Random Gradient Hyper-heuristics Can Learn to Escape Local Optima in Multimodal Optimisation

Yuxuan Ma[1] Pietro S. Oliveto[1] John Alasdair Warwicker[2]

[1]Southern University of Science and Technology, Shenzhen, Guangdong, China
[2]Karlsruhe Institute of Technology, Germany

July 16, 2025

Motivation and Highlight

- Generalised Random Gradient Selection Hyper-heuristics (GRG SHH) have been shown to have remarkable performance on unimodal functions(e.g. LeadingOnes,OneMax,Ridge) [Lissovoi et al., 2017, Doerr et al., 2018, Lissovoi et al., 2020].

Motivation and Highlight

- Generalised Random Gradient Selection Hyper-heuristics (GRG SHH) have been shown to have remarkable performance on unimodal functions(e.g. LeadingOnes,OneMax,Ridge) [Lissovoi et al., 2017, Doerr et al., 2018, Lissovoi et al., 2020].
- Our aim is to analyse for the first time the performance of GRG in multimodal optimisation.

Motivation and Highlight

- Generalised Random Gradient Selection Hyper-heuristics (GRG SHH) have been shown to have remarkable performance on unimodal functions(e.g. LeadingOnes,OneMax,Ridge) [Lissovoi et al., 2017, Doerr et al., 2018, Lissovoi et al., 2020].
- Our aim is to analyse for the first time the performance of GRG in multimodal optimisation.
- We consider the TwoRates function introduced by [Krejca and Witt, 2024] to analyse the performance of the Flex-EA that saves the promising mutation operators in an archive.

Motivation and Highlight

- Generalised Random Gradient Selection Hyper-heuristics (GRG SHH) have been shown to have remarkable performance on unimodal functions(e.g. LeadingOnes,OneMax,Ridge) [Lissovoi et al., 2017, Doerr et al., 2018, Lissovoi et al., 2020].
- Our aim is to analyse for the first time the performance of GRG in multimodal optimisation.
- We consider the TwoRates function introduced by [Krejca and Witt, 2024] to analyse the performance of the Flex-EA that saves the promising mutation operators in an archive.
- The expected runtime of flex-EA with the recommended parameters setting on TwoRates is bounded by $\mathcal{O}(n^{4.5})$ [Krejca and Witt, 2024].

Motivation and Highlight

- Generalised Random Gradient Selection Hyper-heuristics (GRG SHH) have been shown to have remarkable performance on unimodal functions(e.g. LeadingOnes,OneMax,Ridge) [Lissovoi et al., 2017, Doerr et al., 2018, Lissovoi et al., 2020].
- Our aim is to analyse for the first time the performance of GRG in multimodal optimisation.
- We consider the TwoRates function introduced by [Krejca and Witt, 2024] to analyse the performance of the Flex-EA that saves the promising mutation operators in an archive.
- The expected runtime of flex-EA with the recommended parameters setting on TwoRates is bounded by $\mathcal{O}(n^{4.5})$ [Krejca and Witt, 2024].
- We show that GRG is considerably faster by learning the best operator without switching.

Motivation and Highlight

- Generalised Random Gradient Selection Hyper-heuristics (GRG SHH) have been shown to have remarkable performance on unimodal functions(e.g. LeadingOnes,OneMax,Ridge) [Lissovoi et al., 2017, Doerr et al., 2018, Lissovoi et al., 2020].
- Our aim is to analyse for the first time the performance of GRG in multimodal optimisation.
- We consider the TwoRates function introduced by [Krejca and Witt, 2024] to analyse the performance of the Flex-EA that saves the promising mutation operators in an archive.
- The expected runtime of flex-EA with the recommended parameters setting on TwoRates is bounded by $\mathcal{O}(n^{4.5})$ [Krejca and Witt, 2024].
- We show that GRG is considerably faster by learning the best operator without switching.
- This is the first time that GRG is analysed with all RLS operators $H = \{\mathrm{RLS}_1, \ldots, \mathrm{RLS}_n\}$.

