How Powerful/Efficient Is Your Evolutionary Algorithm

Xin Yao [a] (x.yao@cs.bham.ac.uk)

The Center of Excellence for Research in Computational Intelligence and Applications (CERCIA)

The University of Birmingham

Egbaston, Birmingham, U.K.

http://www.cercia.ac.uk

[a] Joint work with Jun He.

The Answer

Very Powerful and Efficient!

Evidences

Actual quotes from published papers:

- “*A GA is applied to* **solve** *a maximum flow problem ...*”
- “*We use a GA to produce* **very high quality** *solutions to the travelling salesman problem*”
- “*The results indicate that the proposed algorithm is able to arrive at* **high quality** *solution in a* **relatively short** *time*”
- “*We describe results computing* **approximate** *solutions to Set Partitioning Problems using GAs*”

Note: Boldface added by XY.

Here Comes the ‘Trouble’

- Most likely to be one of your colleagues in the same department, who is envious of your visit to Hawaii (CEC’02) and Australia (CEC’03).
- “Hmmm ... Evolutionary algorithms?! Aren’t they just some random algorithms for people who can think deterministically?
- ... how can an algorithm that produces a different result every time it runs possibly work?”

Note: The characters described in this film (slide) is entirely fictional.

Back to Reality

Although theory has been lagging behind practice in evolutionary computation, there have been many progresses made in the theory of evolutionary computation recently, e.g., 29 papers in “Theory of Evolutionary Computation” sessions at CEC’04.

One thread of the theoretical work:

- Convergence
- Convergence Rate
- Computational Complexity (Scalability): the relationship between computation time and the problem size.

Computational Complexity

- The ‘Trouble’: “I know. I know. It’s that big-O thing, isn’t it? ...”
- Unlike the analysis of deterministic algorithms, which is usually about the worst case, we need to consider the mean (average) of the first hitting time in our case.
- Many existing tools used in analysing complexity are difficult to be applied to evolutionary algorithms.
- We need ‘new’ analytical tools and a unified framework for analysing different EAs for different problems, and for convergence, computation time, characterisation of EA-hardness/-easiness, ...

Outline

- Analytical Tools and Approaches
- Drift Analysis
- Convergence of EAs
- Computation Time of EAs
- Why Populations
- Characteristics of EA-Easy/-Hard Problems
- Discussions and Conclusions

Acknowledgement

An incomplete list in random order:

- Convergence and Convergence Rate: G. Rudolph, T. Back, H.-G. Beyer, H. Mühlenbein, J.A. Lozano, C. Reeves, J. Rowe, M.D. Vose, L. D. Whitley, D. Goldberg, A.H. Wright, E. A. Nix, D. Fogel, T.E Davis, J. Suzuki, L. Schmit, ...
- Computation Times and Time Complexity: I. Wegener, S. Droste and T. Jansen, R. Salomon, J. Garnier, L. Kallel, M. Schoenauer, B. Naudts, B. K. Ambati, J. Ambati, D. Fogel, D. Goldberg ...

Popular Approaches

- Markov chain Approach: a widely used approach to model the behavior of EAs.
  - A population = a state in the state (probability) space.
  - Generation of a new population = state (probability) transition
  - Populations sequence $\{\xi_t; t = 0, 1, \cdots\}$ = a Markov chain.
- Convergence: Many good results have been obtained.
- Convergence rate: Relatively few results in comparison with convergence.

Markov Chain Approach to Computation time

- Let $\boldsymbol{\tau} = (\tau_i)$ be the first hitting time to optimal solutions when starting from a non-optimal population i; $\mathbf{P}$ is the transition probability matrix written in the form:

$$\mathbf{P} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{pmatrix},$$

  where $\mathbf{I}$ is the unit matrix, then the first hitting time satisfies (He and Yao, 2003) [a]

$$(\mathbf{I}' - \mathbf{T})\boldsymbol{\tau} = \mathbf{1}.$$

- The above linear system is not easy to solve directly and analytically.

[a] J. He and X. Yao. “Towards an analytic framework for analysing the computation time of evolutionary algorithms.” Art. Intell. 145(1-2):59-97, 2003.

