Multi-Population Methods with Clustering in Dynamic Environments

Shengxiang Yang

shengxiang.yang@brunel.ac.uk

http://people.brunel.ac.uk/~csstssy

Department of Information Systems and Computing, Brunel University

Uxbridge, Middlesex UB8 3PH, United Kingdom

Acknowledgment

- Funded by EPSRC project "Evolutionary Algorithms for Dynamic Optimization Problems"
  - A joint project between University of Birmingham, University of Leicester, and Brunel University, with support from BT and Honda
- This is a joint work with Mr Changhe Li
  - Submitted to *IEEE Trans on Evol Comput*, 2011

Outline of the Talk

- Introduction to dynamic optimization problems (DOPs)
- Multi-population methods for DOPs
- Framework of multi-population with clustering
- Experimental study
- Conclusions

Dynamic Optimization Problems (DOPs)

- DOPs: problems that change over time

$$F = f(\vec{x}, \vec{\phi}, t)$$

  $\vec{x}$: decision variable(s), $\vec{\phi}$: parameter(s), t: time
- Change may involve factors:
  - Objectives, constraints, environmental parameters
- Key characteristics of dynamism
  - Speed, severity, periodicity, detectability/predictability
- DOPs attracted a growing interest in recent years
  - Books, journals, competition, workshops/conferences

Approaches for EAs to Address DOPs

- Diversity schemes: handle convergence directly
- Multi-population schemes: co-operate sub-populations
- Memory schemes: store and reuse useful information
- Adaptive schemes: adapt generators and parameters
- Prediction schemes: predict changes and anticipate
- No clear winner and different interactions exist among approaches
- Golden rule: balancing exploration & exploitation over time

Multi-Population Methods for DOPs – I

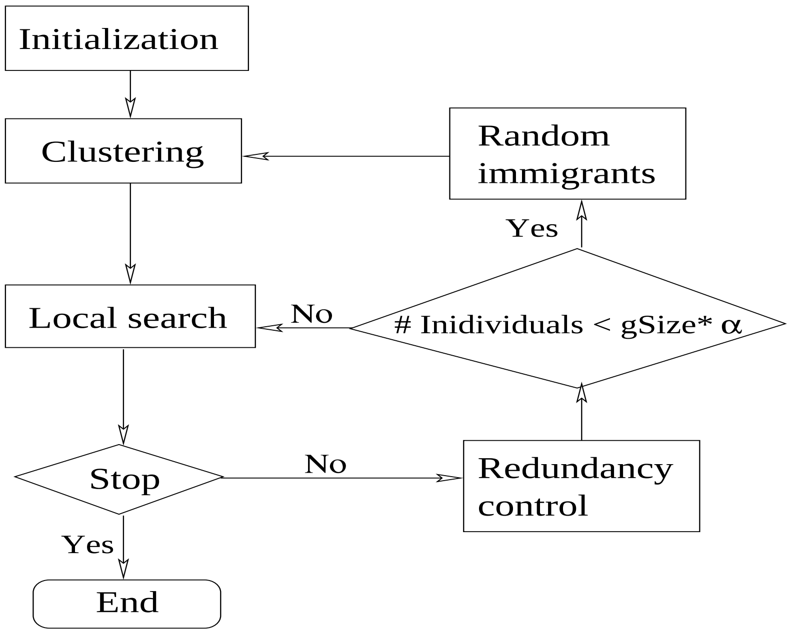
- Aim: maintain multi-populations on different peaks to locate and track multiple optima
- Key questions:
  - How to determine the proper number of sub-populations
  - How to calculate the search area of each sub-population
  - How to create sub-populations
- Algorithms:
  - k-means clustering algorithm: Kennedy'00
  - Shifting balance GA (SBGA): Oppacher & Wineberg '99
  - Self organizing scouts (SOS) GA: Branke et al '00
  - *nbest* PSO and niching PSO (NichePSO): Brits '02
  - Speciation based PSO (SPSO): Parrott and Li '04
  - Charged PSO (mCPSO) and quantum swarm optimization (mQSO): Blackwell and Branke '06