Generalised Random Gradient Hyper-heuristic

Algorithm 1 Generalised Random Gradient Hyper-heuristic(GRG)

```
Choose x ∈ S uniformly at random
while stopping conditions not satisfied do
    Choose h ∈ H uniformly at random
    c_t ← 0
    while c_t < τ do
        c_t ← c_t + 1; x′ ← h(x)
        if f(x′) > f(x) then
            c_t ← 0; x ← x′
```

TwoRates and CutTwoRates fitness functions

$$\textsc{TwoRates}(x) := \begin{cases} \mathrm{OM}(x) & \text{if } |x|_1 < s \text{ or } |x|_1 \geq 3n/4 \\ & \quad \text{or } |x|_1 = s + ig \text{ for } i \in \{0, 2, 3, 5, 6, \ldots\}; \\ -1 & \text{otherwise.} \end{cases}$$

$$\textsc{CutTwoRates}(x) := \begin{cases} \textsc{TwoRates}(x) & \text{if } |x|_1 \leq 3n/4; \\ -1 & \text{otherwise.} \end{cases}$$

- $s := 3n/4 - \sqrt{n} \implies$ the length of the $\textsc{Hurdle}$ section is $\sqrt{n}$
- alternating gaps of size $2g := 2\log_2 n$ and $g := \log_2 n$

Note: g and $\sqrt{n}/g$ are defined to be integers $\implies$ g has to be even.

TwoRates and CutTwoRates fitness functions

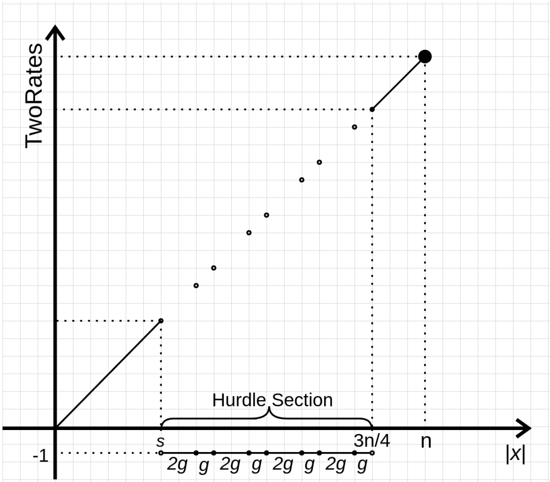


Figure: TWORATES fitness function, where s denotes $3n/4 - \sqrt{n}$ and g denotes $\log_2 n$. Note that the graph magnifies the size of the hurdle region

The Best Operator for the Hurdles is RLS_{4g} where $g = \log_2 n$

Theorem

Let $\rho > \gamma > 0$ be some constants, and let $\beta = \rho/\gamma$. Then the probability of using $RLS_{\rho g}$ and getting an improvement of γg in the hurdles region of the TwoRates *function is equal to*

$$p(n, a, \gamma g, (\rho - \gamma)g/2) = (1 - o(1))\sqrt{2\rho/(\pi(\rho^2 - \gamma^2)\log n)}\, n^{-\gamma f(\beta)},$$

where $f(\beta) = \beta[1 + \log\sqrt{(\beta^2 - 1)/(3\beta^2)}] + \log\sqrt{3(\beta + 1)/(\beta - 1)}$.

