Evolutionary Computation for Dynamic Optimization Problems

Shengxiang Yang

Centre for Computational Intelligence
School of Computer Science and Informatics
De Montfort University, United Kingdom
Email: syang@dmu.ac.uk
http://www.tech.dmu.ac.uk/∼syang

http://www.sigevo.org/gecco-2013/

GECCO'13 Companion, July 6-10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

Instructor/Presenter — Shengxiang Yang

- Education and career history:
  - PhD, Northeastern University, China, 1999
  - Worked at King's College London, University of Leicester, and Brunel University, 1999-2012
  - Joined De Montfort University as Professor in Computational Intelligence (CI) in July 2012
  - Director of Centre for Computational Intelligence (CCI)
- Research interests:
  - Evolutionary computation (EC) and nature-inspired computation
  - Dynamic optimisation and multi-objective optimisation
  - Relevant real-world applications
- Over 160 publications and over £1M funding as the PI
- Editor, *Evolutionary Computation* and 3 other journals
- Chair of two IEEE CIS Task Forces
  - EC in Dynamic and Uncertain Environments
  - Intelligent Network Systems

Centre for CI (CCI), De Montfort University

- CCI (www.cci.dmu.ac.uk):
  - Mission: Developing fundamental theoretical and practical solutions to real world problems using a variety of CI paradigms
  - Members: 15 staff, 4 research fellows, 30+ PhDs, visiting researchers
  - Components: 3 Labs (Robotics, Gaze, Game Programming) & Bio-Health Informatics Research Group
- Funding:
  - Research Councils: EPSRC, TSB, Royal Academy of Engineering, Royal Society, KTP, Innovation Fellowships, HOPE foundation
  - 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, MSc Intelligent Systems & Robotics
  - BSc Artificial Intelligence with Robotics
- YouTube page: http://www.youtube.com/thecci

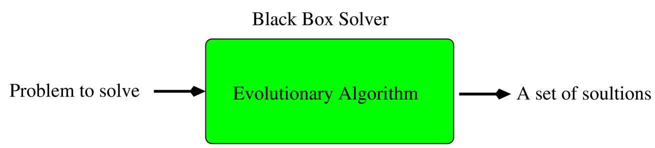
Outline of the Tutorial

Part I: Set up the stage
- Introduction to evolutionary computation (EC)
- EC for dynamic optimization problems (DOPs): Concept and motivation
- Benchmark and test problems
- Performance measures

Part II: Play the game
- EC approaches for DOPs
- Case studies
- Relevant issues
- Future work

What Is Evolutionary Computation (EC)?

- EC encapsulates a class of stochastic optimization 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 inspired from biological evolution



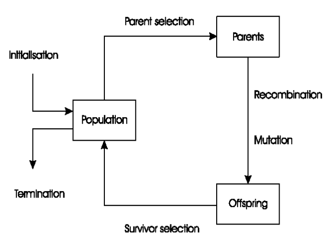
Design and Framework of an EA

Given a problem to solve, first consider two key things:

- Representation of solution into individual
- Evaluation or fitness function

Then, design the framework of an EA:

- 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

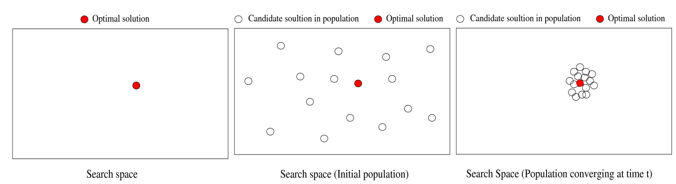


EC Applications

- EAs are easy-to-use: No strict requirements to problems
- Widely used for optimisation and search problems
  - Financial and economical systems
  - Transportation and logistics systems
  - Industry engineering
  - Automatic programming, art and music design
  - ......

EC for Optimisation Problems

- Traditionally, research on EAs has focused on static problems
  - Aim to find the optimum *quickly* and *precisely*



- But, many real-world problems are dynamic optimization problems (DOPs), where changes occur over time
  - In transport networks, travel time between nodes may change
  - In logistics, customer demands may change

What Are DOPs?

- In general terms, "optimization problems that change over time" are called *dynamic problems/time-dependent problems*

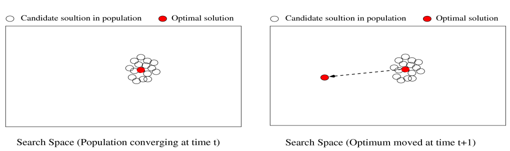
$$F = f(\vec{x}, \vec{\phi}, t)$$

– $\vec{x}$: decision variable(s); $\vec{\phi}$: parameter(s); t: time

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

Why DOPs Challenge EC?

- For DOPs, optima may move over time in the search space
  - Challenge: need to track the moving optima over time



- DOPs challenge traditional EAs
  - Once converged, hard to escape from an old optimum

Why EC for DOPs?

