Evolutionary Computation for Dynamic Optimization Problems

Shengxiang Yang

Head, Innovative Artificial Intelligence (IAI) Cluster
Digital Future Institute
School of Computer Science and Informatics
De Montfort University, Leicester LE1 9BH, UK
http://www.tech.dmu.ac.uk/~syang

Innovative Artificial Intelligence (IAI), DMU

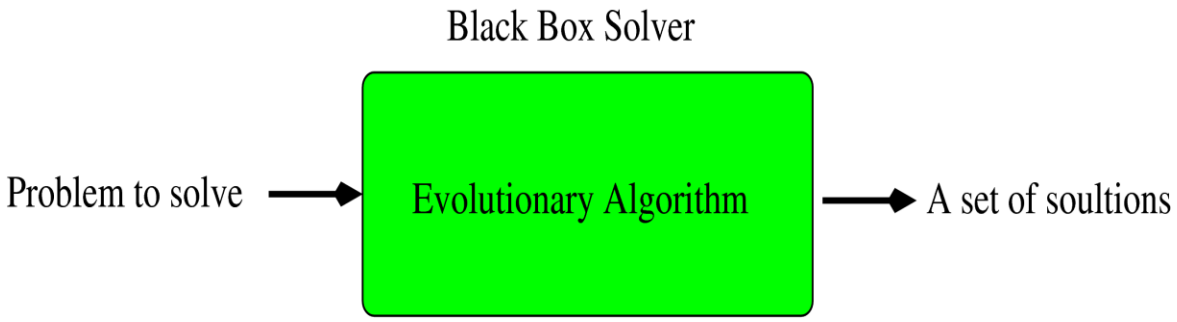
- IAI (https://www.dmu.ac.uk/research/centres-institutes/iai):
  - Mission: Developing fundamental theoretical and practical solutions to real-world problems using a variety of AI paradigms
  - Members: 20+ staff, research fellows, ~30 PhDs, visiting researchers
  - Themes: EC, fuzzy logic, neural networks, data mining, computer vision, game, health, bio-informatics, …
- Funding:
  - Research Councils/Charities: EPSRC, ESRC, EU FP7 & Horizon 2020, Royal Society, Royal Academy of Eng., Innovate UK, KTP, Nuffield Trust …
  - Government: Leicester City Council, DTI
  - Industries: Lachesis, EMDA, RSSB, Network Rail, etc.
- Collaborations:
  - Universities: UK, USA, Spain, and China
  - Industries and local governments
- Teaching/Training:
  - DTP-IS: University Doctor Training Programme in Intelligent Systems
  - MSc: Intelligent Systems (& Robotics); Data Analytics; BI & Data Mining
  - BSc: AI with Robotics; Computer Game Programming; Math; Comp Sci

Outline of the Tutorial

- Part I: Fundamentals
  - Evolutionary computation (EC) for dynamic optimization problems (DOPs): Concept and motivation
  - Benchmark and test problems
  - Performance measures
  - EC approaches for DOPs
- Part II: Case Studies, Issues and Future Work
  - Case study: Particle swarm optimization (PSO) for continuous DOPs
  - Case study: Ant Colony Optimization (ACO) for combinatorial DOPs
  - Advanced topics and challenges
  - EDOLAB: introduction
- Summary

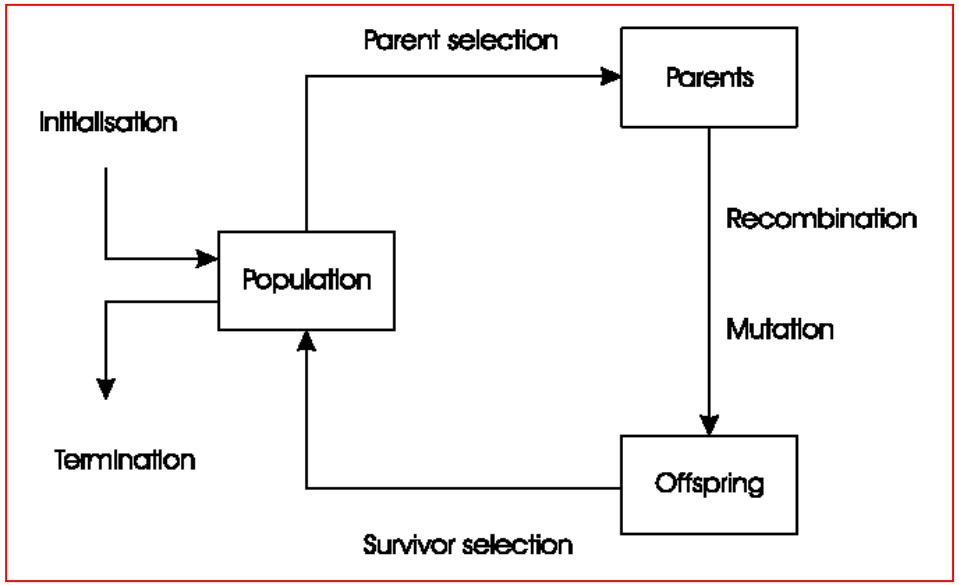
What is Evolutionary Computation (EC)?

- EC encapsulates a class of stochastic optimisation algorithms, dubbed Evolutionary Algorithms (EAs)
- An EA is an optimisation algorithm that is
  - Generic: a black-box tool for many problems
  - Population-based: evolves a population of candidate solutions
  - Stochastic: uses probabilistic rules
  - Bio-inspired: uses principles from biological evolution



Design and Framework of EAs

- Given a problem to solve, key things to consider:
  - Representation of solution into individual
  - Evaluation or fitness function
- EA Framework
  - Initialization of population
  - Evolve the population
    - Selection of parents
    - Variation operators (recombination, mutation)
    - Selection of offspring into next generation
  - Termination condition: a given number of generations

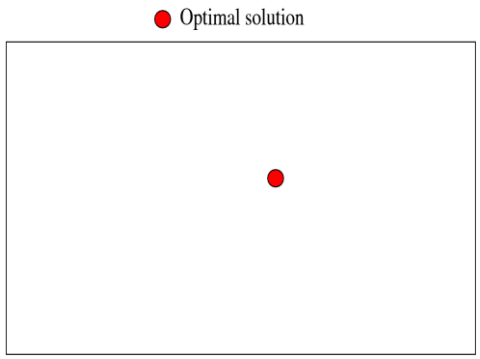


EC for Solving Optimization Problems

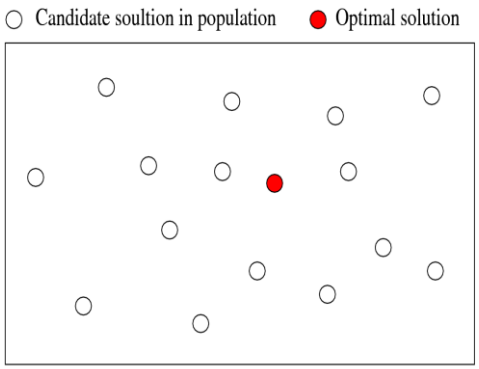
- Easy to use: No strict requirements to problems
- Widely used in different fields:
  - Search-based software engineering
  - Financial and economical systems
  - Transportation and logistics systems
  - Automatic programming, art and music design
  - ……
- Successfully used for different types of problems
  - Stationary vs dynamic optimization problems
  - Single objective vs multi-objective optimization problems
  - Single modal vs multi-modal optimization problems
  - Small scale vs large scale optimization problems
  - Single level vs multi-level optimization problems
  - ……

EC for Dynamic Optimisation Problems: Motivation

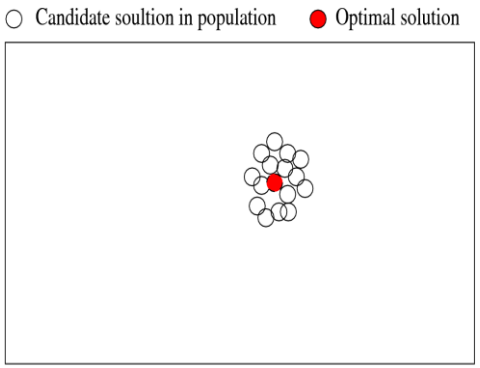
- Traditionally, EAs are mainly applied for static problems
  - Aim: find optimum quickly and precisely in the search space



Search space



Search space (Initial population)



Search Space (Population converging at time t)

- But, many real-world problems are dynamic optimisation problems (DOPs), where changes occur over time
  - Transport systems: travel time between nodes may change
  - Logistics: customer demands may change

What Are DOPs?

- In general terms, “optimisation problems that change over time” are called dynamic problems or time-dependent problems:

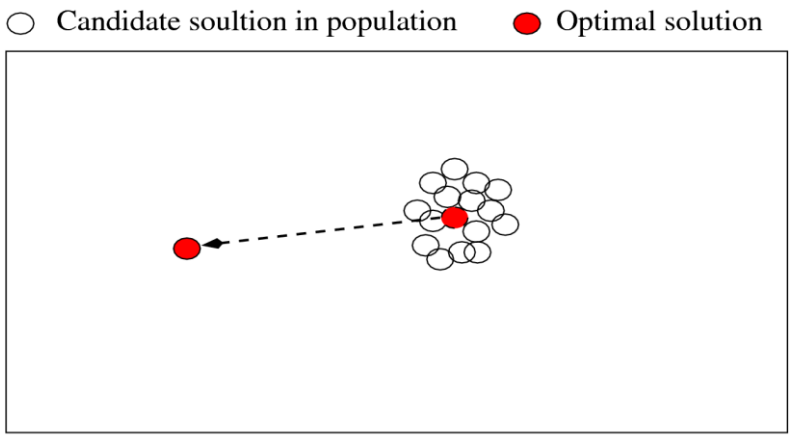
$$F = f(X, S, t)$$

where X: decision variable(s); S: parameters; t: time

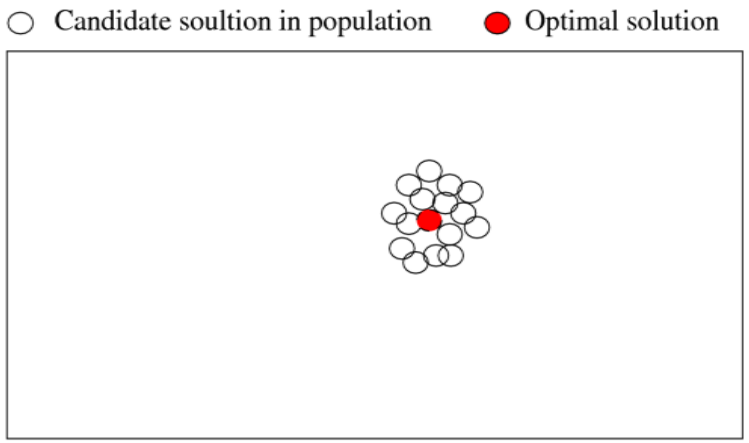
- DOPs: a special class of dynamic problems that are solved online by an algorithm as time goes by

Why DOPs Challenging EC?

- For DOPs, optima may move over time in search space
  - We need to track the moving optima over time
- DOPs challenge traditional EAs
  - Once converged, hard to escape from the old optimum



Search Space (Optimum moved at time t+1)



Search Space (Population converging at time t)

Why EC for DOPs?

- Many real-life problems are DOPs
  - Desirable to present solutions to decision makers over time
- EAs, once properly enhanced, are good choice
  - Inspired by biological behaviour, always in dynamic environments
  - Intrinsically, should be fine to deal with DOPs
- Research on EC for DOPs rises recently
  - Books, PhD Theses
  - Journal special issues
  - Workshops and conference special sessions
  - IEEE Symposium on CIDUE (2011, 2013-2022)
  - IEEE Competitions: within 2009 & 2012 IEEE CEC

Benchmark and Test DOPs

- Basic idea: change base static problem to create DOPs
- Real space:
  - Switch between different functions
  - Move/reshape peaks in the fitness landscape
- Binary space:
  - Switch between ≥ 2 states of a problem: knapsack
  - Use binary masks: XOR DOP generator (Yang & Yao'05)
- Combinatorial space:
  - Change decision variables: item weights/profits in knapsack problems
  - Add/delete decision variables: new jobs in scheduling, nodes added/deleted in network routing problems

Moving Peaks Benchmark (MPB) Problem

- Proposed by Branke (1999)
- The MPB problem in the D-dimensional space:

$$\mathrm{F}(\vec{x}, t) = \max_{i=1,\dots,P} \frac{H_i(t)}{1 + W_i(t) \sum_{j=1}^{j=D} (x_j(t) - X_{ij}(t))^2}$$

  - H_i(t), $W_i(t)$, $X_i(t) = \{X_{i1} \dots X_{iD}\}$: height, width, location of peak *i* at *t*

- The dynamics:

$$H_i(t) = H_i(t-1) + height\_severity * \sigma$$

$$W_i(t) = W_i(t-1) + width\_severity * \sigma$$

$$\vec{V}_i(t) = \frac{s}{\left|\vec{r} + \vec{V}_i(t-1)\right|} ((1-\lambda)\vec{r} + \lambda \vec{V}_i(t-1))$$

$$\vec{X}_i(t) = \vec{X}_i(t-1) + \vec{V}_i(t)$$

  - $\sigma \sim N(0,1)$; λ: correlated parameter
  - $\vec{V}_i$(t): shift vector, which combines random vector $\vec{r}$ and $\vec{V}_i$(t-1) and is normalized to the shift length *s*

Dynamic Traveling Salesman Problem

- Stationary traveling salesman problem (TSP):
  - Given a set of cities, find the shortest route that visits each city once and only once
- Dynamic TSP (DTSP):
  - May involve dynamic cost (distance) matrix

$$D(t) = \{d_{ij}(t)\}_{n*n}$$

$d_{ij}(t)$: cost from city i to j; n: the number of cities

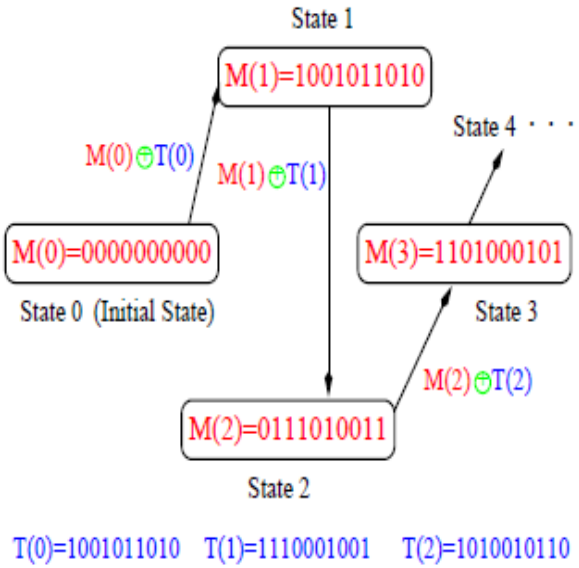
  - The aim: to find a minimum-cost route containing all cities at time t
  - DTSP can be defined as f (x, t):

$$f(x,t) = Min(\sum_{i=1}^{n} d_{x_i,x_{i+1}}(t))$$

where $x_i \in 1, \cdots, n$. If $i \neq j$, $x_i \neq x_j$, and $x_{n+1} = x_1$

The XOR DOP Generator

- The **XOR DOP generator** can create DOPs from any binary $f(\vec{x})$ by an XOR operator "$\oplus$" (Yang, 2003; Yang & Yao, 2005)
- Suppose the environment changes every τ generations
- For each environmental period $k = \lfloor t/\tau \rfloor$, do:



1. Create a template T_k with $\rho * l$ ones
2. Create a mask $\vec{M}(k)$ incrementally

$$\vec{M}(0) = \vec{0} \text{ (the initial state)}$$

$$\vec{M}(k+1) = \vec{M}(k) \oplus \vec{T}(k)$$

3. Evaluate an individual:

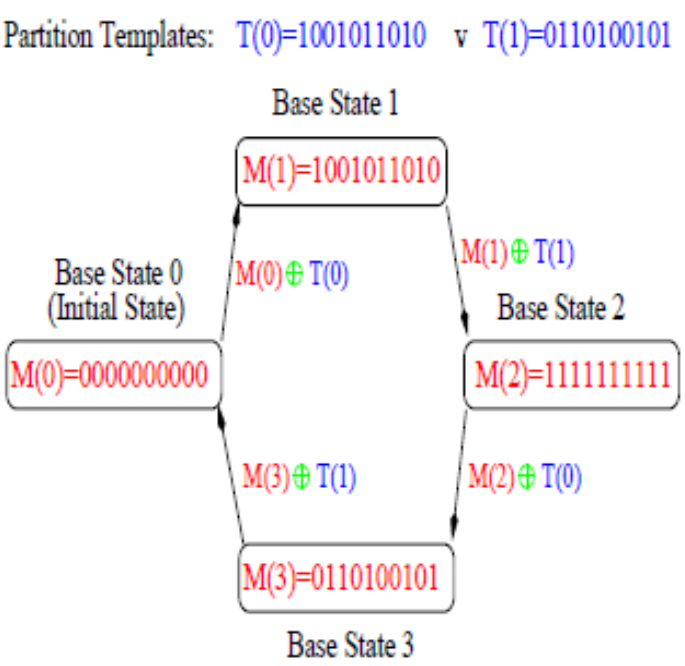
$$f(\vec{x}, t) = f(\vec{x} \oplus \vec{M}(k))$$

- τ and ρ controls the speed and severity of change respectively

S. Yang and X. Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. Soft Computing, 9(11): 815-834, November 2005.

Constructing Cyclic Dynamic Environments

- Extend XOR DOP generator to create cyclic environments



1. Construct K templates $\vec{T}(0),\cdots,\vec{T}(K-1)$
   - Form a partition of the search space
   - Each contains $\rho \times l = l/K$ ones
2. Create $2K$ masks $\vec{M}(i)$ as *base states*

$$\vec{M}(0) = \vec{0} \text{ (the initial state)}$$

$$\vec{M}(i+1) = \vec{M}(i) \oplus \vec{T}(i\%K), i = 0,\cdots,2K-1$$

3. Cycle among $\vec{M}(i)$'s every τ generations

$$f(\vec{x},t) = f(\vec{x} \oplus \vec{M}(I_t)) = f(\vec{x} \oplus \vec{M}(k\%(2K)))$$

- $k = \lfloor t/\tau \rfloor$: environmental index
- $I_t = k\%(2K)$: mask index

S. Yang and X. Yao. Population-based incremental learning with associative memory for dynamic environments. IEEE Transactions on Evolutionary Computation, 12(5): 542-561, October 2008.

Performance Measures

- For EC for static problems, 2 key performance measures:
  - Convergence speed, Success rate of reaching optimality
- For EC for DOPs, over 20 measures (Nguyen et al., 2012)
  - Optimality-based performance measures
    - Collective mean fitness or mean best-of-generation
    - Accuracy
    - Adaptation
    - Offline error and offline performance
    - .....
  - Behaviour-based performance measures
    - Reactivity
    - Stability
    - Robustness
    - Diversity measures
    - .....

T. T. Nguyen, S. Yang, and J. Branke. Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evolutionary Computation, 6: 1-24, October 2012

EC for DOPs: Things to Do

- To detect potential environmental changes
  - Individual-level detection: Re-evaluate individuals' objective values every generation
  - Population-level detection: Check population-related statistical information (i.e., distribution) every generation; Significant change means environment change
  - Both methods could fail to detect changes (not 100% guaranteed)
- To track the changing optima
  - To expect a steady and fast change response
  - To reduce the cost of tracking (given the budget limit, i.e., time, memory)

Response/Enhancing: First Thinking

- Recap: traditional EAs are not good for DOPs
- Goal: to track the changing optimum
- How about restarting an EA after a change?
  - Natural and easy choice
  - But, not good choice because:
    - It may be inefficient, wasting computational resources
    - It may lead to very different solutions before and after a change. For real-world problems, we may expect solutions to remain similar
- Extra approaches are needed to enhance EAs for DOPs

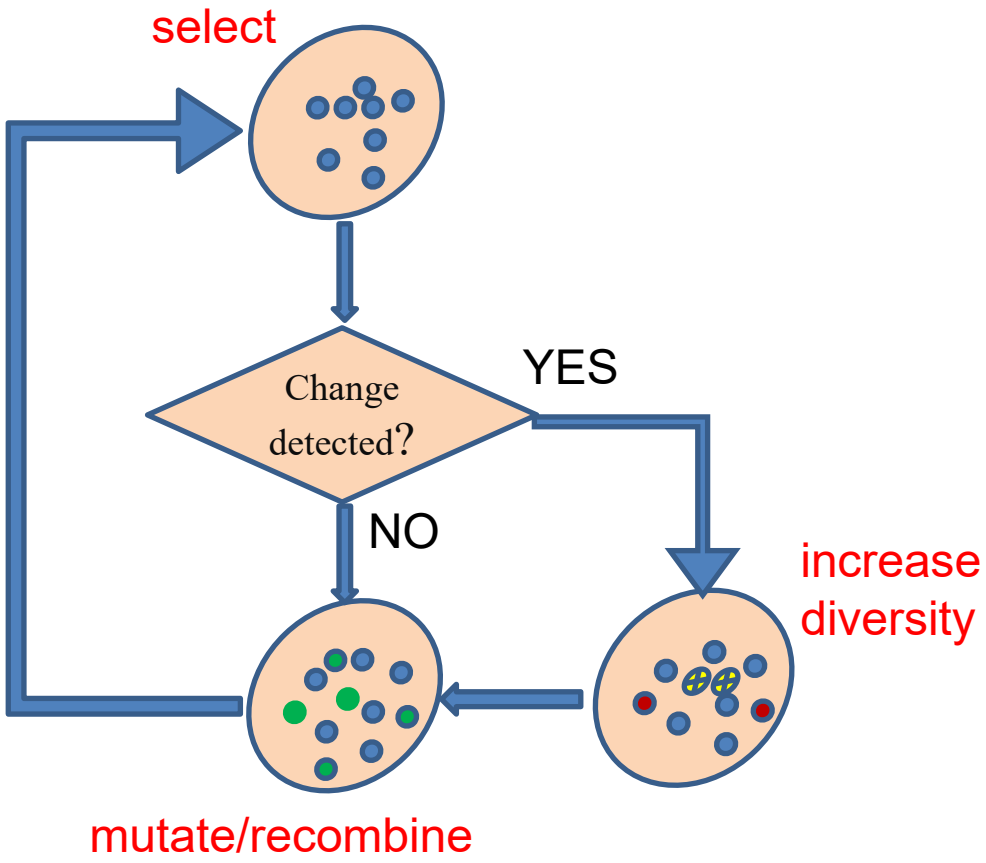
Response/Enhancing: General Approaches

- Many approaches developed to enhance EAs for DOPs
- Typical approaches:
  - Diversity: handle convergence directly
  - Memory: store and reuse useful information
  - Multi-population: co-operate sub-populations
  - Adaptive: adapt generators and parameters
  - Prediction: predict changes and take actions in advance
  - Hybridization: use hybridization tech to improve EAs for DOPs

- D. Yazdani, R. Cheng, D. Yazdani, J. Branke, Y. Jin, and X. Yao. A survey of evolutionary continuous dynamic optimization over two decades—Part A. IEEE Trans. on Evol. Comput., 25(4): 609-629, 2021.
- D. Yazdani, R. Cheng, D. Yazdani, J. Branke, Y. Jin, and X. Yao. A survey of evolutionary continuous dynamic optimization over two decades—Part B. IEEE Trans. on Evol. Comput., 25(4): 630-650, 2021.
- M. Mavrovouniotis, C. Li, and S. Yang. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. Swarm and Evolutionary Computation, 33: 1-17, April 2017
- T. T. Nguyen, S. Yang, and J. Branke. Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evolutionary Computation, 6: 1-24, October 2012

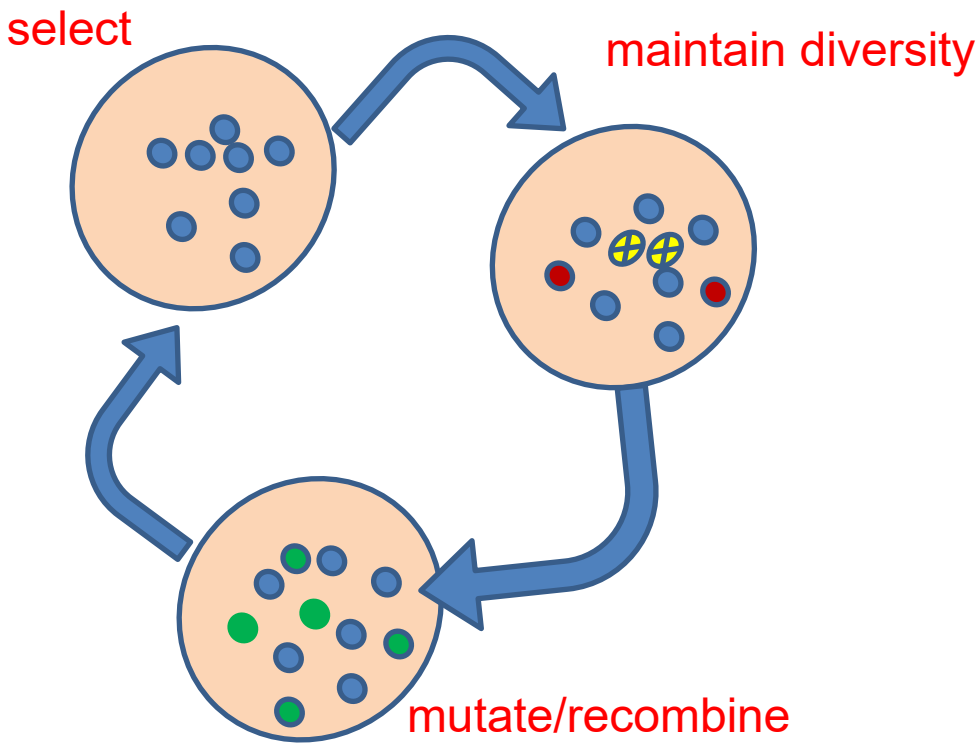
Diversity Approaches

- Diversity increase: introduce diversity after a change
  - Partially random restart, hyper-mutation, variable local search



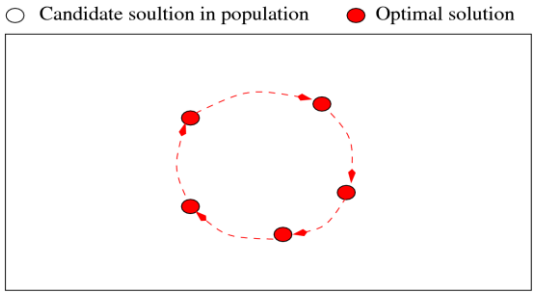
Diversity Approaches

- Diversity maintenance: maintain diversity throughout the run (even if no change occurs)
  - Random immigrants



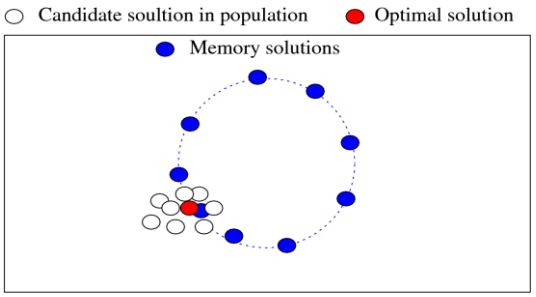
Memory Approaches

- Cyclic DOPs: change cyclically among a fixed set of states

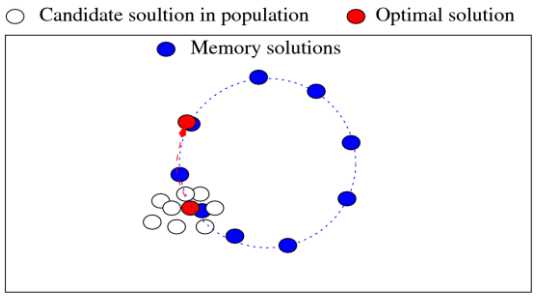


Search space (Optimum moves cyclically)

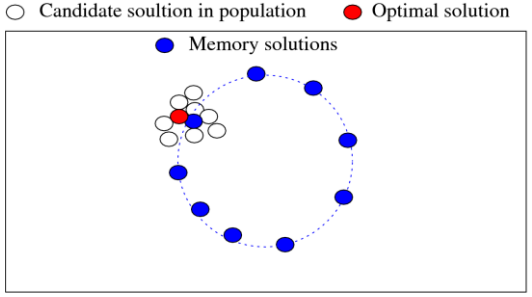
- For cyclic DOPs, use memory to store & reuse good solutions
  - With time, store the best solution of population into memory
  - When a change occurs, memory helps to track new optimum



Search space (Memory stores best solutions)



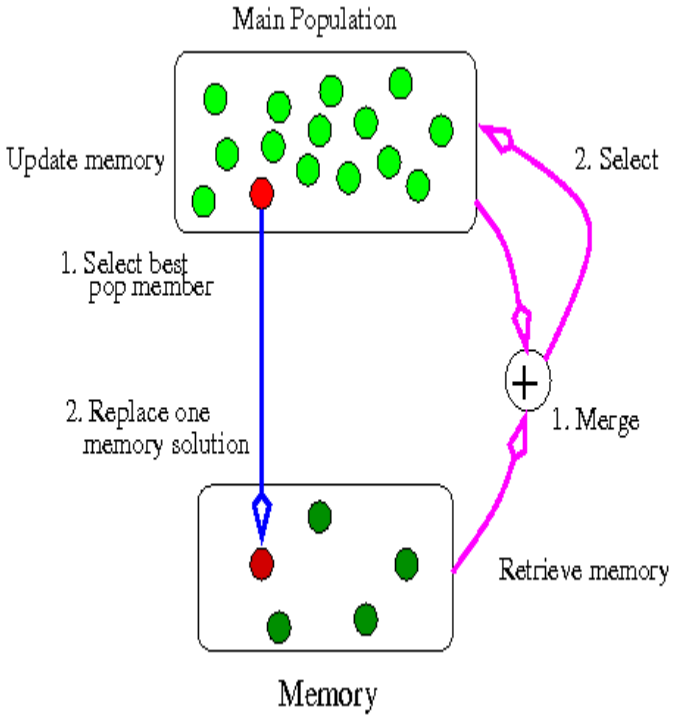
Search space (Optimum moves to next state)

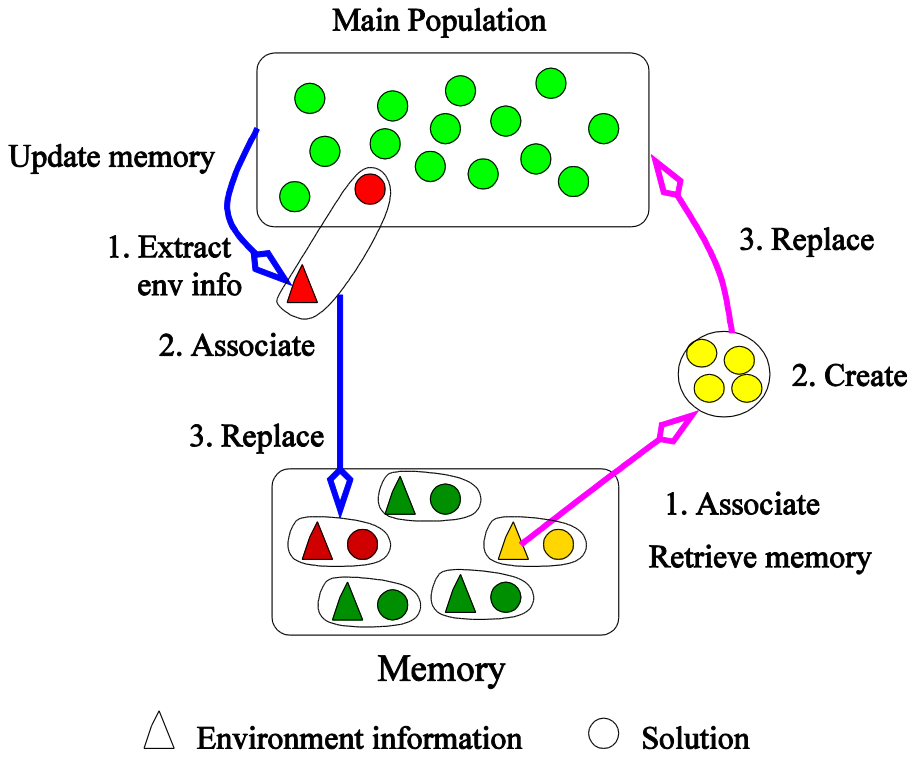


Search space (Population moves to new optimum)