Less Popular but More Useful Approaches

- Martingale Approach: less commonly used, but seems to be more useful and powerful analytical tools to us.
- A stochastic process $\{X_t; t = 0, 1, \cdots\}$ is called a martingale, if
  1. Expectation $E[X_{t+1} \mid X_t] < \infty$, $\forall t \geq 0$,
  2. Expectation $E[X_{t+1} \mid X_t] = X_t$, $\forall t \geq 0$.
- A super-martingale is defined similarly, except (2) is replaced by

$$E[X_{t+1} \mid X_t] \leq X_t, \forall t \geq 0,$$

- A sub-martingale is defined with (2) is replaced by

$$E[X_{t+1} \mid X_t] \geq X_t, \forall t \geq 0.$$

Super-martingale for EAs

- Define a distance function $d(x)$ to measure how good or how far a population x is from the optimal solution;
- If the mean local performance at each generation:

$$E[d(\xi_t) - d(\xi_{t+1})] \geq 0,$$

  then the sequence $\{d(\xi_t); t = 0, 1, \cdots\}$ is a super-martingale;
- Martingale theory can be used to analyse convergence, mean first hitting time, EA-hardness, ...

How? Drift Analysis as a General Framework

Consider the population sequence $\{\xi_t; t = 0, 1, \cdots, n\}$ on a probability space, the general approach to analysing EAs using drift analysis can be summarised as follows:

- Define a **distance function** $d(x)$ to measure the distance between a population x and the optimal solution.
- Estimate the **one-step mean drift** of ξ_t towards the optimal solution:

$$E[d(\xi_t) - d(\xi_{t+1})].$$

- Derive some properties of the sequence, e.g., the first hitting time, convergence, convergence rate, etc., through the mean drift.

An Example: Conference Dinner Tonight

- XY walks from the dinning room to his room.
  - The dinning room is n meters away from the bedroom.
  - XY always moves forward 1m in each step.
- Question: how many steps does XY need to reach his bedroom?

Simple and Correct Answer

n steps!

Making Simple Things Complicated

- Define a **distance function** to measure the distance between a point x and the bedroom,

$$d(x) = x, \quad x \in \{0, \cdots, n\}.$$

- Estimate the 'speed' (drift) (ξ_t is his position at t-step):

$$d(\xi_t) - d(\xi_{t+1}) = \begin{cases} 0, & \xi_t = 0, \\ 1, & \xi_t \in \{1 \cdots, n\}. \end{cases}$$

- Then the first hitting time τ to the bedroom:

$$\tau = \frac{\textbf{distance}}{\textbf{drift}} = n.$$

An Example: After a Few Drinks

- The impact of the few drinks:
  - At the bar, he always leaves for the bedroom;
  - In between the dinning room and bedroom, he moves 1 meter in each step, forward with probability 0.6, backward with probability 0.4.
- Question: how many steps does he need to go to his bedroom?

Getting Back After a Few Drinks

- Define the same distance function $d(x) = x$.
- Estimate his speed (drift) (ξ_t is his position at t-step):

$$d(\xi_t) - d(\xi_{t+1}) = \begin{cases} 1, & \text{if } \xi_t = n; \\ 1, & \text{if } 0 < \xi_t < n, \quad \text{with probability } 0.6, \\ -1, & \text{if } 0 < \xi_t < n, \quad \text{with probability } 0.4. \end{cases}$$

- Compute the mean speed/drift before arriving at the bedroom:

$$1 \geq E[d(\xi_t) - d(\xi_{t+1})] \geq 0.6 \times 1 - 0.4 \times 1 = 0.2.$$

- The mean first hitting time to the bedroom is:

$$n \leq E[\tau] \leq \frac{n}{0.2} = 5n.$$

An Example: Too Many Drinks

- After drink situation:
  - He always heads towards his bedroom.
  - In between the dinning room and the bedroom, he moves 1 meter in each step, forward with probability 0.4 and backward with probability 0.6.
- Question: how many steps does he need to arrive at his bedroom?

Journey Towards the Bedroom

- Define a new distance function:

$$d(x) = \begin{cases} 0, & x = 0, \\ \sum_{i=0}^{x} 1.5^{n-i}, & x \in \{1, \cdots, n\}. \end{cases}$$

- Speed/drift in between the dinning room and bedroom ($\xi_t = x$ is his position at t-step),

$$d(\xi_t) - d(\xi_{t+1}) = \begin{cases} 1.5^{n-x}, & \text{with probability } 0.4, \\ -1.5^{n-(x+1)}, & \text{with probability } 0.6. \end{cases}$$

- Mean speed/drift before arriving at his bedroom:

$$1 \geq E[d(\xi_t) - d(\xi_{t+1})] \geq 0.4 \times 1.5^{n-x} - 0.6 \times 1.5^{n-(x+1)} = 0.$$

- The mean first hitting time τ to home: $E[\tau] \geq \sum_{i=0}^{n} 1.5^{n-i}$.

