Evolutionary Large-Scale Global Optimization
An Introduction: Part I

Mohammad Nabi Omidvar\textsuperscript{1} Xiaodong Li\textsuperscript{2} Daniel Molina\textsuperscript{3} Antonio LaTorre\textsuperscript{4}

\textsuperscript{1}School of Computer Science, University of Birmingham, UK
\textsuperscript{2}School of Science, RMIT University, Australia,
\textsuperscript{3}DASCI Andalusian Institute of Data Science, University of Granada, Spain,
\textsuperscript{4}Universidad Politécnica de Madrid, Spain,
Outline

1. Introduction: Large Scale Global Optimization
2. Approaches to Large-Scale Optimization
3. Variable Interaction: Definitions and Importance
4. Interaction Learning: Exploiting Modularity
5. Conclusion
6. Questions
Optimization

\[
\begin{align*}
\min \ f(x), \ x &= (x_1, \ldots, x_n) \in \mathbb{R}^n \\
\text{s.t.:} \ g(x) &\leq 0 \\
h(x) &= 0
\end{align*}
\]

Can be converted to unconstrained optimization using:
- Penalty method;
- Lagrangian;
- Augmented Lagrangian.

Our focus is unconstrained optimization. We must learn how to walk before we can run.
Large Scale Global Optimization (LSGO)

How large is large?

- The notion of large-scale is not fixed.
- Changes over time.
- Differs from problem to problem.
- The dimension at which existing methods start to fail.

State-of-the-art (EC)

- Binary: \(\approx 1\) billion \([a]\).
- Integer (linear): \(\approx 1\) billion \([b],[c]\).
- Real: \(\approx 1000\)–\(5000\).


Large Scale Global Optimization: Applications

Why large-scale optimization is important?

- Growing applications in various fields.
  - Target shape design optimization [a].
  - Satellite layout design [b].
  - Parameter estimation in large scale biological systems [c].
  - Seismic waveform inversion [d].
  - Parameter calibration of water distribution systems [e].
  - Vehicle routing [f].


Large Scale Global Optimization: Research

![Scopus Graph]

Copyright © 2017 Elsevier B.V. All rights reserved. Scopus® is a registered trademark of Elsevier B.V.
Large Scale Global Optimization: Research

Scopus

Copyright © 2017 Elsevier B.V. All rights reserved. Scopus® is a registered trademark of Elsevier B.V.
The Challenge of Large Scale Optimization

Why is it difficult?

- Exponential growth in the size of search space (**curse of dimensionality**).

Research Goal

- Improving search quality (get to the optimal point).
- Improving search efficiency (get there fast).
Initialization
Sampling and Variation Operators
Approximation and Surrogate Modeling
Local Search and Memetic Algorithms
Decomposition and Divide-and-Conquer
Parallelization (GPU, CPU)
Hybridization
Initialization Methods

- Study the importance of initialization methods [1] in large-scale optimization.

Initialization Methods

- Inconclusive evidence for or against initialization methods:
  - Uniform design works worse than RNG, while good-lattice point and opposition-based methods perform better [1].
  - Another study showed that population size has a more significant effect than the initialization [2].
  - Achieving uniformity is difficult in high-dimensional spaces [3].
  - Yet another study suggest comparing average performances may not reveal the effect of initialization [4].

- Shortcomings:
  - It is difficult to isolate the effect of initialization.
  - Different effect on different algorithms (mostly tested on DE).
  - Numerous parameters to study.

---


Sampling and Variation Operators

- Opposition-based sampling [1]
- Center-based sampling [2].
- Quantum-behaved particle swarm [3].
- Competitive Swarm Optimizer [4].
- Social learning PSO [5].
- Mutation operators [6], [7].


Approximation Methods and Surrogate Modeling

- High-Dimensional Model Representation (HDMR) [1].
- Radial Basis Functions [2].
- Kriging and Gradient-Enhanced Kriging Metamodels [3].
- Piecewise Polynomial (Spline) [4].
- Turning large-scale problems into expensive optimization problems [5].

---


Local Search and Memetic Algorithms

- Multiple Trajectory Search (MTS) [1].
- Memetic algorithm with local search chaining [2].
  - MA-SW-Chains [3].
  - MA-SSW-Chains [4].
- Multiple offspring sampling (MOS) [5], [6].