- Many real-world problems are DOPs
- EAs, once properly enhanced, are good choice
  - Inspired by natural/biological evolution, always in dynamic environments
  - Intrinsically, should be fine to deal with DOPs
- Many events on EC for DOPs recently

Relevant Events

- Books (Monograph or Edited):
  - Yang & Yao, 2013; Yang *et al.*, 2007; Morrison, 2004; Weicker, 2003; Branke, 2002
- PhD Theses:
  - Mavrovouniotis, 2013; du Plessis, 2012; Li, 2011; Nguyen, 2011; Simoes, 2010
- Journal special issues:
  - Neri & Yang, 2010; Yang *et al.*, 2006; Jin & Branke, 2006; Branke, 2005
- Workshops and conference special sessions:
  - EvoSTOC (2004–2013): part of Evo*
  - ECiDUE (2004–2013): part of IEEE CEC
  - EvoDOP ('99, '01, '03, '05, '07, '09): part of GECCO
- IEEE Symposium on CIDUE (Paris, 2011; Singapore, 2013)
- IEEE Competitions: within IEEE CEC 2009 & CEC 2012

Benchmark and Test DOPs

- Basic idea: change base static problem(s) 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

The DF1 Generator

- Proposed by Morrison & De Jong (1999)
- The base landscape in the D-dimensional real space:

$$f(\vec{x}) = \max_{i=1,\dots,p} \left[H_i - R_i \times \sqrt{\sum_{j=1}^{D} (x_j - X_{ij})^2} \right]$$

– $\vec{x} = (x_1, \cdots, x_D)$: a point in the landscape; p: number of peaks
– H_i, R_i, $X_i = (X_{i1}, \cdots, X_{iD})$: height, slope, center of peak i

- The dynamics is controlled by a logistics function:

$$\Delta_t = A \cdot \Delta_{t-1} \cdot (1 - \Delta_{t-1})$$

– $A \in [1.0, 4.0]$: a constant; Δ_t: step size of changing a parameter

Moving Peaks Benchmark (MPB) Problem

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

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

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

- The dynamics:

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

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

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

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

– $\sigma \sim N(0,1)$; 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 Knapsack Problems (DKPs)

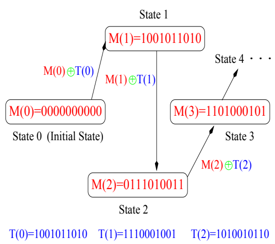
- Static knapsack problem:
  - Given n items, each with a weight and a profit, and a knapsack with a fixed capacity, select items to fill up the knapsack to maximize the profit while satisfying the knapsack capacity constraint
- The DKP:
  - Constructed by changing weights and profits of items, and/or knapsack capacity over time as:

$$Max\ f(\vec{x}(t), t) = \sum_{i=1}^{n} p_i(t) \cdot x_i(t), \quad s.\ t.: \quad \sum_{i=1}^{n} w_i(t) \cdot x_i(t) \leq C(t)$$

– $\vec{x}(t) \in \{0,1\}^n$: a solution at time t
– $x_i(t) \in \{0,1\}$: indicates whether item i is included or not
– $p_i(t)$ and $w_i(t)$: profit and weight of item i at t
– $C(t)$: knapsack capacity at t

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

$$\vec{M}(0) = \vec{0} \quad \text{(the initial state)}$$
$$\vec{M}(k+1) = \vec{M}(k) \oplus \vec{T}(k)$$

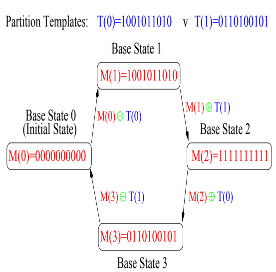
3. Evaluate an individual:

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

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

Constructing Cyclic Dynamic Environments

Can extend the 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*

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

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

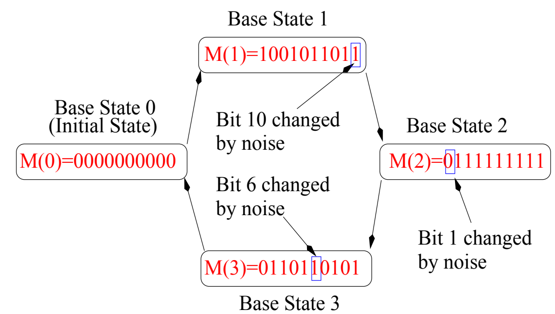
$$f(\vec{x}, t) = f(\vec{x} \oplus \vec{M}(I_t)) = f(\vec{x} \oplus \vec{M}(k\%(2K)))$$

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

Constructing Cyclic Environments with Noise

We can also construct cyclic environments with noise:

- Each time before a base state is entered, it is bitwise changed with a small probability



Dynamic Traveling Salesman Problems

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

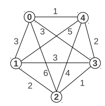
$$f(x, t) = Min(\sum_{i=1}^{n} d_{x_i, x_{i+1}}(t))$$

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

Dynamic Permutation Benchmark Generator

- The dynamic benchmark generator for permutation-encoded problems (DBGP) can create a DOP from any stationary TSP/VRP by swapping objects:

Optimum -> (0,4,3,2,1,0) = **9**

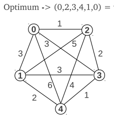


Distance matrix before change

	0	1	2	3	4
0	0	3	6	5	1
1	3	0	2	3	3
2	6	2	0	1	4
3	5	3	1	0	2
4	1	3	4	2	0

Swap City Location (4,2)