Direct Memory vs Associative Memory

- Direct memory: store good solutions (Branke, CEC'99)
- Associative memory: store environmental information + good solutions

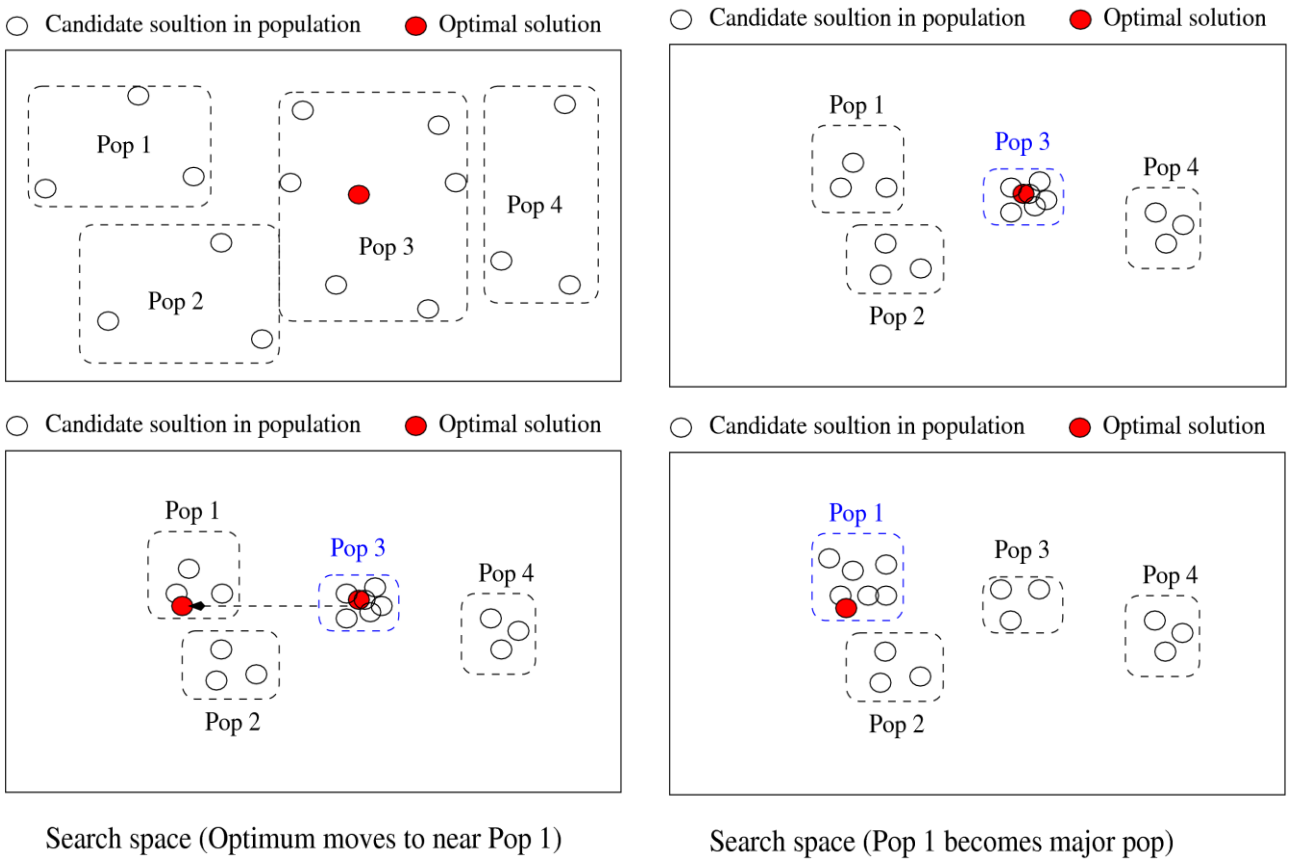




S. Yang and X. Yao. Population-based incremental learning with associative memory for dynamic environments. IEEE Transactions on Evolutionary Computation, 12(5): 542-561, October 2008

Multi-population Approaches

- Idea: Use several cooperative populations
  - Populations evolve independently in different areas of search space
  - Populations exclude each other to avoid overlap
  - When optimum moves, nearby population will take action

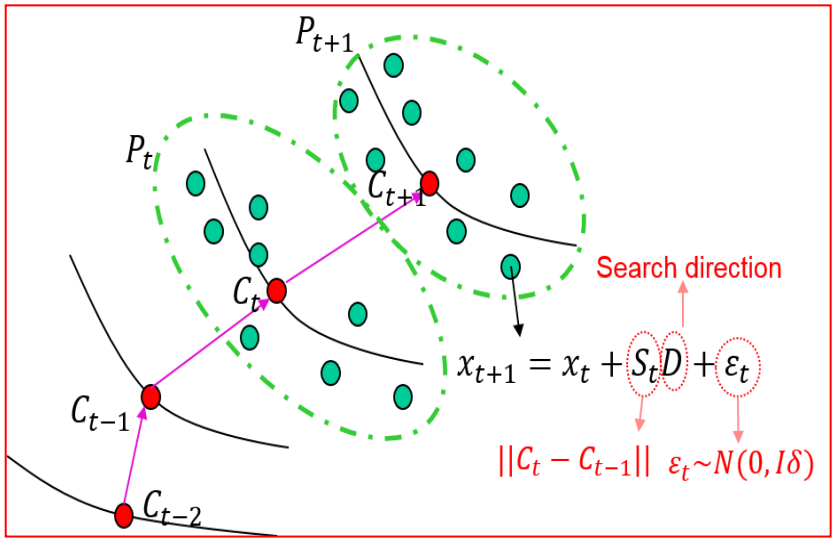


Search space (Optimum moves to near Pop 1)

Search space (Pop 1 becomes major pop)

Prediction Approaches

- For some DOPs, changes exhibit predictable patterns
- Often to predict:
  - The location of new optima after a change
  - When the next change may occur
  - Which environment may appear



- Techniques:
  - Kalman filter (Muruganantham et al. 2016)
  - Population prediction strategy (Zhou et al. 2014)
  - Feed-forward prediction (Hatzakis & Wallace 2006)
  - Directed search strategy (Wu et al. 2015)
  - Evolutionary gradient search (Koo et al. 2010)

Adaptive Approaches

- Aim: Adapt operators/parameters, usually after a change
  - Hypermutation (Cobb & Grefenstette'93): raise the mutation rate temporarily
  - Hyper-selection (Yang & Tinos'08): raise the selection pressure temporarily
  - Hyper-learning (Yang & Richter'09): raise the learning rate for Population-Based Incremental Learning (PBIL) temporarily

- H.G. Cobb, J.J. Grefenstette (1993). Genetic algorithms for tracking changing environments. Proc. ICGA, 523-530.
- S. Yang, R. Tinos (2008). Hyper-selection in dynamic environments. CEC'08, pp. 3185–3192.
- S. Yang, H. Richter (2009). Hyper-learning for population-based incremental learning in dynamic environments. CEC'09, pp. 682–689.

Hybridization Approaches

- Idea: Using hybridization tech to improve the performance of EC for DOPs
- Hybridize EC with local search + diversity schemes, e.g.:
  - P-ACO: Hybridize ACO with local search and random immigrants
  - Multi-strategy ensemble PSO (MEPSO): Hybridize PSO with Gaussian local search + differential mutation
- Hybridize EC with other meta-heuristic methods
  - PSO + Cellular Aotomata
  - PSO + Fuzzy C-means

- M. Mavrovouniotis, S. Yang, A memetic ant colony optimization algorithm for the dynamic travelling salesman problem, Soft Comput. 15 (7) (2011) 1405–1425
- W. Du, B. Li, Multi-strategy ensemble particle swarm optimization for dynamic optimization, Inf. Sci. 178 (15) (2008) 3096–3109
- A. Hashemi, M. Meybodi, A multi-role cellular PSO for dynamic environments, in: 14th International Computer Conference (CSICC 2009), 2009, pp. 412–417
- M. Kamosi, A. Hashemi, M. Meybodi, A hibernating multi-swarm optimization algorithm for dynamic environments, in: 2010 2nd World Congress on Nature and Biologically Inspired Computing, 2010, 363–369

Remarks on Enhancing Approaches

- No clear winner among the approaches
- Memory is efficient for cyclic environments
- Multi-population is good for multimodal problems
  - Able to maintain diversity
  - The search ability will decrease if too many sub-populations
- Diversity schemes are usually useful
  - Guided immigrants may be more efficient
- Thumb of rule: balancing exploration & exploitation over time

Case Study: PSO for Continuous DOPs

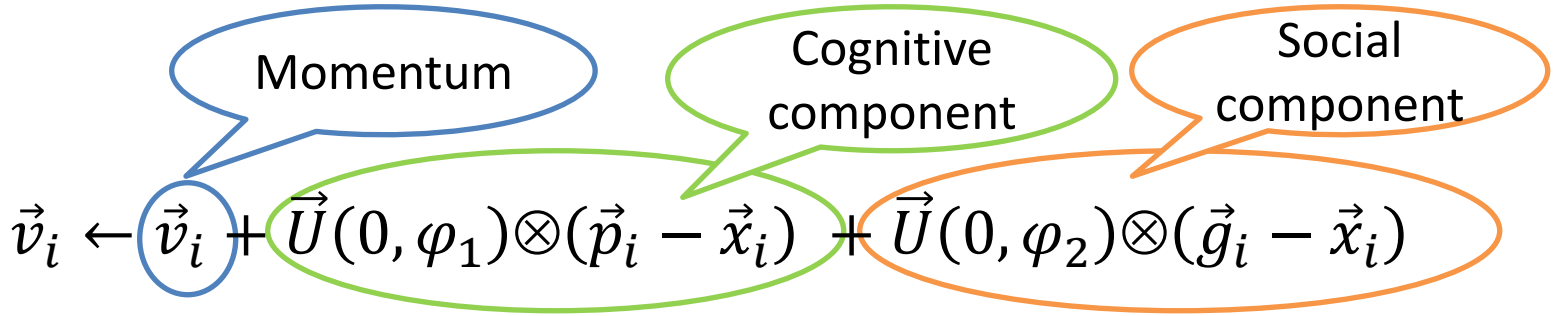
- Particle Swarm Optimization (PSO): Developed by Kennedy and Eberhart (1995)
- A population-based optimization technique inspired by social behaviour of bird flocking or fish schooling
- Swarm members can profit from their own discovery and previous experience of all other members of the school



Kennedy, J. and Eberhart, R.: Particle Swarm Optimization. Proceedings of the Fourth IEEE International Conference on Neural Networks, Perth, Australia. IEEE Service Center 1942-1948, 1995.

Particle Swarm Optimization (PSO)

- PSO consists of a **swarm** of **particles**
- Each particle resides at a **position** in the search space and flies over the search space with a certain **velocity**
- The **velocity** of each particle is influenced by
  - **Momentum**: maintaining previous velocity it has travelled so far
  - **Cognitive component**: returning to the best position visited so far
  - **Social component**: moving to the best position found by neighbors so far



- Eventually the swarm will converge to optimal positions

PSO for Continuous DOPs: Issues

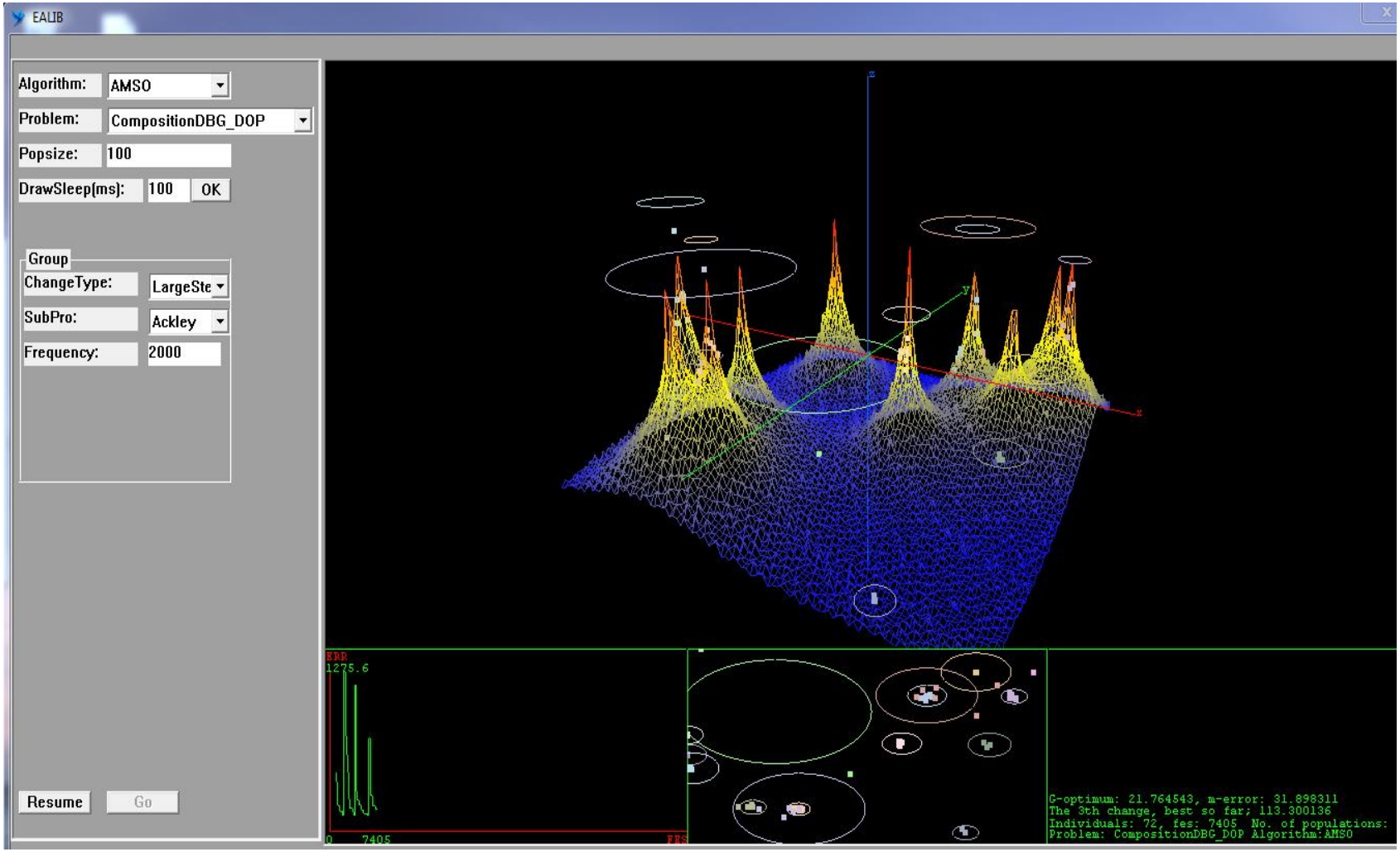
- PSO has been applied for many static problems
- Recently, PSO has been applied for continuous DOPs
- Two aspects to consider:
  - Outdated memory. Two solutions:
    - Simply set *pbest* to the current position
    - Reevaluate *pbest* and reset it to current position if it is worse than the current position
  - Diversity loss. Three solutions:
    - Introduce diversity after a change
    - Maintain diversity during the run
    - Use multi-swarms