---


Parallelization

- Algorithms capable of parallelization [1], [2].
- GPU [3], [4].
- CPU/OpenMP [5].


Hybridization (The best of both worlds)

- **Rationale:** benefiting from unique features of different optimizers.
  - EDA+DE: [1].
  - PSO+ABC: [2].
  - Different DE variants: JADE+SaNSDE [3].
  - PSO+ACO [4].
  - Minimum Population Search+CMA-ES [5].

---


Decomposition Methods

- Divide-and-conquer
Variable Interaction, Linkage, Epistasis

What is variable interaction?

- Genetics: two genes are said to interact with each other if they collectively represent a feature at the phenotype level.
- The extent to which the fitness of one gene can be suppressed by another gene.
- The extent to which the value taken by one gene activates or deactivates the effect of another gene.

Why variable interaction?

- The effectiveness of optimization algorithms is affected by how much they take variable interaction into account.
- Also applies to classic mathematical programming methods.
Variable Interaction, Linkage, Epistasis

**Illustrative Example**

- $f(x, y) = x^2 + \lambda_1 y^2$
- $g(x, y) = x^2 + \lambda_1 y^2 + \lambda_2 xy$
Definitions

Variable Interaction

A variable $x_i$ is separable or does not interact with any other variable iff:

$$\arg \min_{x} f(x) = \left( \arg \min_{x_i} f(x), \arg \min_{\forall x_j, j \neq i} f(x) \right),$$

where $x = (x_1, \ldots, x_n)^\top$ is a decision vector of $n$ dimensions.

Partial Separability

A function $f(x)$ is partially separable with $m$ independent subcomponents iff:

$$\arg \min_{x} f(x) = \left( \arg \min_{x_1} f(x_1, \ldots), \ldots, \arg \min_{x_m} f(\ldots, x_m) \right),$$

$x_1, \ldots, x_m$ are disjoint sub-vectors of $x$, and $2 \leq m \leq n$.

Note: a function is fully separable if sub-vectors $x_1, \ldots, x_m$ are 1-dimensional (i.e., $m = n$).
Definitions

**Full Nonseparability**

A function $f(x)$ is fully non-separable if every pair of its decision variables interact with each other.

**Additive Separability**

A function is *partially additively separable* if it has the following general form:

$$f(x) = \sum_{i=1}^{m} f_i(x_i),$$

where $x_i$ are mutually exclusive decision vectors of $f_i$, $x = (x_1, \ldots, x_n)^\top$ is a global decision vector of $n$ dimensions, and $m$ is the number of independent subcomponents.
Effect of Variable Interaction (1)

Sampling and Variation Operators:

- GAs: inversion operator to promote tight linkage [1].
  - Increasing the likelihood of placing linked genes close to each other to avoid disruption by crossover.
  - Rotation of the landscape has a detrimental effect on GA [2].

- The need for rotational invariance:
  - Model Building Methods:
    - Estimation of Distribution Algorithms and Evolutionary Strategies: Covariance Matrix Adaptation.
    - Bayesian Optimization: Bayesian Networks.
  - DE’s crossover is not rotationally invariant.
  - PSO is also affected by rotation [3].


Effect of Variable Interaction (2)

1. Approximation and Surrogate Modelling:
   - Should be able to capture variable interaction.
   - Second order terms of HDMR.

2. Local Search and Memetic Algorithms:
   - What subset of variables should be optimized in each iteration of local search?
   - Coordinate-wise search may not be effective. Memetics perform well on separable functions! A coincidence?!

3. Decomposition and Divide-and-Conquer:
   - Interacting variables should be placed in the same component.
Linkage Learning and Exploiting Modularity

- Implicit Methods:
  - In EC:
    - Estimation of Distribution Algorithms
    - Bayesian Optimization: BOA, hBOA, Linkage Trees
    - Adaptive Encoding, CMA-ES
  - Classic Optimization:
    - Adaptive Coordinate Descent

- Explicit Methods:
  - In EC:
    - Random Grouping
    - Statistical Correlation-Based Methods
    - Delta Grouping
    - Meta Modelling
    - Monotonicity Checking
    - Differential Grouping
  - Classic Optimization
    - Block Coordinate Descent
Implicit Methods

Scaling Up EDAs:

- Model Complexity Control [1].
- Random Matrix Projection [2].
- Use of mutual information [3].
- Cauchy-EDA [4].