Optimum -> (0,2,3,4,1,0) = **9**



Distance matrix after change

	0	1	2	3	4
0	0	3	1	5	6
1	3	0	3	3	2
2	1	3	0	2	4
3	5	3	2	0	1
4	6	2	4	1	0

1. Generate a random vector $\vec{r}(T)$ that contains all objects every f iterations
2. Generate another randomly re-order vector $\vec{r}'(T)$ that contains only the first $m \times n$ objects of $\vec{r}(T)$
3. Modify the encoding of the problem instance with $m \times n$ pairwise swaps

- More details: M. Mavrovouniotis, S. Yang, & X. Yao (2012). *PPSN XII*, Part II, LNCS 7492, pp. 508–517

Effect on Algorithms

- Similar with the XOR DOP generator, DBGP shifts the population of an alg. to new location in the fitness landscape
- The individual with the same encoding as before a change will have a different cost after the change

Evolutionary Algorithms

Population of Individuals (before Dynamic Change)

1 = (0,1,3,4,2,0) = **18**
2 = (1,4,2,3,0,1) = **16**
3 = (0,4,3,2,1,0) = **9**
⋮
μ = (3,0,4,2,1,3) = **15**

Dynamic Change

Population of Individuals (after Dynamic Change)

1 = (0,1,3,4,2,0) = **12**
2 = (1,4,2,3,0,1) = **16**
3 = (0,4,3,2,1,0) = **15**
⋮
μ = (3,0,4,2,1,3) = **21**

Ant Colony Optimization

Heuristic Information Matrix

	0	1	2	3	4
0	0	0.33	0.16	0.08	1
1	0.33	0	0.5	0.33	0.33
2	0.16	0.5	0	1	0.25
3	0.2	0.33	1	0	0.5
4	1	0.33	0.25	0.5	0

Dynamic Change

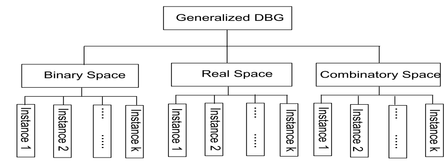
Heuristic Information Matrix

	0	1	2	3	4
0	0	0.33	1	0.2	0.16
1	0.33	0	0.33	0.33	0.5
2	1	0.33	0	0.5	0.25
3	0.2	0.33	0.5	0	1
4	0.16	0.5	0.25	1	0

- Can extend for cyclic and cyclic with noise environments

Generalized DOP Benchmark Generator (GDBG)

- Proposed by Li & Yang (2008), GDBG uses the model below:



- In GDBG, DOPs are defined as:

$$F = f(x, \phi, t),$$

– ϕ: system control parameter
- Dynamism results from tuning ϕ of the current environment

$$\phi(t+1) = \phi(t) \oplus \Delta\phi$$

– $\Delta\phi$: deviation from the current control parameter(s)
- The new environment at $t+1$ is as follows:

$$f(x, \phi, t+1) = f(x, \phi(t) \oplus \Delta\phi, t)$$

GDBG: Dynamic Change Types

- Change types:
  1. Small step: $\Delta\phi = \alpha \cdot \|\phi\| \cdot rand()$
  2. Large step: $\Delta\phi = \|\phi\| \cdot (\alpha + (1-\alpha)rand())$
  3. Random: $\Delta\phi = \|\phi\| \cdot rand()$
  4. Chaotic: $\phi(t+1) = A \cdot \phi(t) \cdot (1 - \phi(t)/\|\phi\|)$
  5. Recurrent: $\phi(t+1) = \phi(t\%P)$
  6. Recurrent with nosy: $\phi(t+1) = \phi(t\%P) + \alpha \cdot \|\phi\| \cdot rand()$
  7. ......
- More details:
  - C. Li & S. Yang (2008). *SEAL'08*, LNCS 5361, pp. 391–400

DOPs: Classification

Classification criteria:

- Time-linkage: Does the future behaviour of the problem depend on the current solution?
- Predictability: Are changes predictable?
- Visibility: Are changes visible or detectable
- Cyclicity: Are changes cyclic/recurrent in the search space?
- Factors that change: objective, domain/number of variables, constraints, and/or other parameters

DOPs: Common Characteristics

Common characteristics of DOPs in the literature:

- Most DOPs are non time-linkage problems
- For most DOPs, changes are assumed to be detectable
- In most cases, the objective function is changed
- Many DOPs have unpredictable changes
- Most DOPs have cyclic/recurrent changes

Performance Measures

- For EC for stationary 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
    - Mean distance to optimum at each generation
    - ......
  - Behaviour-based performance measures
    - Reactivity
    - Stability
    - Robustness
    - Satisfiability
    - Diversity measures
    - ......

Performance Measures: Examples

- Collective mean fitness (mean best-of-generation):

$$\overline{F}_{BOG} = \frac{1}{G} \times \sum_{i=1}^{i=G} \left(\frac{1}{N} \times \sum_{j=1}^{j=N} F_{BOG_{ij}}\right)$$

– G and N: number of generations and runs, resp.