Case Study: Multi-swarm PSO for Continuous DOPs

- Recently, a framework of multi-population approaches
  - Use single linkage hierarchical clustering to create populations
  - Each population will search one peak in the fitness landscape
  - An overcrowding scheme to remove unnecessary populations
  - A special rule to decide proper moments to increase diversity without change detection
  - An adaptive method to create a proper number of populations needed

- C. Li and S. Yang. A general framework of multi-population methods with clustering in undetectable dynamic environments. IEEE Transactions on Evolutionary Computation, 16(4): 556-577, August 2012
- C. Li, S. Yang, and M. Yang. An adaptive multi-swarm optimizer for dynamic optimization problems. Evolutionary Computation, 22(4): 559-594, Winter 2014
- C. Li, T. T. Nguyen, M. Yang, M. Mavrovouniotis, and S. Yang. An adaptive multi-population framework for locating and tracking multiple optima. IEEE Transactions on Evolutionary Computation, 20(4):590-605, 2016

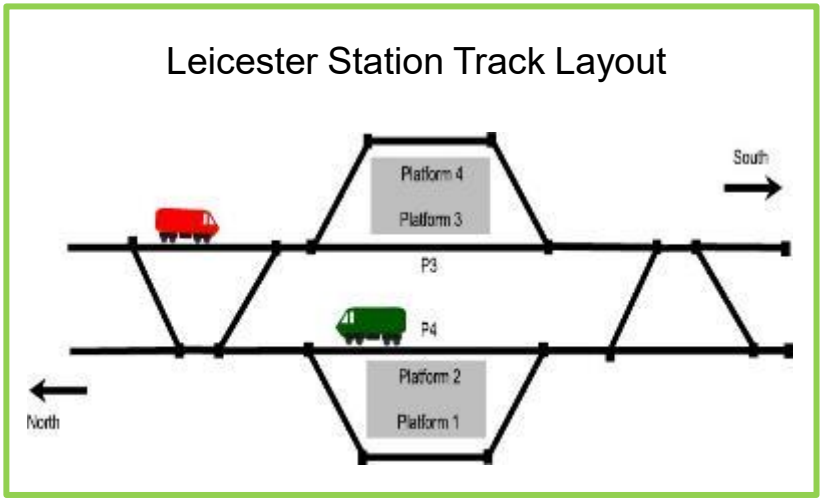
Multi-swarm PSO for DOPs: Demo



Case Study: ACO for Train Platform Reallocation

- A train that arrives late at a station will miss its scheduled time slot and may have to be reallocated to a new platform
- Multiple trains may be delayed in succession, each new delay changes the problem
- Dynamic Railway Platform Reallocation Problem (DRPRP) reallocates multiple successive delayed trains to new timeslots on railway platforms to minimise the ongoing delay in the system

Image source: https://en.wikipedia.org/wiki/Leicester_railway_station

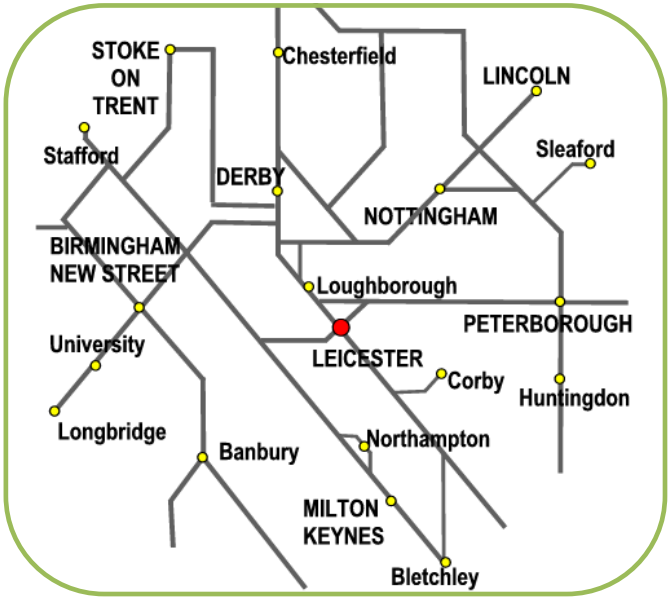


- We considered Leicester station
- A busy UK railway station with 4 bi-directional platforms and trains arriving from 4 different directions
- We consider the effect of the reallocation decisions not only at the station but also on the remainder of these trains' journey

Modelling the Problem

- The model was created from Network Rail's train schedule data from Integrated Train Planning System (ITPS)
- From this we extract details of the movement of trains through the station and the movement of all trains at each of the timing points on each train's route
- We consider timing points within 50 miles of Leicester station (225 timing points)

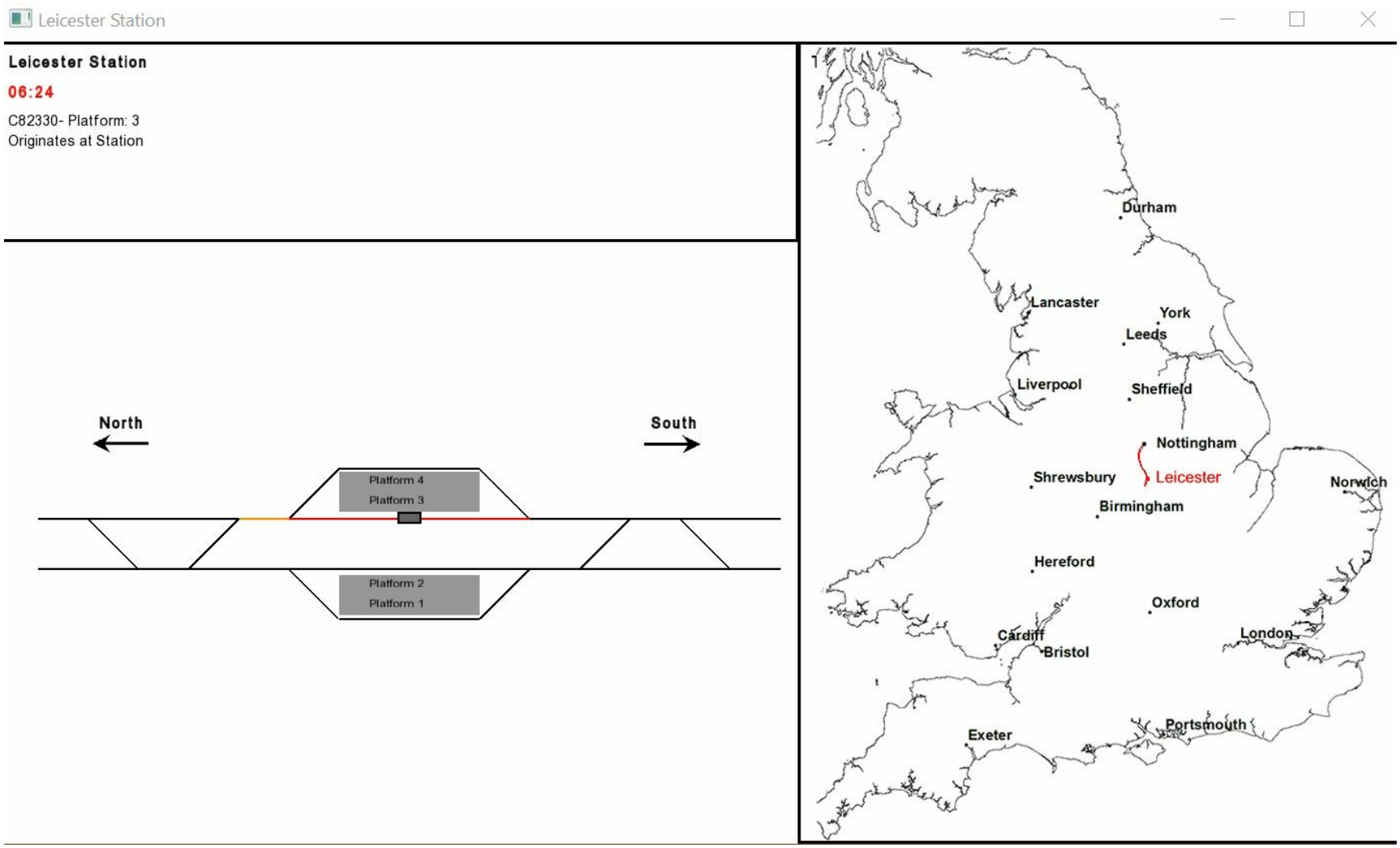
Some timing points in the problem



An example of the schedule feed data

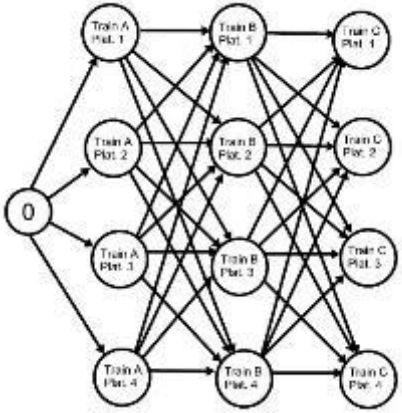
```
H06408,NA,NA,D,600,D,NA,28
LO,RATCFHH,NA,0434,NA,NA,NA,NA,NA,GL
LI,TRENTJ,NA,NA,NA,NA,0437,NA,NA,NA
LI,TRENT,NA,NA,NA,NA,0438,NA,NA,NA
LI,BESTNSJ,NA,NA,NA,NA,0445H,NA,NA,NA
LI,BESTONS,0447H,0516H,NA,NA,NA,NA,NA,NA
LI,BESTNSJ,NA,NA,NA,NA,0518H,NA,NA,NA
LI,TRENT,NA,NA,NA,NA,0523,NA,NA,NA
LI,TRENTJ,NA,NA,NA,NA,0525,NA,NA,GL
LI,RATCLFJ,NA,NA,NA,NA,0527,NA,NA,SL
LI,LOGHBRO,NA,NA,NA,NA,0536,USL,NA,NA
LI,SILEBYJ,NA,NA,NA,NA,0542,NA,NA,NA
LI,SYSTNSJ,NA,NA,NA,NA,0546,NA,NA,NA
LI,LESTER,NA,NA,NA,NA,0554H,3,SL,FL
LI,WGSTNNJ,NA,NA,NA,NA,0600,NA,NA,NA
LI,CROFTS,NA,NA,NA,NA,0606,NA,NA,NA
LI,HINCKLY,NA,NA,NA,NA,0616,NA,NA,NA
LI,NNTN,NA,NA,NA,NA,0626H,7,NA,SL
LI,AMNGTNJ,NA,NA,NA,NA,0637H,NA,SL,SL
LI,LCHTNJ,NA,NA,NA,NA,0646,NA,SL,SL
LI,RUGLYNJ,NA,NA,NA,NA,0653H,NA,SL,SL
LI,COLWICH,NA,NA,NA,NA,0706,NA,SL,FL
LI,MFDB,NA,NA,NA,NA,0708H,NA,NA,SL
LI,STAFTVJ,NA,NA,NA,NA,0715,NA,NA,NA
LI,STAFFRD,NA,NA,NA,NA,0717,5,SL,SL
LI,NTNB,NA,NA,NA,NA,0724,NA,SL,SL
```

Leicester Station Simulation

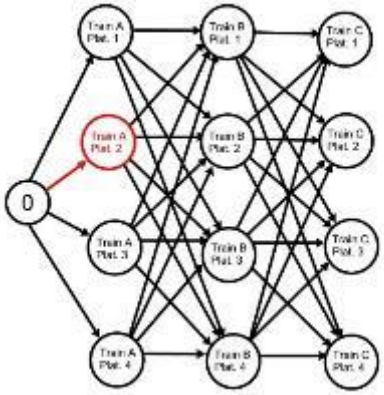


Max-Min Ant System (MMAS)

- In ACO ants communicate indirectly via pheromone trails
- We model the problem with a directed edge graph
- Ants decide which node to choose next based on the values of pheromone trails and problem-specific heuristics

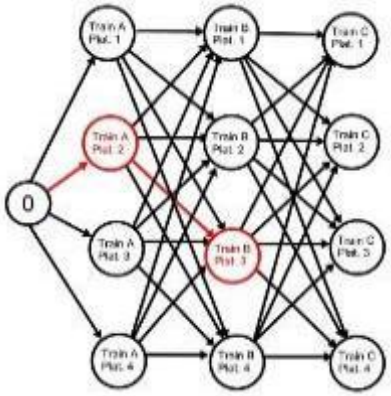


Each node in the graph represents a train and the platform to assign the train to



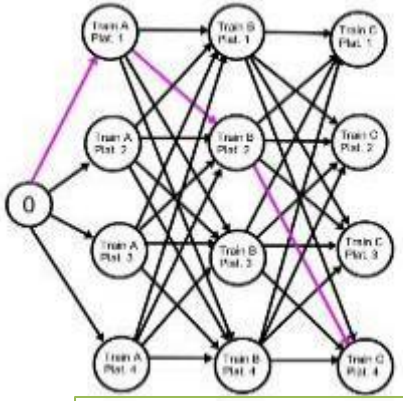
An ant starts on node 0
The ant chooses next node probabilistically

Ant Solution:
<Train A on Platform 2>



The ant now chooses the next train & platform

Ant Solution:
<Train A on Platform 2, Train B on Platform 3>

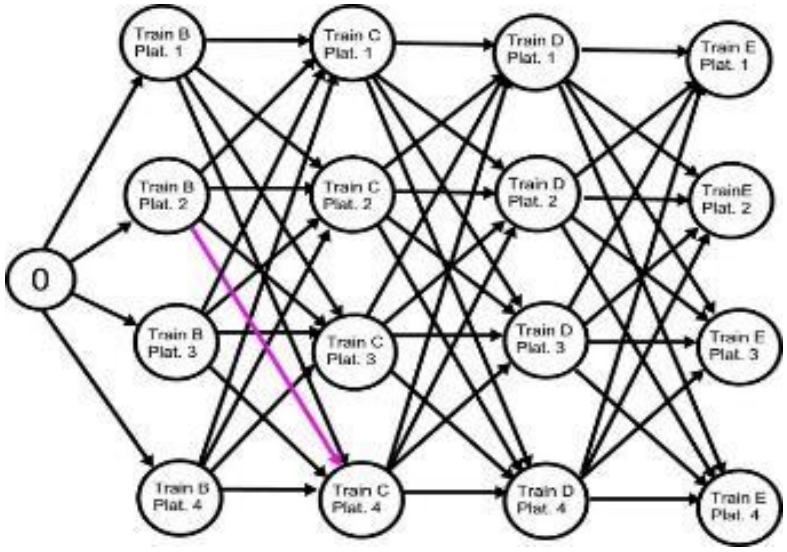


- After all ants have made a tour, all pheromone trails are evaporated
- Pheromone is laid down between the nodes on the best ant's tour

Algorithm Design

After a Dynamic Change:

- More trains have arrived, but some trains have passed through the station
- The graph is updated but pheromones are kept between changes to retain useful information from before change



Unnecessary Platform Reallocation:

- MMAS has no mechanism to persuade it against unnecessarily reallocating trains to platforms. To resolve this, we:
  1. Add a heuristic based on the physical distance between platforms
  2. Introduce a best-so-far ant replacement scheme that discourages unnecessary reallocations of trains to new platforms

Image source: http://www.adelaidenow.com.au//

Comparison Algorithm

First Free Platform (FFP)