Implicit Methods

- **Scaling up CMA-ES:**
  - CC-CMA-ES [1].
  - sep-CMA-ES [2]
  - Reducing space complexity:
    - L-CMA-ES [3].
    - LM-CMA [4].

---


Scalability issues of EDAs

- Accurate estimation requires a large sample size which grows exponentially with the dimensionality of the problem [1].
- A small sample results in poor estimation of the eigenvalues [2].
- The cost of sampling from a multi-dimensional Gaussian distribution increases cubically with the problem size [3].

---


Random Projection EDA [1]

Explicit Methods

- A large problem can be subdivided into smaller and simpler problems.
- Dates back to René Descartes (*Discourse on Method*).
- Has been widely used in many areas:
  - Computer Science: Sorting algorithms (quick sort, merge sort)
  - Optimization: Large-scale linear programs (Dantzig)
  - Politics: Divide and rule (In *Perpetual Peace* by Immanuel Kant: *Divide et impera* is the third political maxims.)

Acknowledgement: the above image is obtained from: http://draininbrain.blogspot.com.au/
Decomposition in EAs: Cooperative Co-evolution [1]

CC is a Framework

CC as a scalability agent:

- CC is not an optimizer.
- Requires a component optimizer.
- CC coordinates how the component optimizer is applied to components.
- A scalability agent.
Challenges of CC

Main Questions

1. How to decompose the problem?
2. How to allocate resources?
3. How to coordinate?
The Decomposition Challenge

How to decompose?
- There are many possibilities.
- Which decomposition is the best?

Optimal decomposition
- It is governed by the interaction structure of decision variables.
- An optimal decomposition is the one that minimizes the interaction between components.
Survey of Decomposition Methods

- Uninformed Decomposition [1]
  - \( n \) 1-dimensional components (the original CC)
  - \( k \) \( s \)-dimensional components.

- Random Grouping [2]

- Statistical Correlation-Based Methods

- Delta Grouping [3]

- Meta Modelling [4]

- Monotonicity Checking [5]

- Differential Grouping [6]


Illustrative Example (Canonical CC)

Figure: Variable interaction of a hypothetical function.

$n$ 1-dimensional components:

- $C_1: \{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}, \{x_6\}, \{x_7\}$
- $C_2: \{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}, \{x_6\}, \{x_7\}$
- ...
- $C_c: \{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}, \{x_6\}, \{x_7\}$
Illustrative Example (fixed $k$ $s$-dimensional)

$k$ $s$-dimensional ($k = 2$, $s = 4$):
- $C_1$: $\{x_1, x_2, x_3, x_4\}$, $\{x_5, x_6, x_7\}$
- $C_2$: $\{x_1, x_2, x_3, x_4\}$, $\{x_5, x_6, x_7\}$
- ...
- $C_c$: $\{x_1, x_2, x_3, x_4\}$, $\{x_5, x_6, x_7\}$

Figure: Variable interaction of a hypothetical function.
Illustrative Example (Random Grouping)

Random Grouping ($k = 2, s = 4$):
- $C_1$: $\{x_2, x_3, x_6, x_5\}, \{x_7, x_1, x_4\}$
- $C_2$: $\{x_3, x_4, x_1, x_2\}, \{x_6, x_7, x_5\}$
- ...
- $C_c$: $\{x_1, x_5, x_6, x_7\}, \{x_2, x_4, x_3\}$

Figure: Variable interaction of a hypothetical function.
Random Grouping

Theorem

Given $N$ cycles, the probability of assigning $v$ interacting variables $x_1, x_2, \ldots, x_v$ into one subcomponent for at least $k$ cycles is:

$$P(X \geq k) = \sum_{r=k}^{N} \binom{N}{r} \left( \frac{1}{m^{v-1}} \right)^r \left( 1 - \frac{1}{m^{v-1}} \right)^{N-r}$$

where $N$ is the number of cycles, $v$ is the total number of interacting variables, $m$ is the number of subcomponents, and the random variable $X$ is the number of times that $v$ interacting variables are grouped in one subcomponent.
Random Grouping