– $F_{BOG_{ij}}$: best-of-generation fitness of generation i of run j

- Adaptation performance (Mori et al., 1997)

$$Ada = \frac{1}{T} \sum_{t=1..T} (f_{best}(t)/f_{opt}(t))$$

- Accuracy (Trojanowski and Michalewicz, 1999)

$$Acc = \frac{1}{K} \sum_{i=1..K} (f_{best}(i) - f_{opt}(i))$$

– $f_{best}(i)$: best fitness for environment i (best before change)

Part II: Play the Game

- EC approaches for DOPs
- Case studies
- Relevant issues
- Future work

EC for DOPs: First Thinking

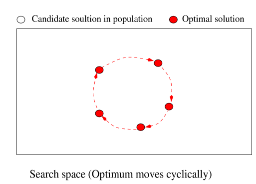
- Recap: traditional EAs are not good for DOPs
- Goal: to track the changing optimum
- How about restarting an EA after a change?
  - Natural and easy choice
  - But, not good choice because:
    1. It may be inefficient, wasting computational resources
    2. It may lead to very different solutions 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

EC for DOPs: General Approaches

- Many approaches developed to enhance EAs for DOPs
- Typical approaches:
  - Memory: store and reuse useful information
  - Diversity: handle convergence directly
  - Multi-population: co-operate sub-populations
  - Adaptive: adapt generators and parameters
  - Prediction: predict changes and take actions in advance
- They have been applied to different EAs for DOPs

Memory Approaches

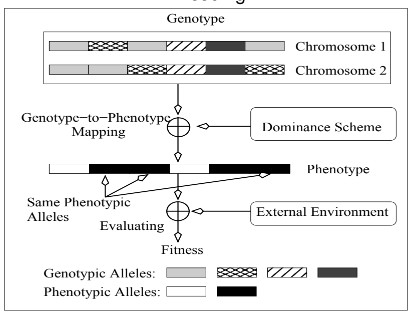
- Cyclic DOPs: change cyclically among a fixed set of states



- Memory works by storing and reusing useful information
- Two classes regarding how to store information
  - Implicit memory: uses redundant representations
    - Multiploidy and dominance (Ng & Wong, 1995; Lewis et al., 1998)
    - Dualism mechanisms (Yang, 2003; Yang & Yao, 2005)
  - Explicit memory: uses extra space to store information

Implicit Memory: Diploid Genetic Algorithm

Encoding



Dominance Scheme

	0	o	1	i
0	0	0	0/1	0
o	0	0	1	0/1
1	0/1	1	1	1
i	0	0/1	1	1

Ng & Wong (1995)

	A	B	C	D
A	0	0	0	1
B	0	0	0	1
C	0	0	1	1
D	1	1	1	1

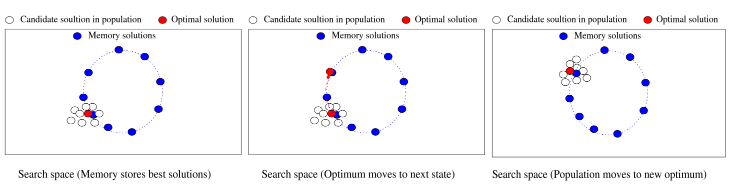
Lewis et al. (1998)

- Each individual has a pair of chromosomes
- Dominance scheme maps genotype to phenotype
- Dominance scheme may change or be adaptive (Uyar & Harmanci, 2005)

Explicit Memory Approaches

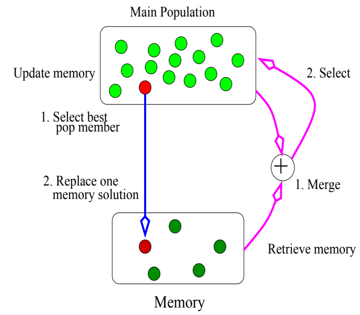
Basic idea: use extra memory

- With time, store useful information of the pop into memory
- When a change occurs, use memory to track new optimum

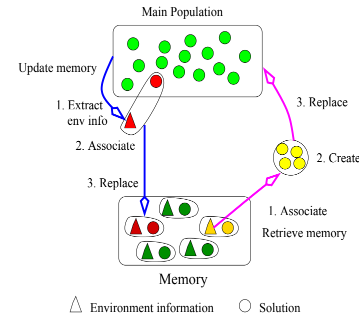


Explicit Memory: Direct vs Associative

- Direct memory: store good solutions (Branke, 1999)
- Associative memory: store environmental information + good solutions (Yang & Yao, 2008)



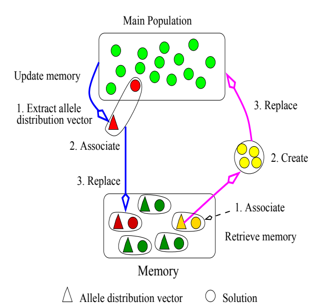
Direct Memory



Associative Memory

Associative Memory Based Genetic Algorithm

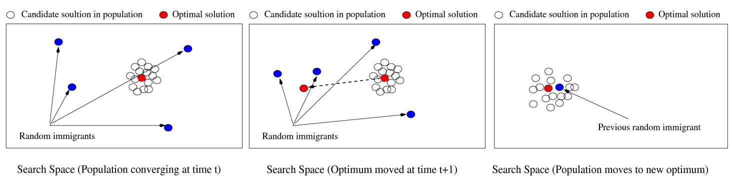
Idea: Use *allele distribution* (AD) $\vec{D}$ to represent environmental info.