Multi-Population Methods for DOPs – II

- Limitations of the above algorithms:
  - The number of sub-swarms is predefined (k-means PSO, mCPSO, and mQSO)
  - The search radius of each sub-swarm must be given by experimental experience (SPSO, mCPSO, and mQSO)
  - Simply create sub-swarms without analysing the population distribution (NichePSO and SPSO)
- Problems might be caused by the above algorithms:
  - There may be improper number of sub-swarms
  - One sub-swarm might cover more than one peaks
  - One peak might be surrounded by more than one sub-swarms

Multi-Population with Clustering: Framework

- Clustering: To create a proper number of sub-populations
- Local search: An EA to fast converge on local optima
- Redundancy control: To save computational resource
- Random immigrants: To increase the population diversity



Clustering Method

- Single Linkage Hierarchical Clustering is used
  - Creates a list G of clusters with each cluster containing one individual in the intitial population pop
  - In each iteration, find a pair of clusters r and s such that they are the closest among all pairs of clusters and the total number of particles in them is not greater than $subSize$; if successful, combines r and s into one cluster
  - This iteration continues until all clusters in G contain more than one individual

Local Search Strategy

- After clustering, each sub-population searches a local area covered
- Any EA can be used as the local search strategy, i.e., the framework can be intantiated into any EA

Redundancy Control

// overlapping check
for each pair of sub-populations (t, s) in $plst$ **do**
  if $r_{overlap}(t, s) > \beta$ **then**
    Merge t and s into t
    Remove s from $plst$
  end if
end for
// overcrowding check
for each sub-population $t \in plst$ **do**
  if $|t| > subSize$ **then**
    Remove worst $(|t| - subSize)$ individuals from t
  end if
end for
// convergence check
for each sub-population $s \in plst$ **do**
  if $radius(s) < \epsilon$ **then**
    Remove s from $plst$
  end if
end for

Maintaining Diversity

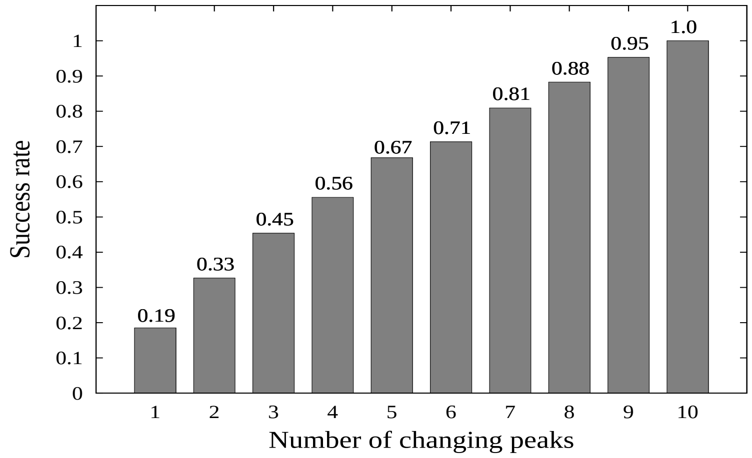
- Idea: Increase the population diversity if it deceases to a certain level
- The population diversity is measured by the following ratio:

$$div(t) = n_{ind}(t)/gSize$$

  where $n_{ind}(t)$ is the number of survived individuals at iteration t and $gSzie$ is the size of the initial population
- If $div(t)$ decreases to a threshold (α), a temporal population of size $gSize - n_{ind}(t)$ is randomly generated
- The temporal population is clustered and the obtained new sub-populations are added to the whole list of sub-populations

Multi-Population with Clustering: Features

- Automatically creates a proper number of sub-pops in different sub-areas
- Works in dynamic environments with any properties
  - Does not depend on the detection of changes. Good for hard-to-detect or undetectable dynamic environments