Summary from the Examples

Given a problem and an EA, we can analyse the EA as follows:

- Define a distance function to measure how far or how good a population is away from the optimal solution;
- Estimate the mean drift towards the optimal solution;
- Prove the convergence, convergence rate, computation time, ...

Convergence

Rudolph's work in 1994 [a]

- Let ξ_t be the t-th generation population. Define a distance based on the fitness function $d(x) = f_{\max} - f(x)$.
- If the following drift conditions hold:
  - for any $d(\xi_t) > 0$:
    $$E[d(\xi_t) - d(\xi_{t+1}) \mid \xi_t] > c_t,$$
    with $c_t \in \{0, 1\}$ and
  - $\sum_{t=0}^{\infty} c_t = \infty$,

  then the stochastic process converges: $\lim_{t\to\infty} d(\xi_t) \overset{L_1}{=} 0$.

[a] G. Rudolph, "Convergence of Non-elitist Strategies," in *Proc. of the First IEEE conference on Evolutionary Computation*, 63-66, 1994.

Computation time: General Result

Given a combinatorial optimisation problem and an EA (He and Yao 2001) [a]

- Define the **first hitting time** to the optimal solution as

$$\tau := \min\{t \geq 0; d(\xi_t) = 0 \mid \xi_0\}.$$

- If the following drift condition holds:
  - for ξ_t is not an optimal solution,

$$b_u \geq E[d(\xi_t) - d(\xi_{t+1})] \geq b_l.$$

  where b_u and b_l are positives.

  then $d(\xi_0)/b_u \leq E[\tau \mid \xi_0] \leq d(\xi_0)/b_l$.

[a] J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 127 (1):57-85, 2001. (Erratum in Artifi cial Intelligence, 140(1):245-248, 2002))

Computation Time: An Example

An application of the general result to the linear function (first considered by Droste et al. 1998): [a]
$f(s) = \sum_{i=1}^{n} w_i s_i$, where $w_1, w_2, \cdots, w_n$ are positives. s_i are bits in binary strings of length n.

A $(1+1)$ EA:

Mutation flip each bit in th binary string with probability $1/n$.

Selection If $f(\xi_t^{(m)}) > f(\xi_t)$, then let $\xi_{t+1} = \xi_t^{(m)}$, otherwise, let $\xi_{t+1} = \xi_t$.

---

[a]S. Droste, T. Jansen and I. Wegener. A Rigorous Complexity Analysis of the (1+1) Evolutionary Algorithm for Linear Functions with Boolean Inputs. *Evolutionary Computation* 6(2): 185–196, 1998.

Computation Time: Results from the Example

- Assume n is an even integer (He and Yao 2004).[a] Define

$$d(x) = n \ln(1 + \sum_{i=1}^{n/2} c \mid s_i - 1 \mid + \sum_{i=n/2+1}^{n} \mid s_i - 1 \mid),$$

where $c (1 < c \leq 2)$ is a constant.

- The mean drift: $E[d(\xi_t) - d(\xi_{t+1})] \geq \frac{c'}{8(1+c)}$ where c' is a positive constant.
- The first hitting time:

$$E[\tau \mid \xi_0] \leq \frac{\max\{d(x)\}}{\min\{drift\}} = O(n \ln n).$$

[a] J. He and X. Yao. A Study of Drift Analysis for Estimating Computation Time of Evolutionary Algorithms. *Natural Computing*, 3(1):21-35, 2004.

Wait a Minute!

One of the key characteristics of EAs is their populations. What about population-based EAs?

- Computational complexity: will the time complexity be changed if we introduce a population into EAs?
- First hitting probability: which one between an $(N+N)$ EA and a $(1+1)$ has a lower failure probability to hit the optimal solution.
- First hitting time to the optimal solution: whether the mean first hitting time of an $(N+N)$ EA would be shorter than that of a $(1+1)$ EA that uses the same mutation operator and, if it is, how much shorter?

In our work: same mutation in the $(1+1)$ EA and the $(N+N)$

EA; but different selection.