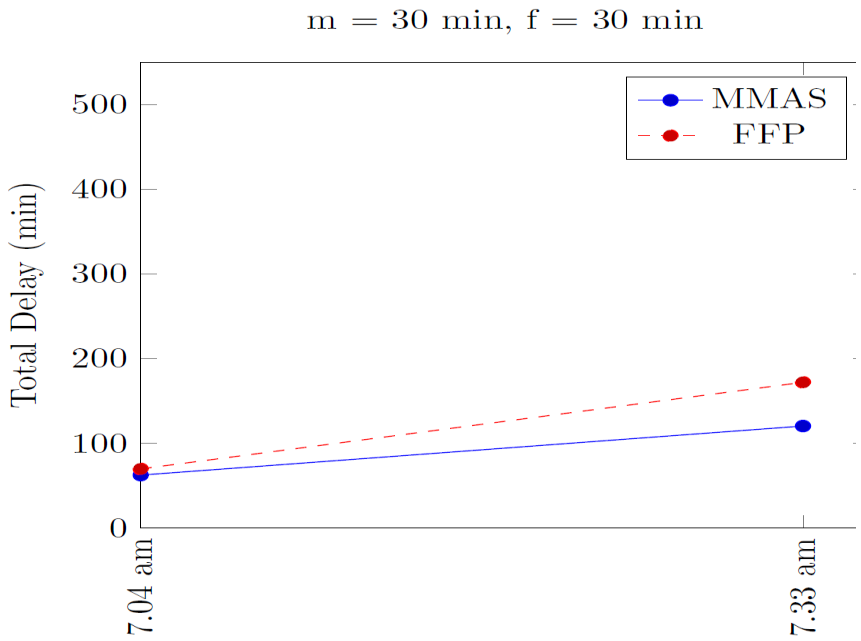
- Discussions with a Network Rail Station Master established that a technique often used to reallocate delayed trains to platforms is to find the first free platform as close as possible to the original platform
- We compared our MMAS algorithm to a heuristic using this principle

Modelling Dynamism

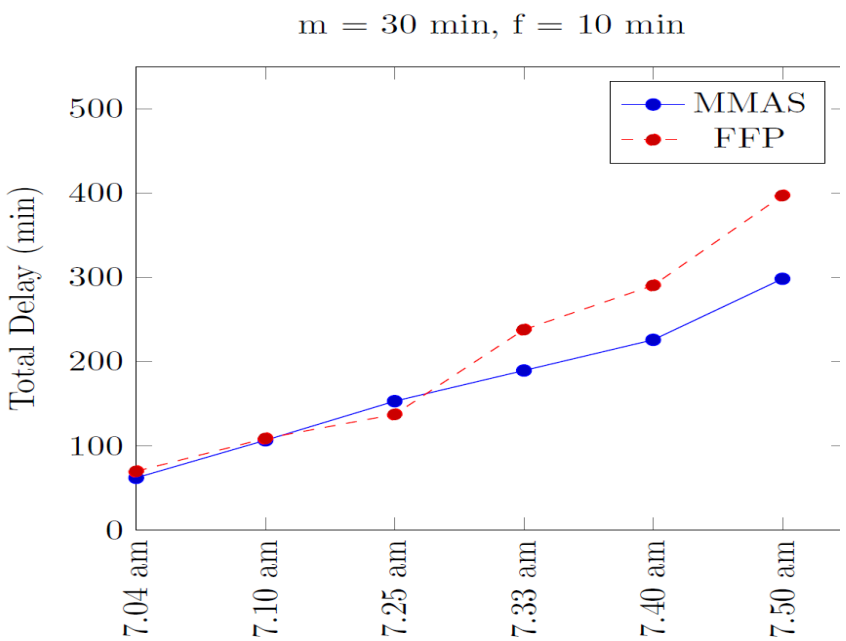
The **frequency of change f** is the time interval between delayed trains.
The **magnitude of change m** is how much the train is delayed by

In this investigation trains were delayed by 10, 20 and 30 min with gaps of 10, 20 and 30 min to give 9 different dynamic scenarios

Experimental Results



Low frequency, high magnitude changes



High frequency, high magnitude changes

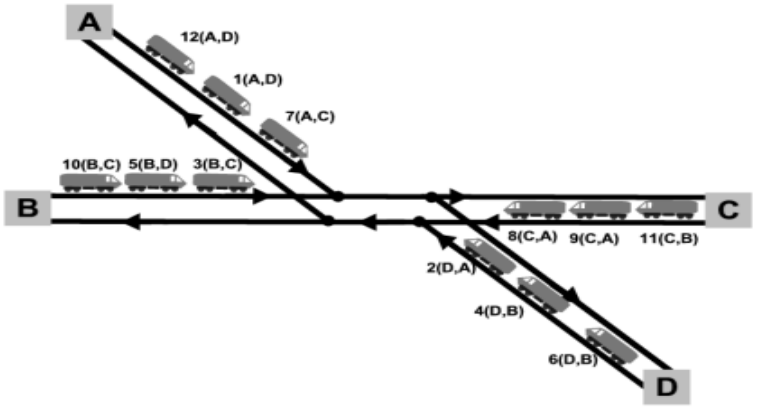
Table 1. One-Sample Wilcoxon Signed Rank Test Results at 0.05 Significance

	m=30			m=20			m=10		
Algorithms	f=10	f=20	f=30	f=10	f=20	f=30	f=10	f=20	f=30
MMAS ⇔ FFP	*s+*	*s+*	*s+*	*s+*	*s+*	*s+*	*s+*	*s+*	*s+*

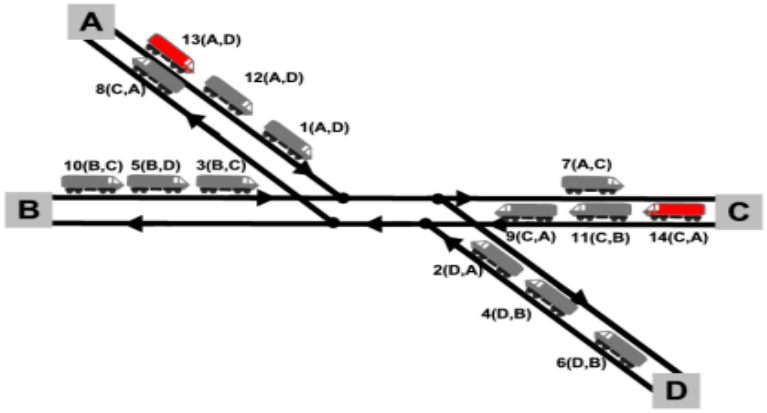
J. Eaton and S. Yang. Railway platform reallocation after dynamic perturbations using ant colony optimisation. Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence, pp. 1-8, 2016

Case Study: ACO for DM-RJRP

- Dynamic multi-objective railway junction re-scheduling problem (DM-RJRP):
  - To find a sequence of trains to pass through two junctions (North Stafford and Stenson) on the Derby to Birmingham line under delays
  - Two objectives:
    - Minimising timetable deviation
    - Minimising additional energy expenditure
  - Dynamic:
    - As trains are waiting to be rescheduled at the junction, more timetabled trains will be arriving, which will change the nature of the problem



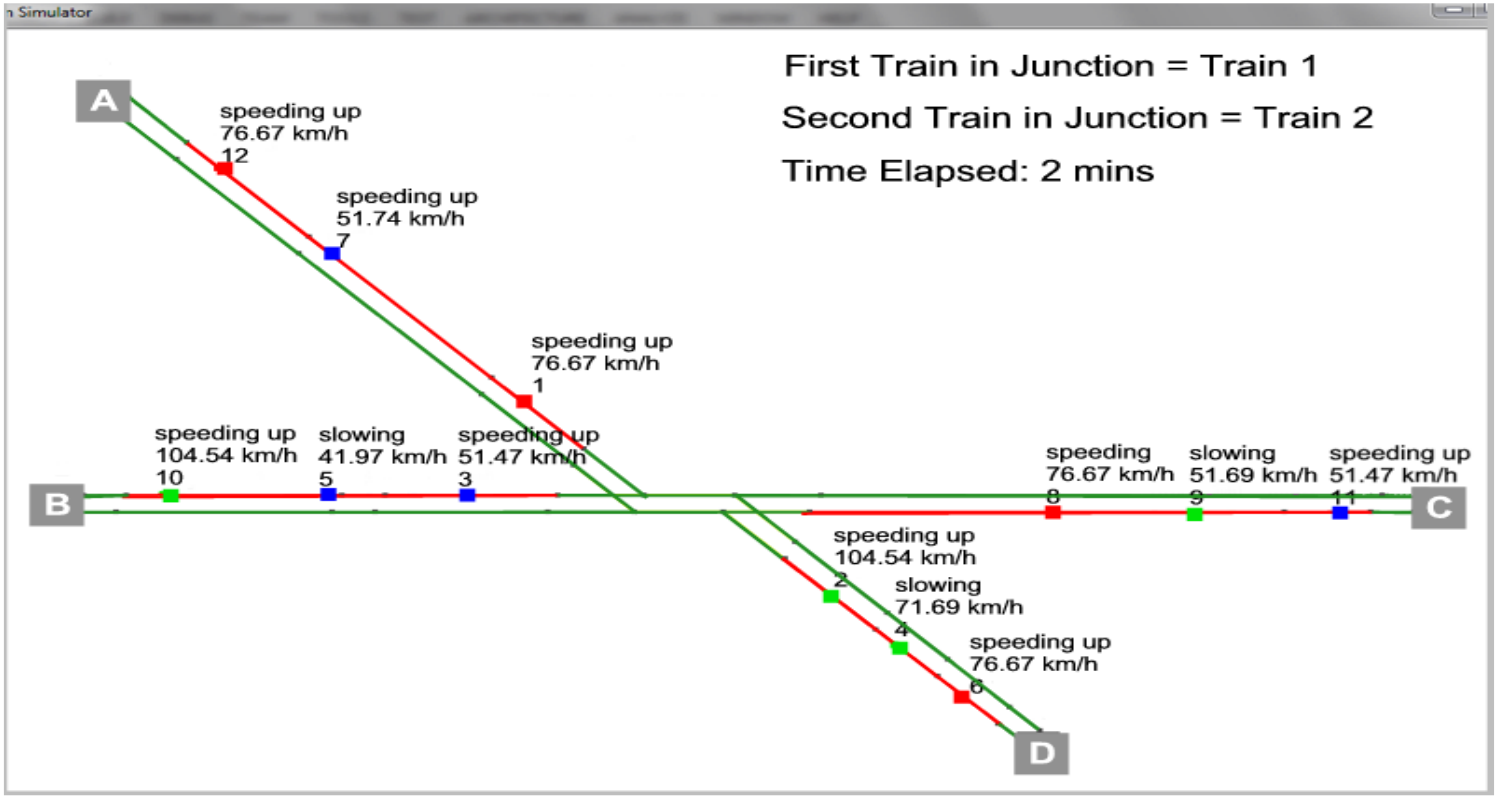
Junction before a change



Junction after a change

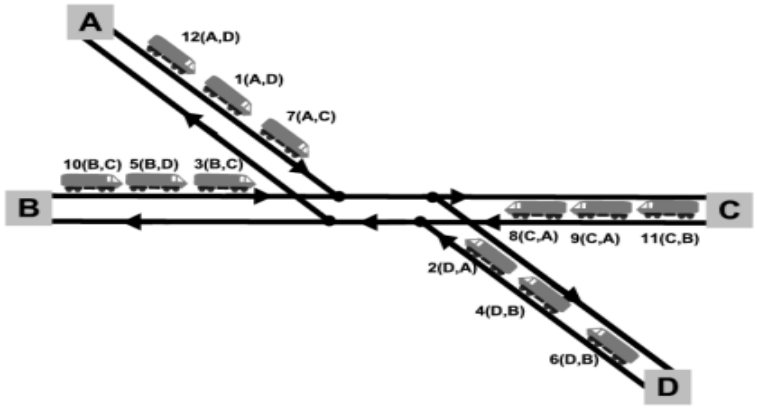
Case Study: ACO for DM-RJRP

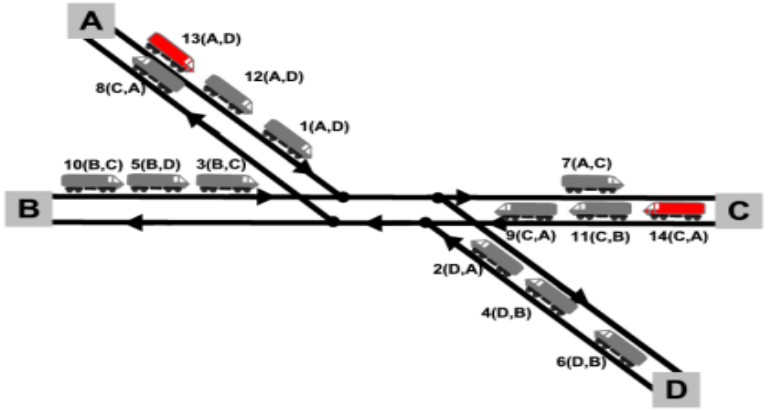
- The North Stafford and Stenson junctions train simulator:
  - Developed using C++ Visual Studio 2012
  - Dynamism:
    - Introduced to the simulator by adding *m* trains at *f* minutes, where *m* and *f* denote the magnitude and frequency of change, respectively

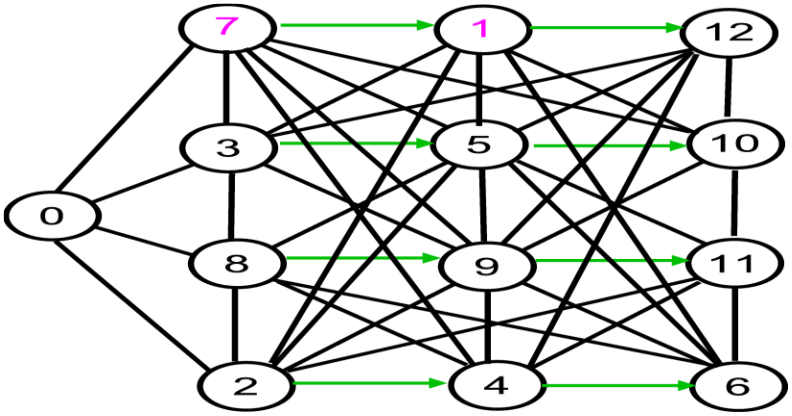


Case Study: ACO for DM-RJRP

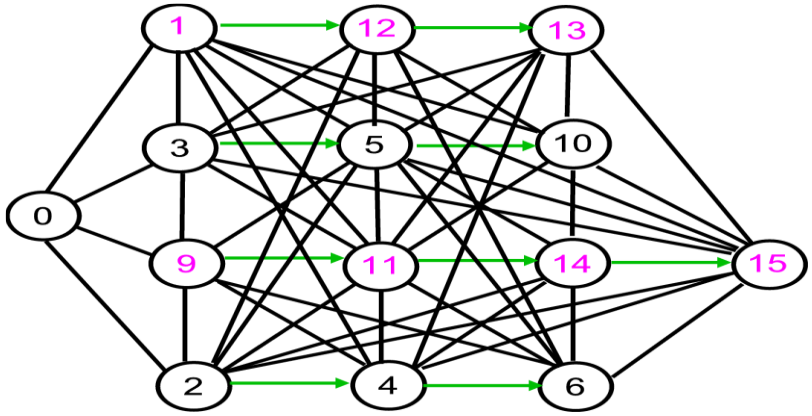
- ACO for DM-RJRP: a graphical representation
  - A fully connected, partially one-directional, weighted graph
  - Each node represents a train
- All ants are initially placed at an imaginary start node (zero)







Node matrix before a change



Node matrix after a change

Case Study: ACO for DM-RJRP

- DM-PACO: a new version of P-ACO for DM-RJRP
  - A pheromone and heuristic matrix for each objective
  - An archive to store non-dominated solutions (repaired after a change)
  - A memory: created from the archive and re-created after a change
- DM-MMAS: a new version of Max-Min Ant System (MMAS)
  - A pheromone matrix for each objective
  - An archive to store non-dominated solutions
  - Four designs based on clearing archive or pheromones after changes