Example

Given $n = 1000$, $m = 10$, $N = 50$ and $v = 4$, we have:

$$P(X \geq 1) = 1 - P(X = 0) = 1 - \left(1 - \frac{1}{10^3}\right)^{50} = 0.0488$$

which means that over 50 cycles, the probability of assigning 4 interacting variables into one subcomponent for at least 1 cycle is only 0.0488. As we can see this probability is very small, and it will be even less if there are more interacting variables.
Figure: Increasing $\nu$, the number of interacting variables will significantly decrease the probability of grouping them in one subcomponent, given $n = 1000$ and $m = 10$. 
**Figure:** Increasing $N$, the number of cycle increases the probability of grouping $v$ number of interacting variables in one subcomponent.
Illustrative Example (Informed with Fixed Groups)

Figure: Variable interaction of a hypothetical function.

• Delta Grouping \((k = 2, s = 4)\):
  - \(C_1\): \(
    \{x_1, x_5, x_2, x_4\}, \{x_3, x_6, x_7\}\)
  - \(C_2\): \(
    \{x_3, x_5, x_6, x_7\}, \{x_1, x_2, x_4\}\)
  - ...
  - \(C_c\): \(
    \{x_3, x_6, x_1, x_4\}, \{x_2, x_5, x_7\}\)
Delta Grouping
Informed Decompositions with Fixed Groups

- Adaptive Variable Partitioning [1].
- Delta Grouping [2].
- Min/Max-Variance Decomposition (MiVD/MaVD) [3].
  ▶ Sorts the dimensions based on the diagonal elements of the covariance matrix in CMA-ES.
- Fitness Difference Partitioning [4], [5], [6].

---


Informed Decompositions with Variable Groups

- Multilevel Grouping: MLCC [1], MLSoft [2].
- Adaptive Variable Partitioning 2 [3].
- 4CDE [4].
- Fuzzy Clustering [5].


Illustrative Example (Exact Methods)

Figure: Variable interaction of a hypothetical function.

Differential Grouping and Variable Interaction Learning:

- $C_1$: $\{x_1, x_2, x_4\}, \{x_3, x_5, x_6, x_7\}$
- $C_2$: $\{x_1, x_2, x_4\}, \{x_3, x_5, x_6, x_7\}$
- ...
- $C_c$: $\{x_1, x_2, x_4\}, \{x_3, x_5, x_6, x_7\}$
Monotonicity Check

\[ \exists \mathbf{x}, x'_i, x'_j : f(x_1, \ldots, x_i, \ldots, x_j, \ldots, x_n) < f(x_1, \ldots, x'_i, \ldots, x_j, \ldots, x_n) \land \\
\quad f(x_1, \ldots, x_i, \ldots, x'_j, \ldots, x_n) > f(x_1, \ldots, x'_i, \ldots, x'_j, \ldots, x_n) \]
Monotonicity Check (Algorithms)

- Linkage Identification by Non-Monotonicity Detection [1]
- Adaptive Coevolutionary Learning [2]
- Variable Interaction Learning [3]
- Variable Interdependence Learning [4]
- Fast Variable Interdependence [5]


Differential Grouping [1]

Theorem

Let \( f(x) \) be an additively separable function. \( \forall \, a, b_1 \neq b_2, \delta \in \mathbb{R}, \delta \neq 0 \), if the following condition holds

\[
\Delta_{\delta, x_p}[f](x)|_{x_p=a, x_q=b_1} \neq \Delta_{\delta, x_p}[f](x)|_{x_p=a, x_q=b_2}, \tag{5}
\]

then \( x_p \) and \( x_q \) are non-separable, where

\[
\Delta_{\delta, x_p}[f](x) = f(\ldots, x_p + \delta, \ldots) - f(\ldots, x_p, \ldots), \tag{6}
\]

refers to the forward difference of \( f \) with respect to variable \( x_p \) with interval \( \delta \).