Population Can Help

A case study on comparing time complexity: population can be beneficial (He and Yao, 2002)) [a]

Given a problem:

$$f(x) = \begin{cases} n+1-\frac{1}{n}\left(\sum_{i=1}^{n} s_i\right), & \text{if } \sum_{i=1}^{n} s_i \text{ is even,} \\ n+1-\frac{1}{n}\left(\sum_{i=1}^{n} s_i\right)-2, & \text{if } \sum_{i=1}^{n} s_i \text{ is odd,} \end{cases}$$

- A $(1+1)$ EA: $E[\tau] = \Omega(qp^{-2}(1+\frac{p}{q})^n)$.
- An $(N+N)$ EA: $E[\tau] = O(n \log n)$.

[a] J. He and X. Yao. From an Individual to a Population: An Analysis of the First Hitting Time of Population-Based Evolutionary Algorithms, IEEE Trans. Evo. Comp. 6(5):495-511, 2002.

But Not Always

A case study on comparing time complexity: population may not be beneficial (He and Yao, 2002))[a]
Given a problem

$$f(x) = \begin{cases} n+1-\frac{1}{n}\left(\sum_{i=1}^{n} s_i\right)-(n-1), & \text{if } \sum_{i=1}^{n} s_i = n-1, \\ n+1-\frac{1}{n}\left(\sum_{i=1}^{n} s_i\right), & \text{otherwise.} \end{cases}$$

- A $(1+1)$ EA: $E[\tau] = O(n^2)$.
- An $(N+N)$ EA: $E[\tau] = \Omega\left((n(1-(1-q)^N))^{-1}\right)$.

[a] J. He and X. Yao. From an Individual to a Population: An Analysis of the First Hitting Time of Population-Based Evolutionary Algorithms, IEEE Trans. Evo. Comp. 6(5):495-511, 2002.

Characterising Problems for EAs

OK, OK, you’ve made your points. Drift analysis can be useful to analyse EAs, some of which may be more efficient than others, but

- How do I know whether I should use a given EA for a particular problem?
- In other words, how to characterise problems for EAs?

EA-Hardness

Given an EA, optimization problems can be divided into two classes based on the mean number of generations (first hitting time, to be more accurate) needed to solve the problems.

- **EA-Easy Class**: For the given EA, starting from any initial population ξ_0, the mean number of generations needed by the EA to solve the problem, i.e., $E[\tau \mid \xi_0]$, is at most polynomial in the problem size.
- **EA-Hard Class:** For the given EA, starting from some initial population ξ_0, the mean number of generations needed by the EA to solve the problem, i.e., $E[\tau \mid \xi_0]$, is at least exponential in the problem size.

EA-Easy Problems

More details are in (He and Yao, 2004) [a]

Given an EA, a problem is EA-easy **if and only if** there exists a distance function $d(x)$ such that

- $d(x) \leq g(n)$, where $g(n)$ is polynomial in the problem size n, and
- the one-step mean drift satisfies: for any population ξ_t which is not an optimal solution,

$$E[d(\xi_t) - d(\xi_{t+1})] \geq c,$$

  where $c > 0$ is a constant.

[a] J. He and X. Yao. A Study of Drift Analysis for Estimating Computation Time of Evolutionary Algorithms. *Natural Computing*, 3(1), pp.21-35, 2004.

EA-Hard Problems

More details are in (He and Yao, 2004) [a]

Given an EA, a problem is GA-hard **if and only if** there exists a distance function $d(x)$ such that

- for some population $x : d(x) \geq g(n)$, where $g(n)$ is exponential in the problem size n, and
- the one-step mean drift satisfies: for any population ξ_t which is not an optimal solution,

$$E[d(\xi_t) - d(\xi_{t+1}) \mid \xi_t] \leq c,$$

where c is a positive constant.

[a] J. He and X. Yao. A Study of Drift Analysis for Estimating Computation Time of Evolutionary Algorithms. *Natural Computing*, 3(1), pp.21-35, 2004.

Discussions on Analytical Tools

Advantages of Drift Analysis:

- Intuitive and simple idea: time= distance/drift;
- Unified framework: it can be used to analyse the computation time, convergence, etc., of EAs;
- Mathematical foundation: it can be thought as an application of martingale theory. We can get the help from real theoreticans.

Disadvantages of Drift Analysis:

- It is nontrivial to define a distance function;
- It is hard to estimate the mean drift.

Conclusions on Results

- Drift analysis can be used to analyse the convergence and computation time of EAs.
- Population can be beneficial to EAs, but not always.
- Necessary and sufficient conditions for EA-hardness can be established using drift analysis.
- Where the EA-hardness is: “wide-gap” and “long-path” problems (not covered in this talk).

Downloadable papers:

www.cs.bham.ac.uk/∼xin/journal_papers.html