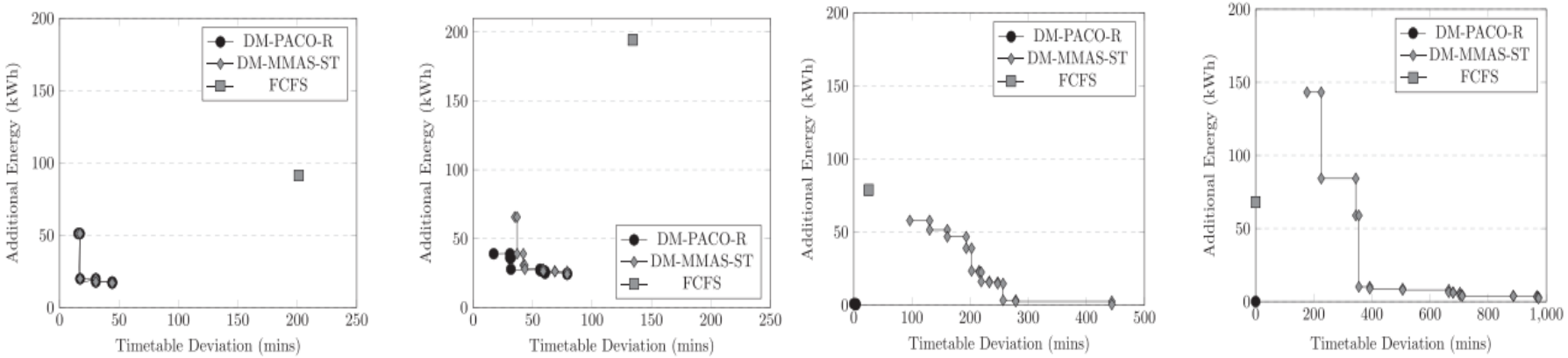
FOUR DIFFERENT VERSIONS OF THE DM-MMAS ALGORITHM

	Clear Pheromones	Retain Pheromones
Clear Archive	DM-MMAS-SC	DM-MMAS-ST
Retain Archive	DM-MMAS-NC	DM-MMAS-NT

- Peer algorithms: NSGA-II and FCFS

Case Study: ACO for DM-RJRP

- Findings:
  - All ACO algorithms can find a POS of solutions for the DM-RJRP
  - DM-PACO outperformed DM-MMAS algorithms
  - DM-PACO also outperformed NSGA-II and FCFS
  - For large and frequent changes:
    - Good to retain an archive of non-dominated solutions
    - Good to update pheromones for new environments
  - Interaction between objectives are more complex than expected



(a) Change 0 (b) Change 1 (c) Change 4 (d) Change 6

J. Eaton, S. Yang, and M. Gongora. Ant colony optimization for simulated dynamic multi-objective railway junction rescheduling. IEEE Transactions on Intelligent Transportation Systems, 18(11): 2980-2992, 2017

Advanced Topics

- EC for DOPs: Theoretical Development
- EC for Dynamic Multi-objective Optimization Problems (DMOPs)
- EC for Dynamic Constrained Optimization Problems (DCOPs)

EC for DOPs: Theoretical Development

- So far, mainly empirical studies. Theoretical analysis has been limited
- Runtime analysis:
  - Stanhope & Daida (1999) first analyzed (1+1) EA on the dynamic bit matching problem (DBMP)
  - Droste (2002) analyzed first hitting time of (1+1) ES on the DBMP
  - Rohlfshagen et al. (2010) analyzed how the magnitude and speed of change may affect the performance of the (1+1) EA on two functions constructed from the XOR DOP generator

- S. Droste. Analysis of the (1+1) EA for a dynamically changing onemax-variant. CEC'02, pp. 55-60, 2002
- S.A. Stanhope, J.M. Daida. (1+1) genetic algorithm fitness dynamics in a changing environnements. Proceedings of the 1999 IEEE Congress on Evol Comput, vol. 3, pp. 1851-1858, 1999.
- P. Rohlfshagen, P.K. Lehre, X. Yao. Dynamic evolutionary optimisation: An analysis of frequency and magnitude of change. GECCO'09, pp. 1713-1720, 2009.

EC for DOPs: Theoretical Development

- Analysis of dynamic fitness landscape:
  - Branke et al. (2005) analyzed the changes of fitness landscape due to changes of the underlying problem instance
  - Richter (2010) analyzed the properties of spatio-temporal fitness landscapes constructed from Coupled Map Lattices (CML)
  - Tinos & Yang (2010, 2014) analyzed properties of the XOR DOP generator based on the dynamical system approach of the GA

- J. Branke, E. Salihoglu, S. Uyar. Towards an analysis of dynamic environments. GECCO'05, pp. 1433-1439, 2005.
- H. Richter. Evolutionary optimization and dynamic fitness landscapes: From reaction-diffusion systems to chaotic cml. Evolutionary Algorithms and Chaotic Systems, Springer, pp. 409-446, 2010.
- R. Tinos, S. Yang. An analysis of the XOR dynamic problem generator based on the dynamical system. PPSN XI, LNCS 6238, Part I, pp. 274-283, 2010.
- R. Tinos, S. Yang. Analysis of fitness landscape modifications in evolutionary dynamic optimization. Inform. Sci., 282: 214-236, 2014.

EC for Dynamic Multi-objective Optimization

- So far, mainly dynamic single-objective optimization
- Dynamic multi-objective optimization problems (DMOPs)
  - Even more challenging
- Recently, rising interest in studying EC for DMOPs
  - Farina et al. (2004) classified DMOPs by changes on Pareto optimal solutions
  - Goh & Tan (2009) proposed a competitive-cooperative coevolutionary algorithm for DMOPs
  - Zeng et al. (2006) proposed a dynamic orthogonal multi-objective EA (DOMOEA) to solve a DMOP with continuous decision variables
  - Jiang & Yang (2017a) proposed a new benchmark MDOP generator
  - Jiang & Yang (2017b) proposed a Steady-Generational EA for DMOPs
  - Eaton et al. (2017) applied ACO for the dynamic multi-objective railway junction rescheduling problem
  - Zhang et al. (2020) proposed novel prediction strategies for DMOPs.
  - Hu et al. (2023) proposed to handle dynamic multi-objective optimization environments via layered prediction and subspace-based diversity maintenance.

- S. Jiang, S. Yang. A steady-state and generational evolutionary algorithm for dynamic multi-objective optimization. IEEE Transactions on Evolutionary Computation, 21(1): 65-82, 2017
- S. Jiang, J. Zou, S. Yang, and X. Yao. Evolutionary dynamic multi-objective optimization: A survey. ACM Computing Survey, 55(4), Article 76, pp. 1-47, April 2023.

EC for Dynamic Constrained Optimization (DCOPs)

- So far, mainly unconstrained DOPs
- Recently, rising interest in studying EC for DCOPs:
  - F. Wang, M. Huang, S. Yang, and X. Wang. Penalty and prediction methods for dynamic constrained multi-objective optimization. Swarm and Evolutionary Computation, 80, Article 101317, July 2023.
  - Q. Chen, J. Ding, Gary G. Yen, S. Yang, T. Chai. Multi-population evolution based dynamic constrained multiobjective optimization under diverse changing environments. *IEEE Trans. Evol. Comput.*, in press, 2023.
  - Y. Wang, J. Yu, S. Yang, S. Jiang, and S. Zhao. Evolutionary dynamic constrained optimization: Test suite construction and algorithm comparisons. Swarm and Evolutionary Computation, 50, Article 100559, 2019.

EC for DOPs: Challenging Issues

- Detecting changes:
  - Most studies assume that changes are easy to detect or visible to an algorithm whenever occurred
  - In fact, changes are difficult to detect for many DOPs
- Understanding the characteristics of DOPs:
  - What characteristics make DOPs easy or difficult?
  - Little work, needs much more effort
- Analysing the behaviour of EC methods for DOPs:
  - Requiring more theoretical analysis tools
  - Big question: Which EC methods for what DOPs?
- Real world applications:
  - How to model real-world DOPs?

EDOLAB - Motivation

- Evolutionary dynamic optimization algorithms (EDOAs) incorporate multiple components and mechanisms, leading to high algorithmic complexity and challenging re-implementation.
- Published descriptions often omit critical implementation details, making re-implementation time-consuming and error-prone.
- Inconsistent handling of changes and random seeds causes unreliable benchmark comparisons.
- **A unified, standardized platform is needed** for fair evaluation, reproducibility, and lowering the entry barrier for EDOA research.

About EDOLAB

- What is EDOLAB?
  - An open-source MATLAB platform for research and teaching in dynamic optimization, with a focus on moving-optimum tracking in single-objective continuous problems.
- Design Goals and Key Features
  - Comprehensive Library: 27 EDOAs + 4 fully-parametric dynamic benchmarks.
  - Experimentation Interface: GUI & script modes, parallel runs, .mat-file save/restore, Excel export
  - Built-in Analysis & Visualization: Automated non-parametric tests, sensitivity analysis, box- and trend-plots
  - Modular & Extensible: Easily add new algorithms, benchmarks, or performance indicators
  - Educational & User-Friendly: 2D dynamic visualization app and clear, well-commented MATLAB code

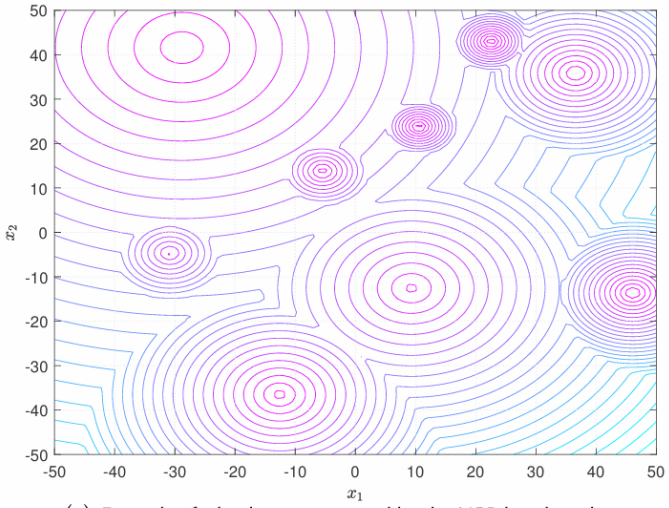
EDOLAB - Comprehensive Library

- 27 Classic Dynamic Optimization Algorithms

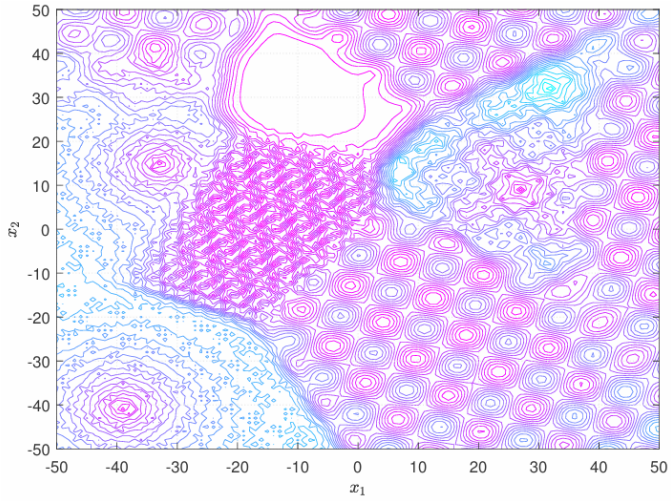
EDOA	Reference	Optimization component	Population structure	Number of sub-populations	Population size	Population clustering	Sub-population heterogeneity
ACF_{PSO}	[Yazdani et al. 2020a]	PSO	Multi-population	Adaptive	Adaptive	By index	Homogeneous
AMP_{DE}	[Li et al. 2016]	DE	Multi-population	Adaptive	Adaptive	By position	Heterogeneous
AMP_{PSO}	[Li et al. 2016]	PSO	Multi-population	Adaptive	Adaptive	By position	Heterogeneous
AmQSO	[Blackwell et al. 2008]	PSO	Multi-population	Adaptive	Adaptive	By index	Homogeneous
AMSO	[Li et al. 2014]	PSO	Multi-population	Adaptive	Adaptive	By position	Heterogeneous
CDE	[Du Plessis and Engelbrecht 2012]	DE/best/2/bin	Multi-population	Fixed	Fixed	By index	Homogeneous
CESO	[Lung and Dumitrescu 2007]	DE/rand/1/exp and PSO	Bi-population	Fixed	Fixed	N/A	Heterogeneous
CPSO	[Yang and Li 2010]	PSO	Multi-population	Adaptive	Fixed	By position	Heterogeneous
CPSOR	[Li and Yang 2012]	PSO	Multi-population	Adaptive	Fixed	By position	Heterogeneous
DSPSO	[Parrott and Li 2006]	PSO	Multi-population	Adaptive	Fixed	By position and fitness	Heterogeneous
DynDE	[Mendes and Mohais 2005]	DE/best/2/bin	Multi-population	Fixed	Fixed	By index	Homogeneous
DynPopDE	[du Plessis and Engelbrecht 2013]	DE/best/2/bin	Multi-population	Adaptive	Adaptive	By index	Homogeneous
FTMPSO	[Yazdani et al. 2013]	PSO	Multi-population	Adaptive	Adaptive	By index	Heterogeneous
HmSO	[Kamosi et al. 2010]	PSO	Multi-population	Fixed	Fixed	By index	Homogeneous
IDSPSO	[Blackwell et al. 2008]	PSO	Multi-population	Adaptive	Fixed	By position and fitness	Heterogeneous
ImQSO	[Kordestani et al. 2019]	PSO	Multi-population	Fixed	Fixed	By index	Homogeneous
mCMA-ES	[Yazdani et al. 2019]	CMA-ES	Multi-population	Adaptive	Adaptive	By index	Homogeneous
mDE	[Yazdani et al. 2019]	DE/best/2/bin	Multi-population	Adaptive	Adaptive	By index	Homogeneous
mjDE	[Yazdani et al. 2019]	jDE	Multi-population	Adaptive	Adaptive	By index	Homogeneous
mPSO	[Yazdani et al. 2019]	PSO	Multi-population	Adaptive	Adaptive	By index	Homogeneous
mQSO	[Blackwell and Branke 2006]	PSO	Multi-population	Fixed	Fixed	By index	Homogeneous
psfNBC	[Luo et al. 2018]	PSO	Multi-population	Adaptive	Fixed	By position and fitness	Homogeneous
RPSO	[Hu and Eberhart 2002]	PSO	Single-population	Fixed	Fixed	N/A	N/A
$SPSO_{AD+AP}$	[Yazdani et al. 2023]	PSO	Multi-population	Adaptive	Adaptive	By position and fitness	Homogeneous
TMIPSO	[Wang et al. 2007]	PSO	Bi-population	Fixed	Fixed	N/A	Heterogeneous
APCPSO	[Liu et al. 2020]	PSO	Multi-population	Adaptive	Adaptive	By position	Heterogeneous
DPCPSO	[Li et al. 2024]	PSO	Multi-population	Adaptive	Adaptive	By position	Heterogeneous

EDOLAB - Comprehensive Library

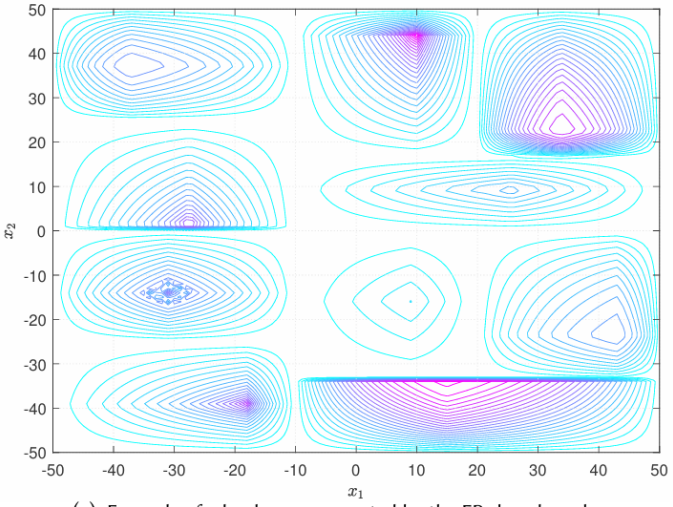
- 4 Dynamic Benchmark Generators (MPB, GDBG, FPs, GMPB)



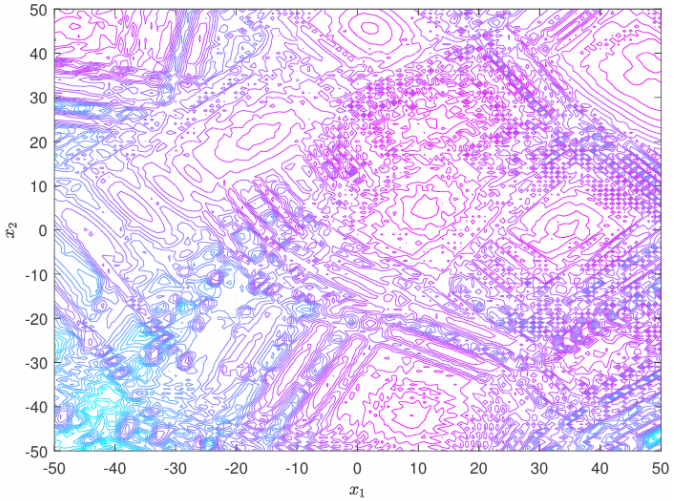
(a) Example of a landscape generated by the MPB benchmark.



