, Seoul, South Korea, 2001, pp. 81–86.

[80] H. Zhang and Q. Hui, "Modified hybrid multiagent swarm optimization algorithms for mixed-binary nonlinear programming," in *46th Hawaii International Conference on System Sciences*, 2013, pp. 1412–1421.

[81] ——, "Binary multiagent coordination optimization with application to formation control design," in *2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 1968–1975.

[82] A. Pinar, J. Meza, V. Donde, and B. Lesieutre, "Optimization strategies for the vulnerability analysis of the electric power grid," *SIAM Journal on Optimization*, vol. 20, no. 4, p. 1786, 2010.

[83] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing.* Springer Science & Business Media, 2003.

[84] J. Kennedy, J. F. Kennedy, and R. C. Eberhart, *Swarm intelligence.* Morgan Kaufmann, 2001.

[85] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative Co-Evolution With Differential Grouping for Large Scale Optimization," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 3, pp. 378–393, Jun. 2014.

[86] Y. Mei, X. Li, and X. Yao, "Cooperative Coevolution With Route Distance Grouping for Large-Scale Capacitated Arc Routing Problems," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 3, pp. 435–449, Jun. 2014.

[87] B. Kazimipour, X. Li, and A. K. Qin, "Initialization methods for large scale global optimization," in *IEEE Congress on Evolutionary Computation.* IEEE, 2013, pp. 2750–2757.

[88] X. Li and X. Yao, "Cooperatively Coevolving Particle Swarms for Large Scale Optimization," *IEEE Trans. Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, Apr. 2012.

[89] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Parallel Problem Solving from Nature, PPSN XI.* Springer, 2010, pp. 300–309.

[90] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *IEEE Congress on Evolutionary Computation.* IEEE, 2008, pp. 1663–1670.

[91] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation," in *2010 IEEE Congress on Evolutionary Computation (CEC).* IEEE, 2010, pp. 1–9.

[92] ——, "Constrained optimization by ε constrained particle swarm optimizer with ε-level control," in *Soft Computing as Transdisciplinary Science and Technology.* Springer Berlin Heidelberg, 2005, pp. 1019–1029.

[93] E. Mezura-Montes and C. A. Coello Coello, "Constraint-handling in nature-inspired numerical optimization: past, present and future," *Swarm and Evolutionary Computation*, vol. 1, no. 4, pp. 173–194, 2011.

[94] L. Zhao and B. Zeng, "Vulnerability analysis of power grids with line switching," *IEEE Trans. Power Systems*, vol. 28, no. 3, pp. 2727–2736, Aug. 2013.

[95] C. Grigg, et al, "The IEEE Reliability Test System-1996. A report prepared by the Reliability Test System Task Force of the Application of Probability Methods Subcommittee," *IEEE Trans. Power Systems*, vol. 14, no. 3, pp. 1010–1020, Aug. 1999.

[96] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature—PPSN III.* Springer, 1994, pp. 249–257.

[97] F. van den Bergh and A. P. Engelbrecht, "A Cooperative approach to particle swarm optimization," *Evolutionary Computation, IEEE Trans.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[98] "Competition & special session on constrained real-parameter optimization," http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC10-Const/CEC10-Const.htm, accessed: 2015-04-22.

[99] R. Mallipeddi and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2010 competition on constrained real-parameter optimization," *Nanyang Technological University, Singapore*, 2010.

[100] Q. Hui and H. Y. Zhang, "Optimal balanced coordinated network resource allocation using swarm optimization," *IEEE Trans. Systems, Man, and Cybernetics: Systems*, vol. 45, no. 5, pp. 770–787, 2015.

[101] Y.-F. Li, G. Sansavini, and E. Zio, "Non-dominated sorting binary differential evolution for the multi-objective optimization of cascading failures protection in complex networks," *Reliability Engineering & System Safety*, vol. 111, pp. 195–205, 2013.

[102] J. Brest, "Constrained real-parameter optimization with ε-self-adaptive differential evolution," in *Constraint-Handling in Evolutionary Optimization*, ser. Studies in Computational Intelligence, E. Mezura-Montes, Ed. Springer Berlin Heidelberg, 2009, vol. 198, pp. 73–93.

[103] S, Zeng, et al, "A lower-dimensional-search evolutionary algorithm and its application in constrained optimization problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 1255–1260.

[104] "Tetsuyuki Takahama's Home Page," http://www.ints.info.hiroshima-cu.ac.jp/~takahama/eng/index.html, accessed: 2015-04-22.

[105] J. Kang, L. Min, Q. Luan, X. Li, and J. Liu, "Novel modified fuzzy c-means algorithm with applications," *Digital Signal Processing*, vol. 19, no. 2, pp. 309–319, 2009.