- Use memory to store <$\vec{D}$, S> pairs
- Update memory by similarity policy
- Re-evaluate memory every generation. If change detected
  - Extract best memory AD: $\vec{D}_M$
  - Create solutions by sampling $\vec{D}_M$
  - Replace them into the pop randomly
- Details:
  - S. Yang (2006). *EvoWorkshops'06*, pp. 788–799

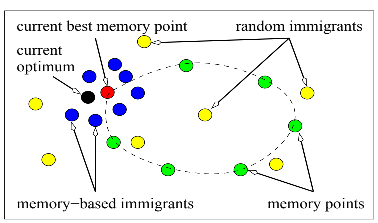
Diversity Approaches: Random Immigrants

- Convergence is the key problem in metaheuristics for DOPs
- Random immigrants:
  - Each generation, insert some random individuals (called *random immigrants*) into the population to maintain diversity
  - When optimum moves, random immigrants nearby take action to draw the pop to the new optimum



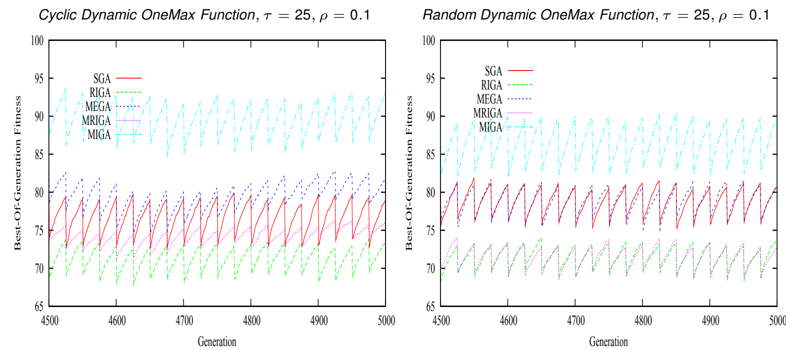
Memory-Based Immigrants

- Random immigrants maintain the diversity while memory adapts an algorithm directly to new environments
- **Memory-based immigrants**: uses memory to guide immigrants towards current environment
  - Re-evaluate the memory every generation
  - Retrieve the best memory point $B_M(t)$ as the base
  - Generate immigrants by mutating $B_M(t)$ with a prob.
  - Replace worst members in the population by these immigrants



Search Space

Experimental Results: Immigrants Based GAs



- Memory-based immigrants GA (MIGA) significantly beats other GAs
- More details:
  - S. Yang (2008). *Evol. Comput.*, 16(3): 385–416

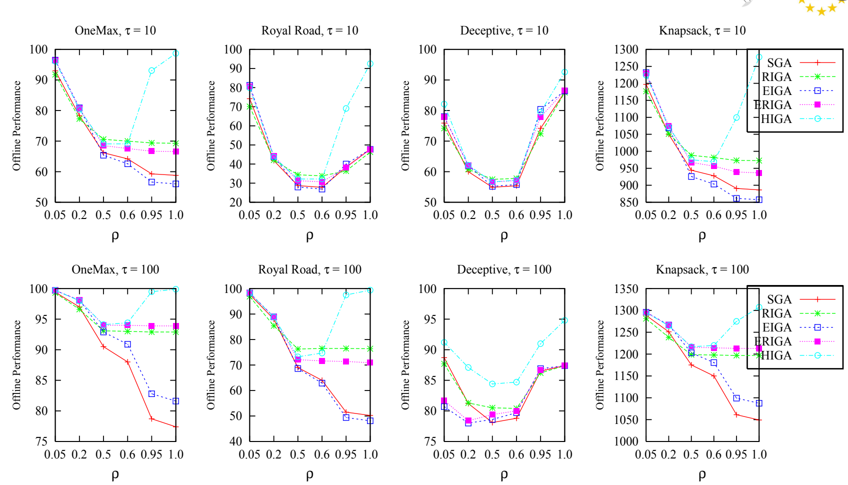
Hybrid Immigrants Approach

- Combines elitism, dualism and random immigrants ideas
- Dualism: Given $\vec{x} = (x_1, \cdots, x_l) \in \{0,1\}^l$, its dual is defined as

$$\vec{x}^d = dual(\vec{x}) = (x_1^d, \cdots, x_l^d) \in \{0,1\}^l$$

  where $x_i^d = 1 - x_i$
- Each generation t, select the best individual from previous generation, $E(t-1)$, to generate immigrants
  - Elitism-based immigrants: Generate a set of individuals by mutating $E(t-1)$ to address slight changes
  - Dualism-based immigrants: Generate a set of individuals by mutating the dual of $E(t-1)$ to address significant changes
  - Random immigrants: Generate a set of random individuals to address medium changes
  - Replace these immigrants into the population
- More details:
  - S. Yang & R. Tinos (2007). *Int. J. of Autom. & Comp.*, 4(3): 243–254