(b) Example of a landscape generated by the GDBG benchmark.



(c) Example of a landscape generated by the FPs benchmark.



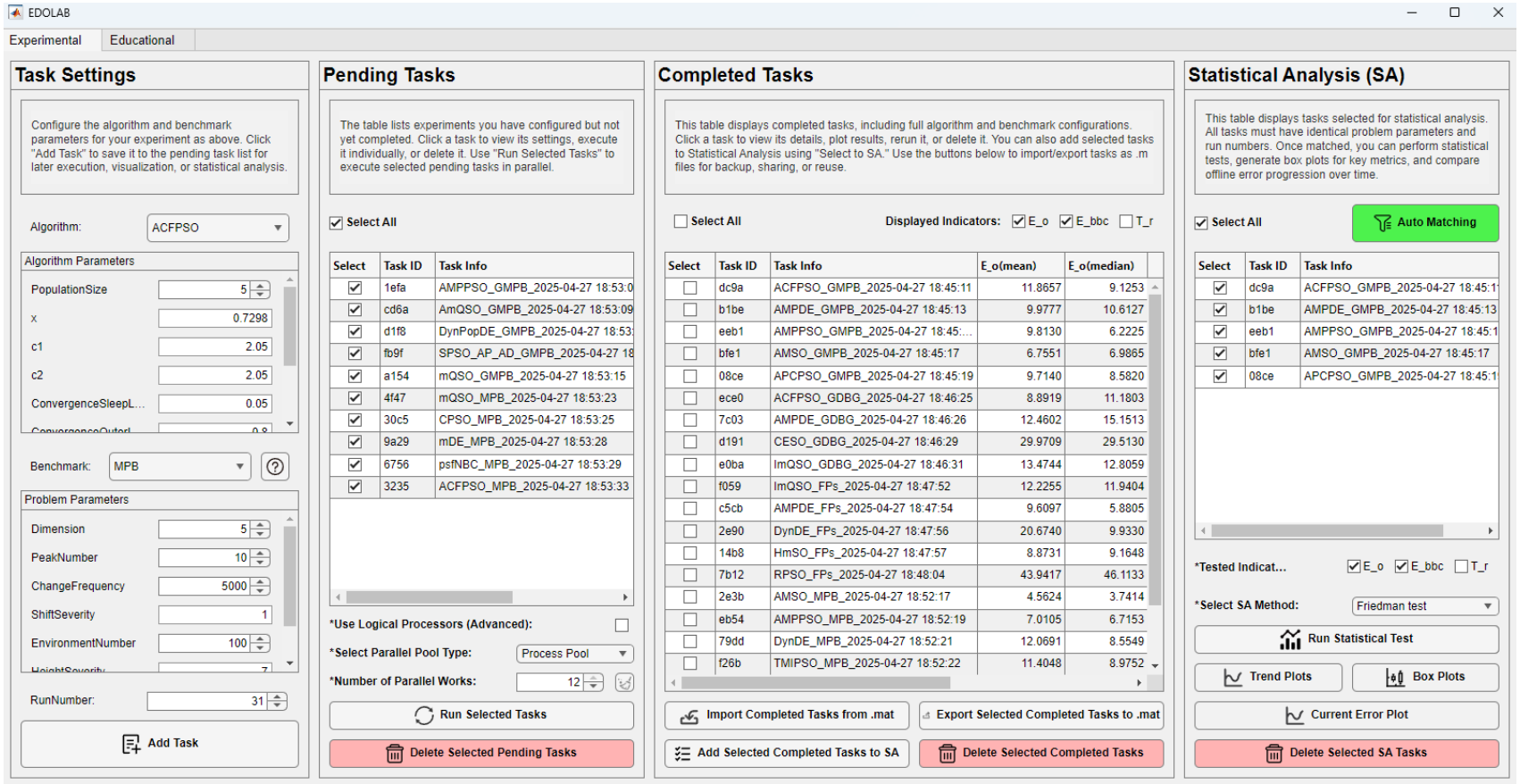
(d) Example of a landscape generated by the GMPB benchmark.

EDOLAB - Experimentation Application

- Experimentation application is designed for conducting and managing optimization experiments.
  - Configure all algorithm and benchmark parameters via GUI
  - Run multiple experiments in parallel
  - Save/load experiment results using .mat files
  - Export results to Excel for easy reporting
  - Built-in statistical analysis:
    - Friedman test
    - Wilcoxon signed-rank test
    - Wilcoxon rank-sum test
  - Visualization tools:
    - Box plots for performance distributions
    - Trend plots for over time behavior analysis

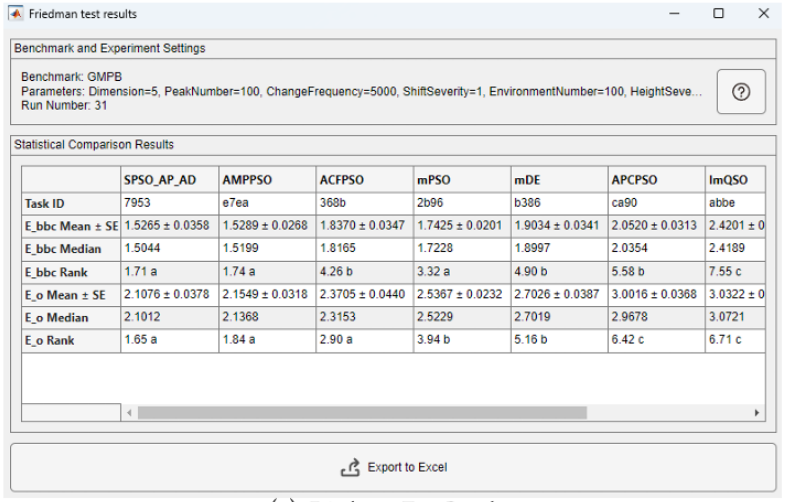
EDOLAB - Experimentation Application

- Experimentation application is designed for conducting and managing experiments.



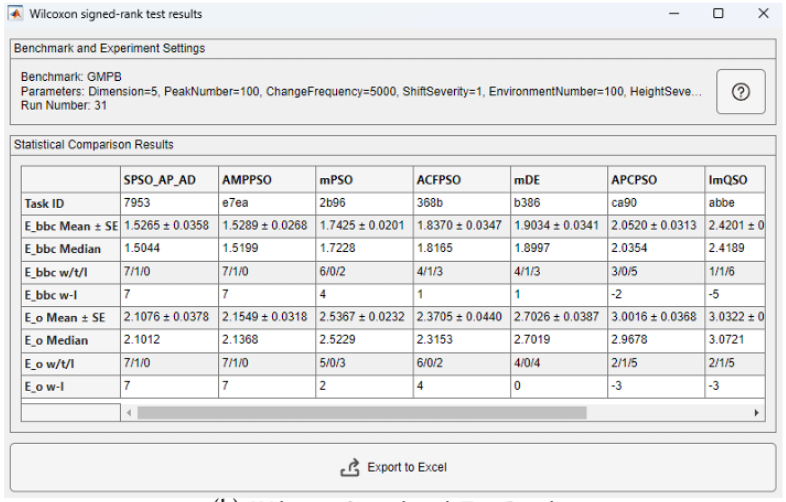
EDOLAB - Experimentation Application

- Built-in statistical analysis:



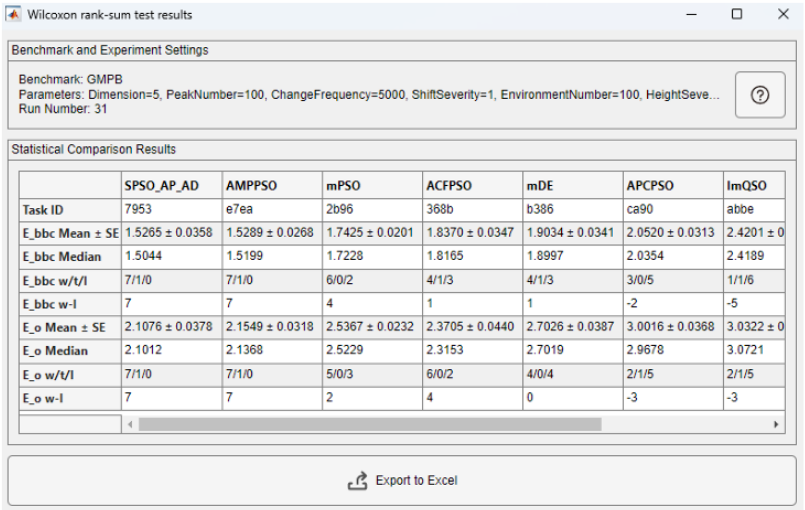
	SPSO_AP_AD	AMPPSO	ACFPSO	mPSO	mDE	APCPSO	ImQSO
Task ID	7953	e7ea	368b	2b96	b386	ca90	abbe
E_bbc Mean ± SE	1.5265 ± 0.0358	1.5289 ± 0.0268	1.8370 ± 0.0347	1.7425 ± 0.0201	1.9034 ± 0.0341	2.0520 ± 0.0313	2.4201 ± 0
E_bbc Median	1.5044	1.5199	1.8165	1.7228	1.8997	2.0354	2.4189
E_bbc Rank	1.71 a	1.74 a	4.26 b	3.32 a	4.90 b	5.58 b	7.55 c
E_o Mean ± SE	2.1076 ± 0.0378	2.1549 ± 0.0318	2.3705 ± 0.0440	2.5367 ± 0.0232	2.7026 ± 0.0387	3.0016 ± 0.0368	3.0322 ± 0
E_o Median	2.1012	2.1368	2.3153	2.5229	2.7019	2.9678	3.0721
E_o Rank	1.65 a	1.84 a	2.90 a	3.94 b	5.16 b	6.42 c	6.71 c

(a) Friedman Test Results.



	SPSO_AP_AD	AMPPSO	mPSO	ACFPSO	mDE	APCPSO	ImQSO
Task ID	7953	e7ea	2b96	368b	b386	ca90	abbe
E_bbc Mean ± SE	1.5265 ± 0.0358	1.5289 ± 0.0268	1.7425 ± 0.0201	1.8370 ± 0.0347	1.9034 ± 0.0341	2.0520 ± 0.0313	2.4201 ± 0
E_bbc Median	1.5044	1.5199	1.7228	1.8165	1.8997	2.0354	2.4189
E_bbc w/t/l	7/1/0	7/1/0	6/0/2	4/1/3	4/1/3	3/0/5	1/1/6
E_bbc w-l	7	7	4	1	1	-2	-5
E_o Mean ± SE	2.1076 ± 0.0378	2.1549 ± 0.0318	2.5367 ± 0.0232	2.3705 ± 0.0440	2.7026 ± 0.0387	3.0016 ± 0.0368	3.0322 ± 0
E_o Median	2.1012	2.1368	2.5229	2.3153	2.7019	2.9678	3.0721
E_o w/t/l	7/1/0	7/1/0	5/0/3	6/0/2	4/0/4	2/1/5	2/1/5
E_o w-l	7	7	2	4	0	-3	-3

(b) Wilcoxon Signed-rank Test Results.

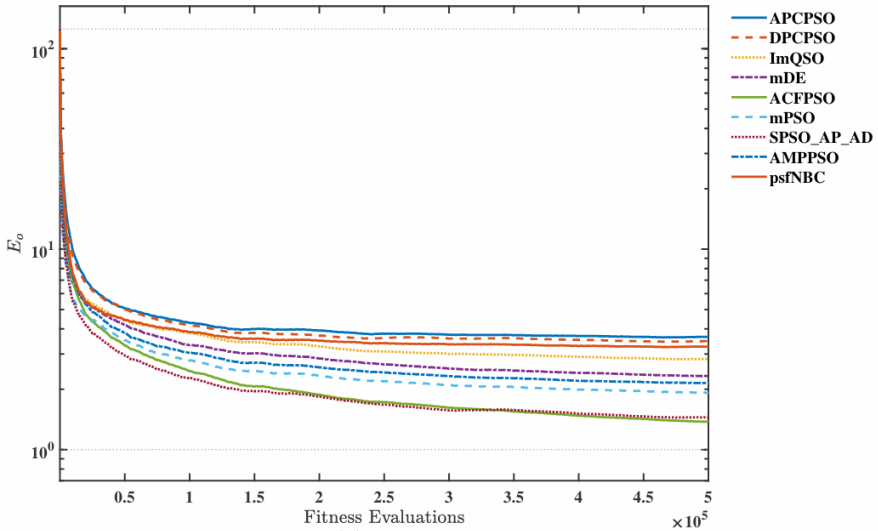


	SPSO_AP_AD	AMPPSO	mPSO	ACFPSO	mDE	APCPSO	ImQSO
Task ID	7953	e7ea	2b96	368b	b386	ca90	abbe
E_bbc Mean ± SE	1.5265 ± 0.0358	1.5289 ± 0.0268	1.7425 ± 0.0201	1.8370 ± 0.0347	1.9034 ± 0.0341	2.0520 ± 0.0313	2.4201 ± 0
E_bbc Median	1.5044	1.5199	1.7228	1.8165	1.8997	2.0354	2.4189
E_bbc w/t/l	7/1/0	7/1/0	6/0/2	4/1/3	4/1/3	3/0/5	1/1/6
E_bbc w-l	7	7	4	1	1	-2	-5
E_o Mean ± SE	2.1076 ± 0.0378	2.1549 ± 0.0318	2.5367 ± 0.0232	2.3705 ± 0.0440	2.7026 ± 0.0387	3.0016 ± 0.0368	3.0322 ± 0
E_o Median	2.1012	2.1368	2.5229	2.3153	2.7019	2.9678	3.0721
E_o w/t/l	7/1/0	7/1/0	5/0/3	6/0/2	4/0/4	2/1/5	2/1/5
E_o w-l	7	7	2	4	0	-3	-3

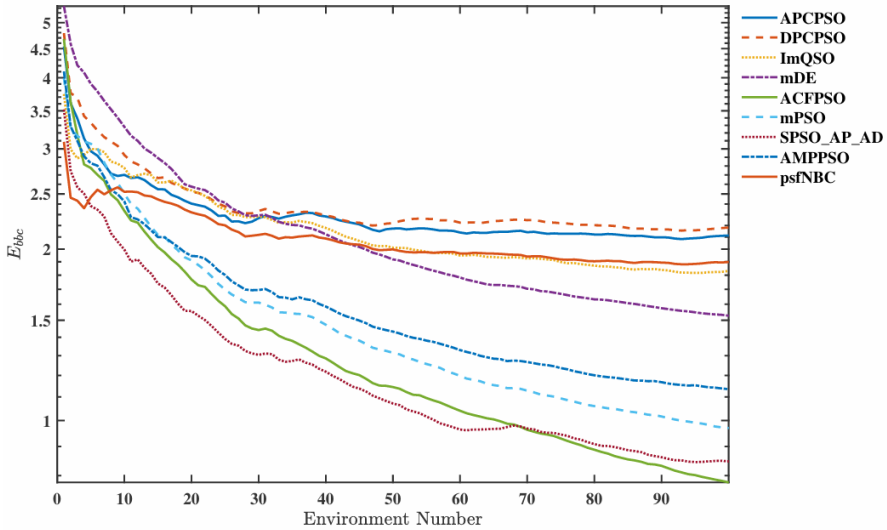
(c) Wilcoxon Rank-sum Test Results.