---

Figure: $f(x_1, x_2) = x_1^2 + x_2^2$
Figure: $f(x_1, x_2) = x_1^2 + x_2^2$
\[ \Delta_1 = f(p_1) - f(p_2) \]

Figure: \[ f(x_1, x_2) = x_1^2 + x_2^2 \]
\[ \Delta_1 = f(p_1) - f(p_2) \]

\[ p_1 = (x_1, x_2') \]

\[ p_2 = (x_1', x_2) \]

\[ f(x_1, x_2) = x_1^2 + x_2^2 \]

Figure: \( f(x_1, x_2) = x_1^2 + x_2^2 \)
\[ \Delta_1 = f(p_1) - f(p_2) \]

Figure: \( f(x_1, x_2) = x_1^2 + x_2^2 \)
\[ \Delta_2 = f(p_1) - f(p_2) \]

\[ \Delta_1 = f(p_1) - f(p_2) \]

\[ p_1 = (x_1, x'_2) \]

\[ p_2 = (x'_1, x_2) \]

*Figure: \( f(x_1, x_2) = x_1^2 + x_2^2 \)
\[ \Lambda_{k,12} = |\Delta_1 - \Delta_2| > 0 \]

\[ \Delta_1 = f(p_1) - f(p_2) \]

\[ \Delta_2 = f(p_1) - f(p_2) \]

\[ \Rightarrow x_1 \text{ and } x_2 \text{ are nonseparable} \]

Figure: \( f(x_1, x_2) = x_1^2 + x_2^2 \)
Separability ⇒ \( \Delta_1 = \Delta_2 \)

Assuming:

\[
f(x) = \sum_{i=1}^{m} f_i(x_i)
\]

We prove that:

Separability ⇒ \( \Delta_1 = \Delta_2 \)

By contraposition (\( P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P \)):

\( \Delta_1 \neq \Delta_2 \Rightarrow \) non-separability

or

\(|\Delta_1 - \Delta_2| > \epsilon \Rightarrow \) non-separability
The Differential Grouping Algorithm

Detecting Non-separable Variables

\[ |\Delta_1 - \Delta_2| > \epsilon \Rightarrow \text{non-separability} \]

Detecting Separable Variables

\[ |\Delta_1 - \Delta_2| \leq \epsilon \Rightarrow \text{Separability (more plausible)} \]
Example

Consider the non-separable objective function \( f(x_1, x_2) = x_1^2 + \lambda x_1 x_2 + x_2^2 \), \( \lambda \neq 0 \).

\[
\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 + \lambda x_2.
\]

This clearly shows that the change in the global objective function with respect to \( x_1 \) is a function of \( x_1 \) and \( x_2 \). By applying the Theorem:

\[
\Delta_{\delta, x_1}[f] = [(x_1 + \delta)^2 + \lambda(x_1 + \delta)x_2 + x_2^2] - [x_1^2 + \lambda x_1 x_2 + x_2^2]
= \delta^2 + 2\delta x_1 + \lambda x_2 \delta.
\]
Differential Grouping vs CCVIL

Figure: Detection of interacting variables using differential grouping and CCVIL on different regions of a 2D Schwefel Problem 1.2.
Differential Grouping Family of Algorithms

- Linkage Identification by Non-linearity Check (LINC, LINC-R) [1]
- Differential Grouping (DG) [2]
- Global Differential Grouping (GDG) [3]
- Improved Differential Grouping (IDG) [4]
- eXtended Differential Grouping (XDG) [5]
- Graph-based Differential Grouping (gDG) [6]
- Fast Interaction Identification [7]
- Recursive Differential Grouping (RDG1 and RDG2) [8]


Shortcomings of Differential Grouping

- Cannot detect the overlapping functions.
- Slow if all interactions are to be checked.
- Requires a threshold parameter ($\epsilon$).
- Can be sensitive to the choice of the threshold parameter ($\epsilon$).
Direct/Indirect Interactions

Indirect Interactions

In an objective function \( f(x) \), decision variables \( x_i \) and \( x_j \) interact directly (denoted by \( x_i \leftrightarrow x_j \)) if

\[
\exists a : \left. \frac{\partial f}{\partial x_i \partial x_j} \right|_{x=a} \neq 0,
\]

decision variables \( x_i \) and \( x_j \) interact indirectly if

\[
\frac{\partial f}{\partial x_i \partial x_j} = 0,
\]

and there exists a set of decision variables \( \{x_{k1}, ..., x_{ks}\} \) such that \( x_i \leftrightarrow x_{l1}, ..., x_{ks} \leftrightarrow x_j \).
Efficiency vs Accuracy

Saving budget at the expense of missing overlaps:
- eXtended Differential Grouping [1].
- Fast Interdependence Identification [2].