Proof idea:

- $p(n, a, \gamma g, (\rho - \gamma)g/2) = \binom{n-a}{(\rho+\gamma)g/2}\binom{a}{(\rho-\gamma)g/2}/\binom{n}{\rho g}$
- $$\frac{(s-t)^t}{t^{t+1/2}}\frac{e^{t-\frac{1}{12t}}}{\sqrt{2\pi}} < \binom{s}{t} < \frac{s^t}{t^{t+1/2}}\frac{e^{t+\frac{1}{360t^3}-\frac{1}{12t}}}{\sqrt{2\pi}}.$$

The Best Operator for the Hurdles is RLS_{4g} where $g = \log_2 n$

Corollary

The value of m, $1 \leq m \leq n$, that minimises the expected runtime of RLS_m for crossing the $\textsc{Hurdle}$ *region of* $\textsc{TwoRates}$ *starting from $|x|_1 = 3n/4 - \sqrt{n}$, converges to $m = 4g$ (i.e., $m = 4\log n$) when n is sufficiently large.*

Proof idea:

- By Hoeffding's bound, the probability for RLS_m, where $m = \omega(\log n)$, to make any improvement is $1/n^{\omega(1)}$

The Best Operator for the Hurdles is RLS_{4g} where $g = \log_2 n$

Corollary

The value of m, $1 \leq m \leq n$, that minimises the expected runtime of RLS_m for crossing the $\textsc{Hurdle}$ *region of* $\textsc{TwoRates}$ *starting from $|x|_1 = 3n/4 - \sqrt{n}$, converges to $m = 4g$ (i.e., $m = 4\log n$) when n is sufficiently large.*

Proof idea:

- By Hoeffding's bound, the probability for RLS_m, where $m = \omega(\log n)$, to make any improvement is $1/n^{\omega(1)}$
- When n is sufficiently large, $n^{-\gamma f(\beta)}$ dominates $(1 - o(1))\sqrt{2\rho/(\pi(\rho^2 - \gamma^2)\log n)}n^{-\gamma f(\beta)}$,

The Best Operator for the Hurdles is RLS_{4g} where $g = \log_2 n$

Corollary

The value of m, $1 \leq m \leq n$, that minimises the expected runtime of RLS_m for crossing the $\textsc{Hurdle}$ *region of* $\textsc{TwoRates}$ *starting from $|x|_1 = 3n/4 - \sqrt{n}$, converges to $m = 4g$ (i.e., $m = 4\log n$) when n is sufficiently large.*

Proof idea:

- By Hoeffding's bound, the probability for RLS_m, where $m = \omega(\log n)$, to make any improvement is $1/n^{\omega(1)}$
- When n is sufficiently large, $n^{-\gamma f(\beta)}$ dominates $(1 - o(1))\sqrt{2\rho/(\pi(\rho^2 - \gamma^2)\log n)}\, n^{-\gamma f(\beta)}$,
- for $\beta > 1$, $f(\beta)$ reaches its minimum at $\beta = \rho/\gamma = 2 \implies$ the best operator crossing $2g$ hurdle converges to RLS_{4g}, and the best operator crossing $2g + g$ hurdle converges to RLS_{6g}, etc.

The Best Operator for the Hurdles is RLS_{4g} where $g = \log_2 n$

Corollary

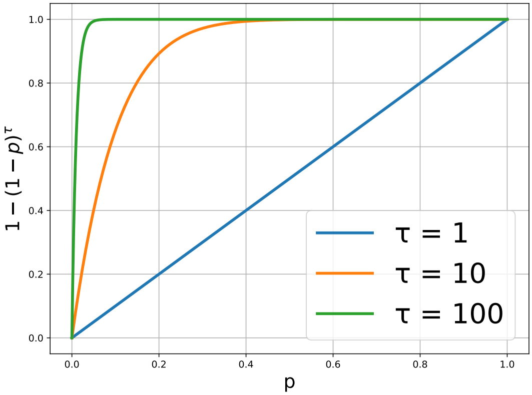
The value of m, $1 \le m \le n$, that minimises the expected runtime of RLS_m for crossing the $\textsc{Hurdle}$ *region of* $\textsc{TwoRates}$ *starting from $|x|_1 = 3n/4 - \sqrt{n}$, converges to $m = 4g$ (i.e., $m = 4\log n$) when n is sufficiently large.*

Proof idea:

- By Hoeffding's bound, the probability for RLS_m, where $m = \omega(\log n)$, to make any improvement is $1/n^{\omega(1)}$
- When n is sufficiently large, $n^{-\gamma f(\beta)}$ dominates $(1 - o(1))\sqrt{2\rho/(\pi(\rho^2 - \gamma^2)\log n)}n^{-\gamma f(\beta)}$,
- for $\beta > 1$, $f(\beta)$ reaches its minimum at $\beta = \rho/\gamma = 2 \implies$ the best operator crossing $2g$ hurdle converges to RLS_{4g}, and the best operator crossing $2g + g$ hurdle converges to RLS_{6g}, etc.
- The probability for RLS_{6g} to make $3g$ improvement is already $1/n$ smaller than the probability for RLS_{4g} to make $2g$ improvement, while crossing the g hurdle is easier.

Why should we set τ as $\tau \geq n^{\log 9+\epsilon}$ and $\tau = poly(n)$

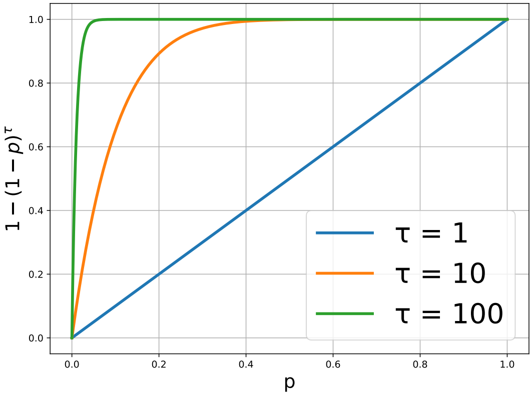
- We want to set τ large s.t. RLS_{4g} can make consecutive improvement in the $\textsc{Hurdle}$ section with high probability.



Why should we set τ as $\tau \geq n^{\log 9+\epsilon}$ and $\tau = poly(n)$

- We want to set τ large s.t. RLS_{4g} can make consecutive improvement in the HURDLE section with high probability.
- We don't want to set τ too large since the expected waiting time will be long. And, even more operators will succeed, which means that the time for each improvement will be longer.

$$\frac{1}{p} - \frac{(1-p)^{\tau}\tau}{1-(1-p)^{\tau}}$$

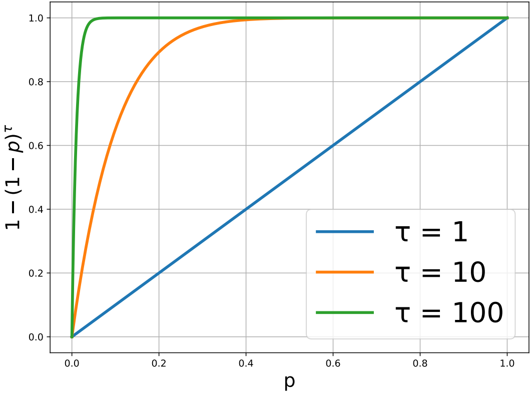


Why should we set τ as $\tau \geq n^{\log 9+\epsilon}$ and $\tau = poly(n)$

- We want to set τ large s.t. RLS_{4g} can make consecutive improvement in the $\mathrm{H{\small URDLE}}$ section with high probability.
- We don't want to set τ too large since the expected waiting time will be long. And, even more operators will succeed, which means that the time for each improvement will be longer.

$$\frac{1}{p} - \frac{(1-p)^\tau \tau}{1-(1-p)^\tau}$$

- The probability for RLS_{4g} to make an improvement of $2g$ bits is $(1-o(1))\sqrt{2/(3\pi \log n)}n^{-\log 9} \implies$ we can set $\tau \geq n^{\log 9+\epsilon}$ while $\tau = poly(n)$.



GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_{4g}\}$ CutTwoRates

Note: To see the best possible speed up compared to Flex-EA, we analyse GRG with two optimal operators on $\textsc{CutTwoRates}$.