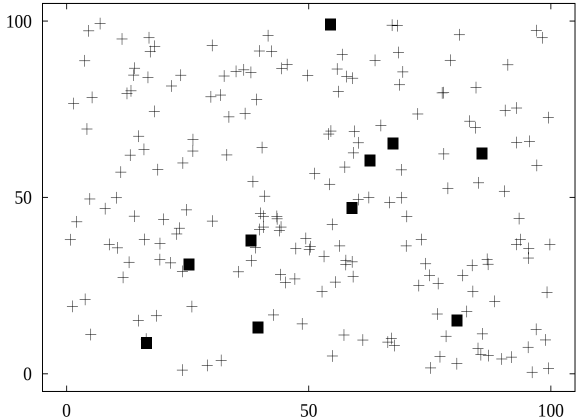


- Easy to be instantiated into any EA, e.g., PSO, GA, and DE

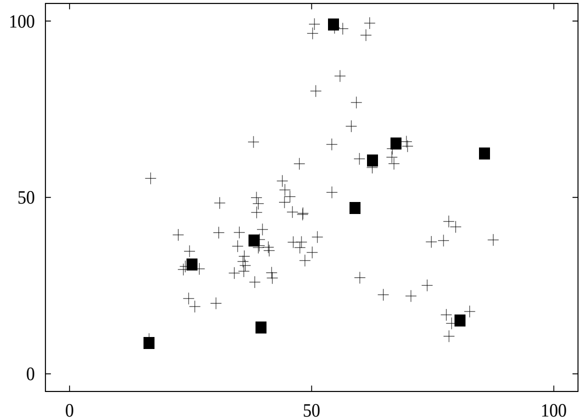
Experiments on the MPB Problem

Parameter	Value	Parameter	Value
p (number of peaks)	10	correlation coefficient λ	0
change frequency	5000	S	[0, 100]
height severity	7.0	H	[30.0, 70.0]
width severity	1.0	W	[1, 12]
peak shape	cone	I	50.0
basic function	no	Population size	100
shift length s	1.0	Number of changes	100
number of dimensions D	5	Number of runs	30
		percentage of changing peaks	100%

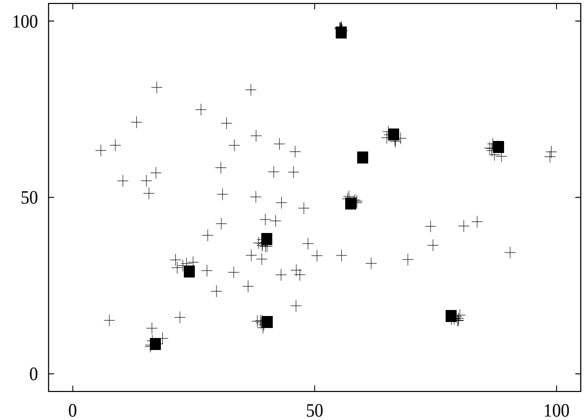
Clustering PSO – Dynamic Behaviour



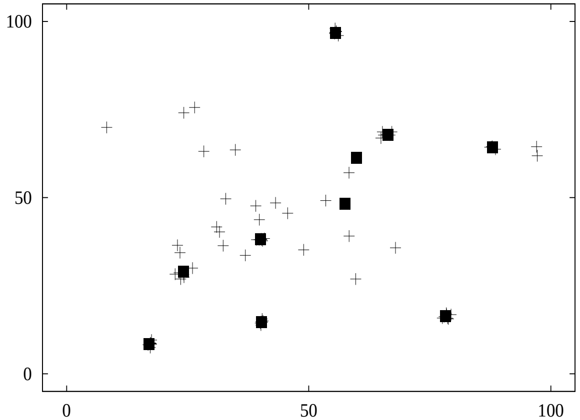
total particles: 194, the number of sub-swarms: 36, evals: 0