Figure: The interaction structures represented by the two graphs cannot be distinguished by XDG.


Differential Grouping 2: Improving Accuracy [1]

DG2 Estimates the computational round-off errors as the threshold value:

\[ e_{\text{inf}} := \gamma_2 \max \{ f(x) + f(y'), f(y) + f(x') \} \]  
\[ e_{\text{sup}} = \gamma \sqrt{n} \max \{ f(x), f(x'), f(y), f(y') \} \]  

- \( \lambda < e_{\text{inf}} \rightarrow \text{separable} \);
- \( \lambda > e_{\text{sup}} \rightarrow \text{non-separable} \).

Otherwise

\[ \epsilon = \frac{\eta_0}{\eta_0 + \eta_1} e_{\text{inf}} + \frac{\eta_1}{\eta_0 + \eta_1} e_{\text{sup}}, \]  

- \( \lambda < \epsilon \rightarrow \text{separable} \);
- \( \lambda \geq \epsilon \rightarrow \text{non-separable} \).

DG2 Error Analysis

\[ \lambda = |\Delta_1 - \Delta_2| \]

Genuine zeros

\[ e_{\inf} \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ e_{\sup} \]

Genuine non-zeros
Differential Grouping 2: Improving Efficiency

Figure: Geometric representation of point generation in DG2 for a 3D function.

\[ x_1 \leftrightarrow x_2: \Delta^{(1)} = f(a', b, c) - f(a, b, c), \Delta^{(2)} = f(a', b', c) - f(a, b', c) \]

\[ x_1 \leftrightarrow x_3: \Delta^{(1)} = f(a', b, c) - f(a, b, c), \Delta^{(2)} = f(a', b, c') - f(a, b, c') \]

\[ x_2 \leftrightarrow x_3: \Delta^{(1)} = f(a, b', c) - f(a, b, c), \Delta^{(2)} = f(a, b', c') - f(a, b, c') \]

\[ \lambda = |\Delta^{(1)} - \Delta^{(2)}| \]
Minimum Evaluations

The minimum number of unique function evaluations in order to detect the interactions between all pairs of variables is

\[ h(n) \geq \frac{n(n + 1)}{2} + 1. \]  

Improving efficiency beyond the given lower bound is impossible unless:

- Sacrifice on the accuracy (partial variable interaction matrix);
- and/or
- Extending the DG theorem.
Fast Interaction Identification (FII) [1]
- identifying separable variables by checking the interaction between a single variable with remaining variables.
- examining pairwise interaction for non-separable variables.

Recursive Differential Grouping (RDG) [2] (\(O(n \log(n))\))
- examining interaction between two variable subsets.
- using a recursive procedure to group variables.

---


Variants of RDG

1. RDG2 [1]: Combining the efficiency of RDG and accuracy of DG2.

2. RDG3 [2]: extending RDG2 for decomposing overlapping problems.


Benchmark Suites

- CEC’2005 Benchmark Suite (non-modular)
- CEC’2008 LSGO Benchmark Suite (non-modular)
- CEC’2010 LSGO Benchmark Suite
- CEC’2013 LSGO Benchmark Suite
Challenges of CC

Main Questions

1. How to decompose the problem?
2. How to allocate resources?
3. How to coordinate?
The Imbalance Problem

- Non-uniform contribution of components.

### Imbalanced Functions

\[
f(x) = \sum_{i=1}^{m} w_i f_i(x_i),
\]

\[
w_i = 10^{s \mathcal{N}(0,1)},
\]
The Imbalance Problem (2)
Contribution-Based Cooperative Co-evolution (CBCC)

Types of CC
- CC: round-robin optimization of components.
- CBCC: favors components with a higher contribution.
  - Quantifies the contribution of components.
  - Optimizes the one with the highest contribution.