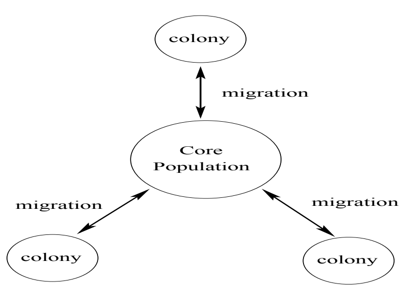
Experimental Results: Hybrid Immigrants GA



- Hybrid immigrants improve GA's performance for DOPs efficiently

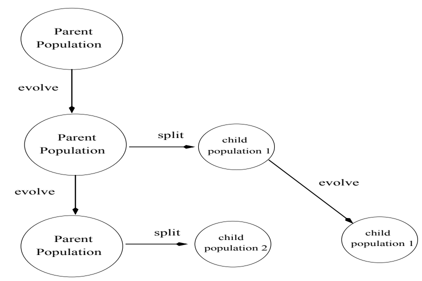
Multi-Populations: Shifting Balance

- Multi-population scheme uses co-operating sub-populations
- Shifting Balance GA (Oppacher & Wineberg, 1999):
  - A core population exploits the promising area
  - Several colonies explore the search space



Multi-Populations: Self-Organizing Scouts

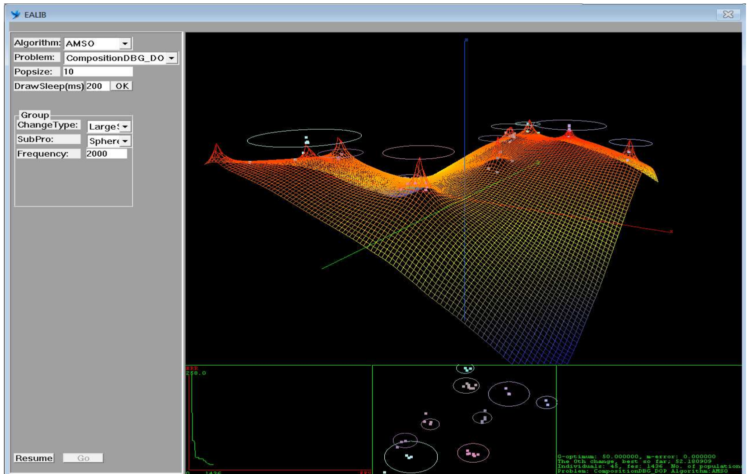
- Self-organizing scouts (SOS) GA (Branke et al., 2000)
  - The parent population explores the search space
  - A child population is split under certain conditions
  - Child populations search limited promising areas



Multi-Populations: Clustering PSO

- Particle Swarm Optimisation (PSO):
  - Motivated by the social behaviour of swarm of animals, e.g., bird flocking and fish schooling
- PSO has been used to address DOPs
- Recently, we developed a Clustering PSO (CPSO) for DOPs
  - Use a clustering technique to construct sub-swarms
  - Each sub-swarm will search among one peak quickly
  - Overlapping and convergence check
  - Strategies to response to changes
- More details:
  - S. Yang & C. Li (2010). *IEEE Trans Evol Comput*, 14(6): 93–106

Demo: Clustering PSO for DOPs



Adaptive Approaches

- Aim: Adapt operators/parameters, usually after a change
  - Hypermutation (Cobb & Grefenstette, 1993): raise the mutation rate temporarily
  - Hyper-selection (Yang & Tinos, 2008): raise the selection pressure temporarily
  - Hyper-learning (Yang & Richter, 2009): raise the learning rate for Population-Based Incremental Learning (PBIL) temporarily
- Combined: Hyper-selection and hyper-learning with re-start or hypermutation

Prediction Approaches

- For some DOPs, changes exhibit predictable patterns
- Techniques (forecasting, Kalman filter, etc.) can be used to predict
  - The location of the next optimum after a change
  - When the next change will occur and which environment may appear
- Some relevant work: see Simões & Costa (2009)

Remarks on Enhancing Approaches

- No clear winner among the approaches
- Memory is efficient for cyclic environments
- Multi-population is good for tracking competing peaks
  - The search ability will decrease if too many sub-populations
- Diversity schemes are usually useful
  - Guided immigrants may be more efficient
- Different interaction exists among the approaches
- Golden rule: balancing exploration & exploitation over time

Case Study: Dynamic Routing in MANETs – 1

- Shortest path routing problem (SPRP) in a fixed network:
  - Find the shortest path between source and destination in a fixed topology
- More and more mobile ad hoc networks (MANETs) appear where the topology keeps changing
- Dynamic SPRP (DSPRP)in MANETs:
  - Find a series of shortest paths in a series of highly-related network topologies
- We model the network dynamics as follows:
  - For each change, a number of nodes are randomly selected to sleep or wake up based on their current status

Case Study: Dynamic Routing in MANETs – 2

- A specialized GA for the DSPRP:
  - Path-oriented encoding
  - Tournament selection
  - Path-oriented crossover and mutation with repair
- Enhancements: Immigrants and memory approaches
- Experimental results:
  - Both immigrants and memory enhance GA's performance for the DSPRP in MANETs.
  - Immigrants schemes show their power in acyclic environments
  - Memory related schemes work well in cyclic environments
- More details:
  - S. Yang, H. Cheng, & F. Wang (2010). IEEE Trans SMC Part C: Appl. & Rev., 40(1): 52–63