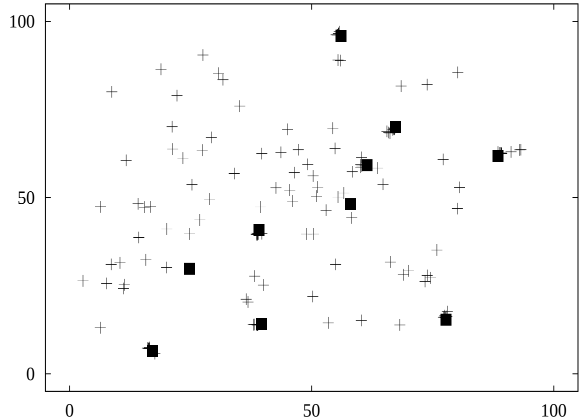
total particles: 78, the number of sub-swarms: 14, evals: 1019



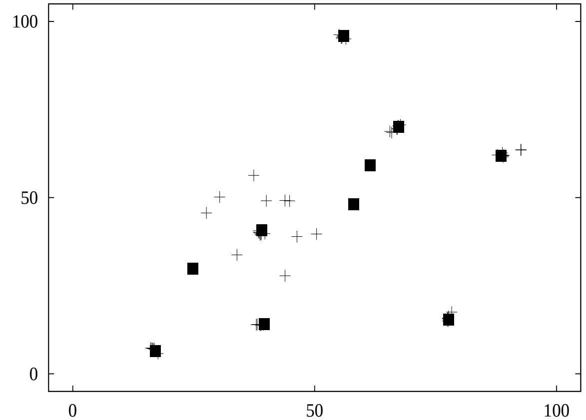
total particles: 107, the number of sub-swarms: 17, evals: 6000



total particles: 78, the number of sub-swarms: 14, evals: 7748



total particles: 137, the number of sub-swarms: 24, evals: 12000



total particles: 61, the number of sub-swarms: 11, evals: 12497

Clustering EAs – Comparison Results

- Algorithm rankings on MPB with different shift severities

Algorithm	Severity of shift length (s)							Total
	0	1	2	3	4	5	6	
CPSOR	1	1	2	2	2	2	2	2
CGAR	10	10	8	7	8	7	5	7
CDER	11	11	11	11	11	11	11	11
CPSO	2	2	1	1	1	1	1	1
mCPSO	8	8	9	9	10	9	9	10
mQSO	9	7	7	8	9	8	8	8
CESO	4	4	3	3	3	3	3	3
rSPSO	3	6	5	6	7	6	6	6
SPSO	7	9	10	10	5	10	10	9
ESCA	5	5	4	4	4	4	4	4
PSO-CP	6	3	6	5	6	5	7	5

Hard-to-Detect Dynamic Environments

- Fix the percentage of changing peaks ($cPeaks$)

s	$cPeaks$	0.1	0.3	0.5	0.7	0.9	1.0
0	CPSOR	**1.47**	**0.535**	**0.5**	**0.6**	1.72	**0.418**
	CPSO	3.01	2.7	0.904	0.765	**1.68**	0.465
1	CPSOR	**1.77**	**1.09**	**0.633**	**0.742**	**1.83**	**0.599**
	CPSO	3	2.76	0.912	1.02	2.11	0.715
2	CPSOR	**1.89**	**1.17**	**0.781**	**0.99**	**2.04**	0.849
	CPSO	3.24	2.96	0.939	1.19	2.24	**0.843**
3	CPSOR	**1.94**	**1.61**	0.995	**1.24**	**2.35**	0.964
	CPSO	3.11	3.38	**0.928**	1.32	2.58	**0.911**
4	CPSOR	**2.06**	**2.09**	1.25	1.58	**2.6**	1.38
	CPSO	3.17	3.29	**1.11**	**1.43**	2.66	**0.997**
5	CPSOR	**2.05**	**2.6**	1.34	1.72	2.88	1.69
	CPSO	3.27	3.79	**1.14**	**1.45**	**2.69**	**1.08**

Conclusions and Future Work

- Conclusions
  - Multi-population approaches are effective for DOPs
  - Clustering can automatically create a proper number of sub-pops in different sub-regions
  - Multi-population with clustering can address different dynamic environments, esp., with hard-to-detect or undetectable changes
- Future Work
  - To improve the performance of the clustering method
  - To design effective local search EAs within this framework
  - To test the framework in completely undetectable dynamic environments