How to Quantify the Contribution
- For quantification of contributions a relatively accurate decomposition is needed.
- Changes in the objective value while other components are kept constant.
(c) Round-Robin CC

(d) Contribution-Based CC
Contribution-Aware Algorithms

- Contribution-Based Cooperative Co-evolution (CBCC) [1], [2].
- Bandit-based Cooperative Coevolution (BBCC) [3].
- Incremental Cooperative Coevolution [4]
- Multilevel Framework for LSGO [5]

---


Some Auxiliary Topics

- Variable Interaction and Constraint Handling [1], [2], [3]
- Large-Scale Multiobjective Optimization
- Available Benchmark Suites

---


Variable Interaction and Constraint Handling [1]

\[
\begin{align*}
\min f(x) & = x_1^2 x_2 + 4x_5 \\
s.t \ g_1(x) & = \frac{x_3}{x_2} + \sqrt{x_5} - x_6 \leq 0 \\
g_2(x) & = x_1 - x_2 e^{-x_6} \leq 0
\end{align*}
\]

\[
\Theta_0 = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}, \quad \Theta_1 = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}, \quad \Theta_2 = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix},
\]

\[
\Theta_{\text{glob}} = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
\end{pmatrix}.
\]

Large-Scale Multiobjective Optimization

Large-scale multiobjective optimization is growing popularity:

- **Benchmark development and analysis:**
  - Development of a benchmark [1].
  - Analysis of the existing benchmarks [2].

- **Algorithm development:**
  - Exploiting modularity using CC [3], [4], [5], [6].
  - Problem transformation [7].

---


Analysis of ZDT

\[
\begin{pmatrix}
0 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 0
\end{pmatrix}
\]

\(x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6\)

**Figure:** Variable interaction structures of the \(f_2\) function of ZDT test suite [1].

Analysis of DTLZ1-DTLZ4

Figure: Variable interaction graphs of DTLZ1 to DTLZ4.

**Proposition 1**

For DTLZ1 to DTLZ4, \( \forall f_i, i \in \{1, \cdots, m\} \), we divide the corresponding decision variables into two non-overlapping sets: \( x_I = (x_1, \cdots, x_\ell)^T, \ell = m - 1 \) for \( i \in \{1, 2\} \) while \( \ell = m - i + 1 \) for \( i \in \{3, \cdots, m\} \); and \( x_{II} = (x_m, \cdots, x_n)^T \). All members of \( x_I \) not only interact with each other, but also interact with those of \( x_{II} \); all members of \( x_{II} \) are independent from each other.
Analysis of DTLZ5-DTLZ7

Figure: Variable interaction graphs of DTLZ5 and DTLZ6.

**Proposition 2**

For DTLZ5 and DTLZ6, \( \forall f_i, i \in \{1, \cdots, m\} \), we divide the corresponding decision variables into two non-overlapping sets: \( x_I = (x_1, \cdots, x_\ell)^T, \ell = m - 1 \) for \( i \in \{1, 2\} \) while \( \ell = m - i + 1 \) for \( i \in \{3, \cdots, m\} \); and \( x_{II} = (x_m, \cdots, x_n)^T \). For \( f_i \), where \( i \in \{1, \cdots, m - 1\} \), all members of \( x_I \) and \( x_{II} \) interact with each other; for \( f_m \), we have the same interaction structure as DTLZ1-DTLZ4.

**Proposition 3**

All objective functions of DTLZ7 are fully separable.
Decomposition Based Large-Scale EMO

Figure: Image taken from [1]

Weighted Optimization Framework (WOF) [1], [2]

Figure: Weighted Optimization Framework


Some Future Directions (I)

- What if the components have overlap?
- Differential group is time-consuming. Is there a more efficient method?
- Do we need to get 100% accurate grouping? What is the relationship between grouping accuracy and optimality achieved by a CC algorithm?
CC for combinatorial optimization, e.g.,


However, every combinatorial optimization problem has its own characteristics. We need to investigate CC for other combinatorial optimization problems.
Learning variable interdependencies is a strength of estimation of distribution algorithms (EDAs), e.g.,


Interestingly, few work exists on scaling up EDAs.
LSGO Resources

- There is an IEEE Computational Intelligence Society (CIS) Task Force on LSGO:
- LSGO Repository: http://www.cercia.ac.uk/projects/lsgo
Acknowledgement

Thanks to

- Dr. Yuan Sun from RMIT University, for assistance in revising the slides;
- Dr. Ata Kaban and Dr. Momodou L. Sanyang for allowing us to use some figures from their publications.
Questions

Thanks for your attention!

$q_0 \quad q_0 \quad q \quad q \quad ?$