Case Study: Dynamic Vehicle Routing – 1

- The basic Vehicle Routing Problem (VRP):
  - A number of vehicles with a fixed capacity need to satisfy the demand of all customers, starting from and finishing to the depot
- Dynamic extensions of VRP that model real-world scenarios:
  - Dynamic demands
  - Traffic factors
- Dynamic test cases can be generated using the DBGP generator (Mavrovouniotis et al., 2012)

Case Study: Dynamic Vehicle Routing – 2

- ACO algorithms with immigrants schemes are used to address the dynamic VRP with traffic factors
- Each ant constructs a solution that contains all the routes of the vehicles
- Diversity is maintained using immigrant ants
- Experimental results:
  - ACO with elitism-based immigrants outperforms other ACO algorithms
  - ACO with random immigrants is outperformed by other ACO algorithms
- Usually, ACO with guided diversity performs well for DOPs
- More details:
  - M. Mavrovouniotis & S. Yang (2012a). *EvoApplications'12*, LNCS 7248, pp. 519–528
  - M. Mavrovouniotis & S. Yang (2012b). *CEC'12*

Case Study: GA for Dynamic TSP

- Dynamic TSP:
  - 144 Chinese cities, 1 geo-stationary saterllite, and 3 mobile satellites
  - Find the path that cycles each city and satellite once with the minimum length over time
- Solver: A GA with memory and other schemes
- More details:
  - C. Li, M. Yang, & L. Kang (2006). *SEAL'06*, LNCS 4247, pp. 236–243

DE MONTFORT UNIVERSITY LEICESTER

Theoretical Development

- So far, mainly empirical studies
- Theoretical analysis has just appeared
- Runtime analysis:
  - Stanhope & Daida (1999) first analyzed a (1+1) EA on the dynamic bit matching problem (DBMP)
  - Droste (2002) analyzed the first hitting time of a (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
- 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 and Yang (2010) analyzed the properties of the XOR DOP generator based on the dynamical system approach of the GA

DE MONTFORT UNIVERSITY LEICESTER

EC for Dynamic Multi-objective Optimization

- So far, mainly dynamic single-objective optimization
- Dynamic multi-objective optimization problems (DMOPs): even more challenging
- A few studies have addressed EC for DMOPs
  - Farina et al. (2004) classified DMOPs based on the changes on the 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
  - Zhang & Qian (2011) proposed an artificial immune system to solve constrained DMOPs

DE MONTFORT UNIVERSITY LEICESTER

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?
  - The work has started, but needs much more effort
- Analysing the behaviour of EAs for DOPs:
  - Requiring more theoretical analysis tools
  - Addressing more challenging DOPs and EC methods
  - Big question: Which EC methods for what DOPs?
- Real world applications:
  - How to model real-world DOPs?

DE MONTFORT UNIVERSITY LEICESTER

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: deserves much more effort
- Real world applications: also greatly needed
  - Fields: logistics, transport, MANETs, data streams, social networks, ...

Summary

- EC for DOPs: challenging but important
- The domain is still young and active:
  - More challenges to be taken regarding approaches, theory, and applications
- More young researchers are greatly welcome!

Acknowledgements

- Two EPSRC funded projects on EC for DOPs
  - "EAs for DOPs: Design, Analysis and Applications"
    - Linked project among Brunel Univ. (Univ. of Leicester before 7/2010), Univ. of Birmingham, BT, and Honda
    - Funding/Duration: over £600K / 3.5 years (1/2008–7/2011)
    - http://www.cs.le.ac.uk/projects/EADOP/
  - "EC for Dynamic Optimisation in Network Environments"
    - Linked project among DMU, Univ. of Birmingham, RSSB, and Network Rail
    - Funding/Duration: ∼£1M / 4 years (2/2013–2/2017)
    - http://www.cci.dmu.ac.uk/research-grants/
- Research team members:
  - Research Fellows: Dr. Hui Cheng, Dr. Crina Grosan, Dr. Changhe Li, Dr. Michalis Mavrovouniotis
  - PhD students: Changhe Li, Michalis Mavrovouniotis, Lili Liu, Hongfeng Wang, Yang Yan
- Research cooperators:
  - Prof. Xin Yao, Prof. Juergen Branke, Dr. Renato Tinos, Dr. Hendrik Richter, Dr. Trung Thanh Nguyen, etc.