EDOLAB - Experimentation Application

- Visualization tools: Trend Plots
  - Show the evolution of selected performance indicators over time, e.g., E_o versus fitness evaluations or E_{bbc} versus environment number



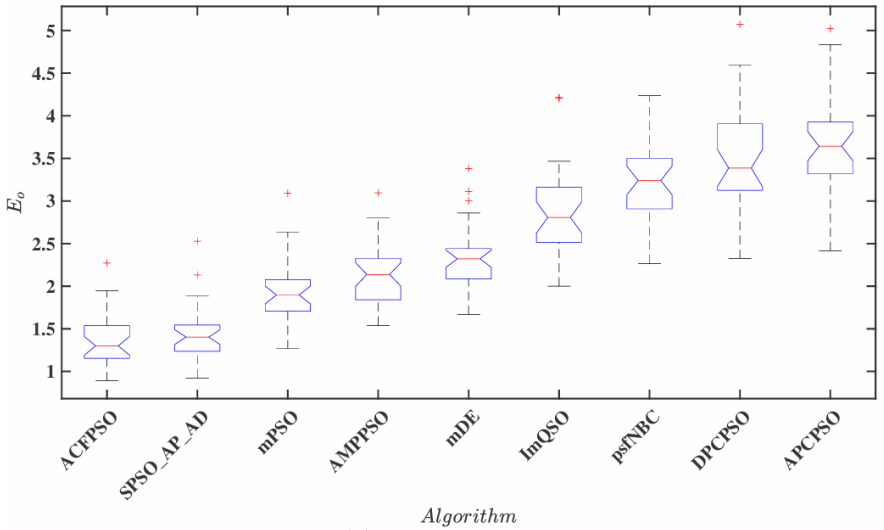
(a) Trend Plot of E_o.



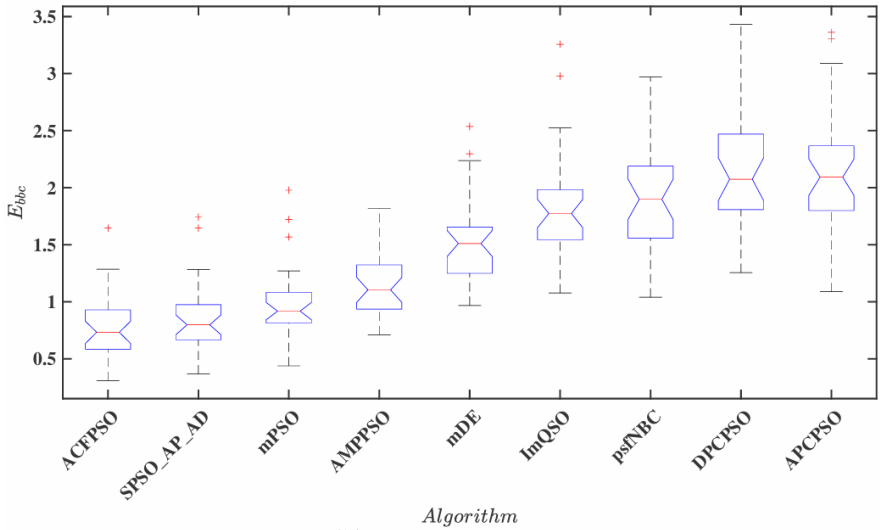
(b) Trend Plot of E_{bbc}

EDOLAB - Experimentation Application

- Visualization tools: Box Plots
  - Notched box plots visualize the distribution of final performance indicator values across tasks



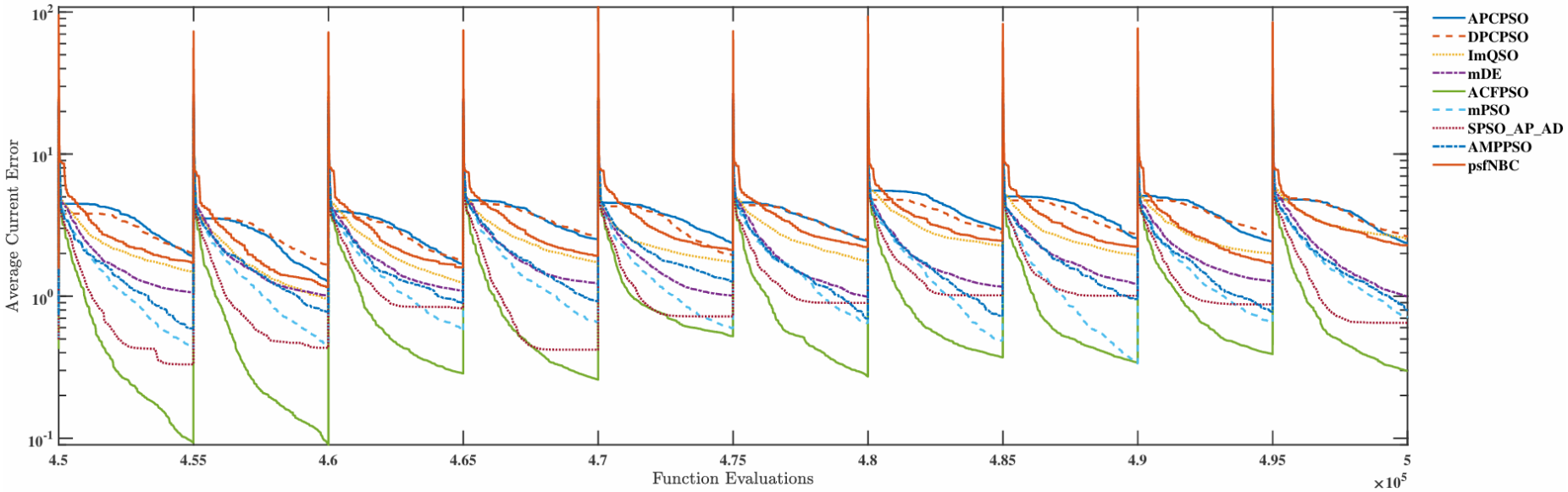
(a) Box Plot of E_o.



(b) Box Plot of E_{bbc}

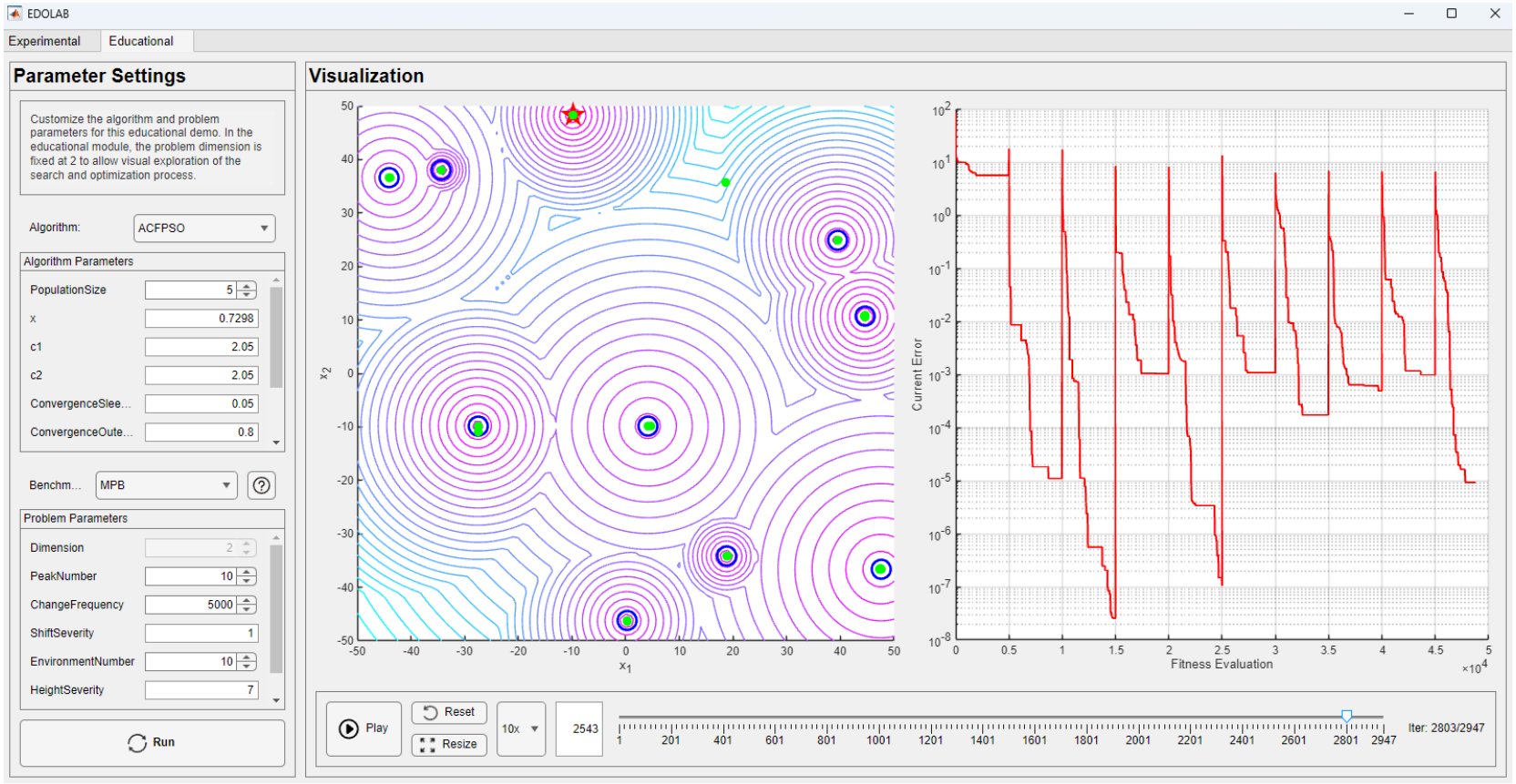
EDOLAB - Experimentation Application

- Visualization tools: Current Error Plot
  - The current error of the best-so-far solution within the current environment, averaged across all runs



EDOLAB – Education Application

- The education application allows users to visually observe the current environment, environmental changes, and the behavior of individuals over time.



Get Started with EDOLAB

- Check out our GitHub repository(https://github.com/Danial-Yazdani/EDOLAB-MATLAB)
  - Read the User Manual: Detailed usage instructions are available in the EDOLAB folder
  - Report Issues & Ask Questions: Use the “Issues” tab on GitHub to report bugs, request new features, or ask for help
  - Share Your Algorithms & Benchmarks: Fork the repo, add your own EDOAs or benchmarks, and submit a pull request
  - Stay Updated: Star ⭐ the project to follow updates, watch the repo to receive notifications about releases and discussions

EC for DOPs: Future Work

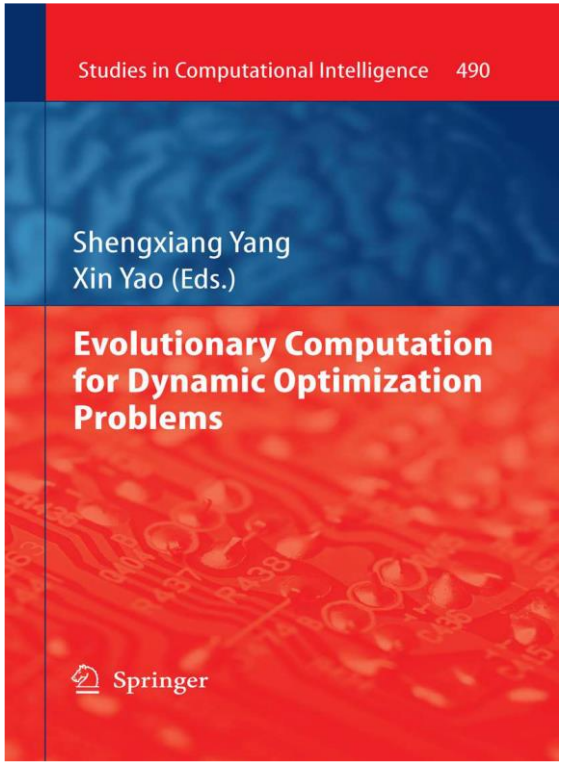
- The domain has attracted a growing interest recently
  - But, far from well-studied
- New approaches needed: esp. hybrid approaches
- Theoretical analysis: greatly needed
- EC for DMOPs, DCOPs, and DCMOPs
- Real world applications: also greatly needed
  - Fields: logistics, transport, MANETs, data streams, social networks,

Summary

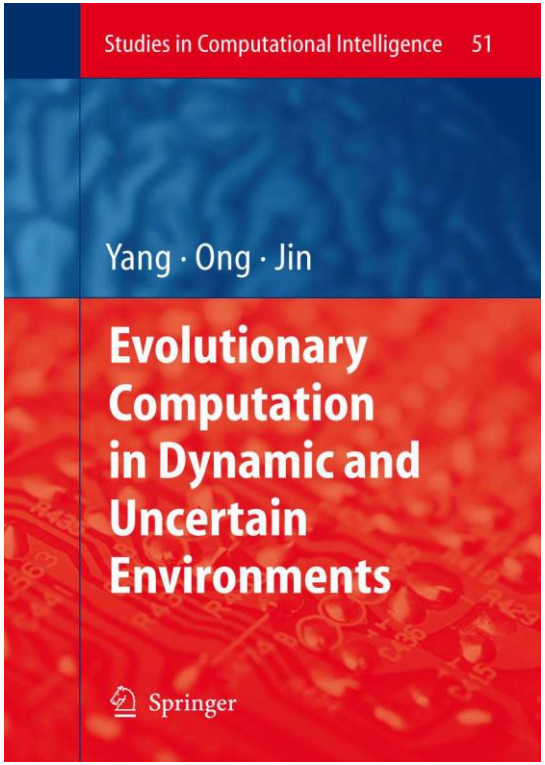
- EC for DOPs: important area
  - The domain is still young and active
  - Many challenges to be taken
- More young researchers are greatly welcome!

Two Relevant Books

- S. Yang and X. Yao (eds.), Evolutionary Computation for Dynamic Optimization Problems, in the series Studies in Computational Intelligence, vol. 490, Springer, 2013



- S. Yang, Y.-S. Ong, Y. Jin (eds.), Evolutionary Computation in Dynamic and Uncertain Environments, in the series Studies in Computational Intelligence, vol. 51. Springer, March 2007



Relevant Information

- IEEE CIS Task Force on EC in Dynamic and Uncertain Environments (https://ieee-tf-ecidue.cug.edu.cn/)
- Source codes: http://www.tech.dmu.ac.uk/~syang/publications.html
- Survey papers:
  - S. Jiang, J. Zou, S. Yang, and X. Yao. Evolutionary dynamic multi-objective optimization: A survey. ACM Comput. Survey, 55(4), Article 76, May 2023.
  - D. Yazdani, R. Cheng, D. Yazdani, J. Branke, Y. Jin, and X. Yao. A survey of evolutionary continuous dynamic optimization over two decades—Part A. IEEE Trans. on Evol. Comput., 25(4): 609-629, 2021.
  - D. Yazdani, R. Cheng, D. Yazdani, J. Branke, Y. Jin, and X. Yao. A survey of evolutionary continuous dynamic optimization over two decades—Part B. IEEE Trans. on Evol. Comput., 25(4): 630-650, 2021.
  - M. Mavrovouniotis, C. Li, and S. Yang. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. Swarm and Evolutionary Computation, 33: 1-17, 2017.
  - T. T. Nguyen, S. Yang, J. Branke. Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evol. Comput., 6: 1-24, 2012.