[106] Z. Yang, F.-L. Chung, and W. Shitong, "Robust fuzzy clustering-based image segmentation," *Applied Soft Computing*, vol. 9, no. 1, pp. 80–84, 2009.

[107] M. Gong, Y. Liang, J. Shi, W. Ma, and J. Ma, "Fuzzy c-means clustering with local information and kernel metric for image segmentation," *Image Processing, IEEE Transactions on*, vol. 22, no. 2, pp. 573–584, 2013.

[108] L. Zhu, F.-L. Chung, and S. Wang, "Generalized fuzzy c-means clustering algorithm with improved fuzzy partitions," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 39, no. 3, pp. 578–591, 2009.

[109] C.-H. Wang and J.-G. Horng, "Constrained minimum-time path planning for robot manipulators via virtual knots of the cubic b-spline functions," *Automatic Control, IEEE Transactions on*, vol. 35, no. 5, pp. 573–577, 1990.

[110] L. Piegl and W. Tiller, *The NURBS book*, 2012.

[111] C.-H. Wang, W.-Y. Wang, T.-T. Lee, and P.-S. Tseng, "Fuzzy b-spline membership function (bmf) and its applications in fuzzy-neural control," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 25, no. 5, pp. 841–851, 1995.

[112] S. M. LaValle, *Planning algorithms*, 2006.

[113] ——, "Motion planning," *Robotics & Automation Magazine, IEEE*, vol. 18, no. 1, pp. 79–89, 2011.

[114] J. K. Tar, I. J. Rudas, and K. R. Kozłowski, "Fixed point transformations-based approach in adaptive control of smooth systems," in *Robot Motion and Control 2007*, 2007, pp. 157–166.

[115] F. Aurenhammer, "Voronoi diagramsa survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 345–405, 1991.

[116] C. Ó'Dúnlaing and C. K. Yap, "A retraction method for planning the motion of a disc," *Journal of Algorithms*, vol. 6, no. 1, pp. 104–111, 1985.

[117] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, vol. 2, no. 1-4, pp. 153–174, 1987.

[118] L. Jin, D. Kim, L. Mu, D.-S. Kim, and S.-M. Hu, "A sweepline algorithm for euclidean voronoi diagram of circles," *Computer-Aided Design*, vol. 38, no. 3, pp. 260–272, 2006.

[119] M. Zavershynskyi and E. Papadopoulou, "A sweepline algorithm for higher order voronoi diagrams," in *2013 10th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD)*, 2013, pp. 16–22.

[120] P. Bhattacharya and M. L. Gavrilova, "Voronoi diagram in optimal path planning," in *Voronoi Diagrams in Science and Engineering, 2007. ISVD'07. 4th International Symposium on*, 2007, pp. 38–47.

[121] H. Sakahara, Y. Masutani, and F. Miyazaki, "Real-time motion planning in unknown environment: Voronoi-based strrt (spatiotemporal rrt)," in *SICE Annual Conference, 2008*, 2008, pp. 2326–2331.

[122] S. Honiden, M. E. Houle, C. Sommer, and M. Wolff, "Approximate shortest path queries using voronoi duals," in *Transactions on computational science IX*, 2010, pp. 28–53.

[123] S. Schwertfeger and A. Birk, "A short overview of recent advances in map evaluation," in *Safety, Security, and Rescue Robotics (SSRR), 2012 IEEE International Symposium on*, 2012, pp. 1–2.

[124] B. Tovar, R. Murrieta-Cid, and C. Esteves, "Robot motion planning for map building," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 1, 2002, pp. 673–680.

[125] S. T. Pfister, S. Roumeliotis, J. W. Burdick *et al.*, "Weighted line fitting algorithms for mobile robot map building and efficient data representation," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 1, 2003, pp. 1304–1311.

[126] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart, "A comparison of line extraction algorithms using 2d laser rangefinder for indoor mobile robotics," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 1929–1934.

[127] J. Elseberg, R. T. Creed, and R. Lakaemper, "A line segment based system for 2d global mapping," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 3924–3931.

[128] G. A. Borges and M.-J. Aldon, "Line extraction in 2d range images for mobile robotics," *Journal of intelligent and Robotic Systems*, vol. 40, no. 3, pp. 267–297, 2004.

[129] Z. Liu, P. Smith, T. Park, Q. Hui *et al.*, "Novel response surface methodologies with design of experiment for source localization in unknown spatial-temporal fields," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, 2014, pp. 5724–5729.

[130] Robotic, *AmigoBOT Users Guide*, 2003.

[131] G. Rote, "Computing the minimum hausdorff distance between two point sets on a line under translation," *Information Processing Letters*, vol. 38, no. 3, pp. 123–127, 1991.

[132] G. Dudek and M. Jenkin, *Computational principles of mobile robotics*, 2010.

[133] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf, *Computational geometry*, 2000.