Relevant Information

- IEEE CIS Task Force on EC in Dynamic and Uncertain Environments
  - http://www.tech.dmu.ac.uk/∼syang/IEEE_ECIDUE.html
  - Maintained by Shengxiang Yang
- Source codes:
  - http://www.tech.dmu.ac.uk/∼syang/publications.html
- IEEE Competitions:
  - 2009 Competition on EC in Dynamic & Uncertain Environments: http://www.cs.le.ac.uk/people/syang/ECiDUE/ECiDUE-Competition09
  - 2012 Competition on EC for DOPs: http://people.brunel.ac.uk/∼csstssy/ECDOP-Competition12.html

References – 1

1. J. Branke (1999). Memory enhanced evolutionary algorithms for changing optimization problems. *CEC'99*, pp. 1875–1882
2. J. Branke (2002). *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers
3. J. Branke, E. Salihoglu & S. Uyar (2005). Towards an analysis of dynamic environments. *GECCO'05*, pp. 1433–1439
4. H.G. Cobb & J.J. Grefenstette (1993). Genetic algorithms for tracking changing environments. Proc. ICGA, pp. 523–530
5. C. Cruz, J. Gonzanlez, & D. Pelta (2011). Optimization in dynamic environments: A survey on problems, methods and measures. *Soft Comput.*, 15: 1427–1448
6. S. Droste (2002) Analysis of the (1+1) ea for a dynamically changing onemax-variant. *CEC'02*, pp. 55–60
7. M. Farina, K. Deb, & P. Amato (2004). Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Trans. Evol. Comput.*, 8(5): 425–442
8. C. Goh & K.C. Tan (2009). A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Trans. Evol. Comput.*, 13(1): 103–127
9. Y. Jin & J. Branke (2005). Evolutionary optimization in uncertain environments–A survey. *IEEE Trans. Evol. Comput.*, 9(3): 303–317
10. R.W. Morrison (2004). *Designing Evolutionary Algorithms for Dynamic Environments*. Springer
11. E.H.J. Lewis & G. Ritchie (1998). A comparison of dominance mechanisms and simple mutation on non-stationary problems. *PPSN V*, pp. 139–148.
12. R.W. Morrison & K.A. De Jong (1999). A test problem generator for non-stationary environments. *CEC'99*, pp. 2047–2053

References – 2

13. K.P. Ng & K.C. Wong (1995). A new diploid scheme and dominance change mechanism for non-stationary function optimisation. *ICGA 6*, pp. 159–166
14. T.T. Nguyen, S. Yang, & J. Branke (2012). Evolutionary dynamic optimization: A survey of the state of the art. *Swarm & Evol. Comput.*, 6: 1–24
15. F. Oppacher & M. Wineberg (1999). The Shifting balance genetic algorithm: Improving the GA in a dynamic environment. *GECCO'99*, vol. 1, pp. 504–510
16. H. Richter (2010). Evolutionary optimization and dynamic fitness landscapes: From reaction-diffusion systems to chaotic cml. *Evolutionary Algorithms and Chaotic Systems*, Springer, pp. 409–446.
17. P. Rohlfshagen, P.K. Lehre, & X. Yao (2009). Dynamic evolutionary optimisation: An analysis of frequency and magnitude of change. *GECCO'09*, pp. 1713–1720
18. S.A. Stanhope & J.M. Daida (1999). (1+1) genetic algorithm fitness dynamics in a changing environments. *CEC'99*, vol. 3, pp. 1851–1858
19. R. Tinos & S. Yang (2010) An analysis of the XOR dynamic problem generator based on the dynamical system. *PPSN XI*, LNCS 6238, Part I, pp. 274–283
20. A. Simões & E. Costa (2009). Improving prediction in evolutionary algorithms for dynamic environments. *GECCO'09*, pp. 875–882
21. K. Trojanowski & Z. Michalewicz (1999). Searching for optima in non-stationary environments. *CEC'99*, vol. 3, pp. 1843–1850
22. A.S. Uyar & A.E. Harmanci (2005). A new population based adaptive domination change mechanism for diploid genetic algorithms in dynamic environments. *Soft Comput.*, 9: 803–814
23. S. Yang (2003). Non-stationary problem optimization using the primal-dual genetic algorithm. *CEC'03*, pp. 2246–2253

References – 3

24. S. Yang, Y. Jiang, & T.T. Nguyen (2013). Metaheuristics for dynamic combinatorial optimization problems. *IMA J of Management Math.*, in press
25. S. Yang, Y.-S. Ong & Y. Jin (2007). *Evolutionary Computation in Dynamic and Uncertain Environments*. Springer
26. S. Yang & H. Richter (2009). Hyper-learning for population-based incremental learning in dynamic environments. *CEC'09*, pp. 682–689
27. S. Yang & R. Tinos (2008). Hyper-selection in dynamic environments. *CEC'08*, pp. 3185-3192
28. S. Yang & X. Yao (2005). Experimental study on population-based incremental learning algorithms for dynamic optimization problems. *Soft Comput.*, 9: 815–834
29. S. Yang & X. Yao (2008). Population-based incremental learning with associative memory for dynamic environments. *IEEE Trans Evol Comput*, 12: 542–561
30. S. Yang & X. Yao (2013). *Evolutionary Computation for Dynamic Optimization Problems*. Springer
31. K. Weicker (2003). *Evolutionary Algorithms and Dynamic Optimization Problems*. Der Andere Verlag
32. S. Zeng et al. (2006). A dynamic multi-objective evolutionary algorithm based on an orthogonal design. *CEC'06*, pp. 573–580
33. Z. Zhang & S. Qian (2011). Artificial immune system in dynamic environments solving time-varying non-linear constrained multi-objective problems. *Soft Comput.*, 15(7): 1333–1349