Theorem

Let ϵ be an arbitrarily small positive constant. The expected runtime of the Generalised Random Gradient hyper-heuristic for $\textsc{CutTwoRates}$ with $H = \{RLS_1, RLS_{4g}\}$ and $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$ is $\mathcal{O}(\tau + n^{\log 9}\sqrt{n/\log n})$.

Proof idea:

- For the first $\textsc{OneMax}$ part, both RLS_1 and RLS_{4g} have overwhelming probability to succeed consecutively until reaching the start point of the $\textsc{Hurdle}$ section, and the whole expected runtime in this stage can be bounded by $\mathcal{O}(n^{5-\log 9}\sqrt{\log n})$.

GRG with $H = \{\textbf{RLS}_1, \textbf{RLS}_{4g}\}$ CutTwoRates

Note: To see the best possible speed up compared to Flex-EA, we analyse GRG with two optimal operators on CutTwoRates.

Theorem

Let ϵ be an arbitrarily small positive constant. The expected runtime of the Generalised Random Gradient hyper-heuristic for CutTwoRates *with $H = \{RLS_1, RLS_{4g}\}$ and $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$ is $\mathcal{O}(\tau + n^{\log 9}\sqrt{n/\log n})$.*

Proof idea:

- For the first OneMax part, both RLS_1 and RLS_{4g} have overwhelming probability to succeed consecutively until reaching the start point of the Hurdle section, and the whole expected runtime in this stage can be bounded by $\mathcal{O}(n^{5-\log 9}\sqrt{\log n})$.
- For the Hurdle section, RLS_1 will directly fail after τ iterations once selected, while RLS_{4g} has overwhelming probability to cross the whole Hurdle section. The expected runtime in this stage can be bounded by $\mathcal{O}(\tau + n^{\log 9}\sqrt{n/\log n})$.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \ldots, \mathbf{RLS}_n\}$ TwoRates

Note: In practice, we don't know which operators are useful so we consider them all for the complete function.

Theorem

The expected runtime of the Generalised Random Gradient Hyper-heuristic with $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$, *equipped with* $H = \{RLS_1 \ldots, RLS_n\}$ *for* $\textsc{TwoRates}$ *is* $\mathcal{O}(n\tau \log n)$.

Proof idea:

- Consider the expected runtime on the first $\textsc{OneMax}$ part, the $\textsc{Hurdle}$ section and the second $\textsc{OneMax}$ part, respectively.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \ldots, \mathbf{RLS}_n\}$ TwoRates

Note: In practice, we don't know which operators are useful so we consider them all for the complete function.

Theorem

The expected runtime of the Generalised Random Gradient Hyper-heuristic with $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$, equipped with $H = \{RLS_1 \ldots, RLS_n\}$ for $\textsc{TwoRates}$ is $\mathcal{O}(n\tau \log n)$.

Proof idea:

- Consider the expected runtime on the first $\textsc{OneMax}$ part, the $\textsc{Hurdle}$ section and the second $\textsc{OneMax}$ part, respectively.
- The expected runtime on the second $\textsc{OneMax}$ part is $\mathcal{O}(n\tau \log n) \implies$ the expected runtime on the first $\textsc{OneMax}$ part is also $\mathcal{O}(n\tau \log n)$.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \ldots, \mathbf{RLS}_n\}$ TwoRates

Note: In practice, we don't know which operators are useful so we consider them all for the complete function.

Theorem

The expected runtime of the Generalised Random Gradient Hyper-heuristic with $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$, equipped with $H = \{RLS_1 \ldots, RLS_n\}$ for $\textsc{TwoRates}$ is $\mathcal{O}(n\tau \log n)$.

Proof idea:

- Consider the expected runtime on the first $\textsc{OneMax}$ part, the $\textsc{Hurdle}$ section and the second $\textsc{OneMax}$ part, respectively.
- The expected runtime on the second $\textsc{OneMax}$ part is $\mathcal{O}(n\tau \log n) \implies$ the expected runtime on the first $\textsc{OneMax}$ part is also $\mathcal{O}(n\tau \log n)$.
- The expected runtime on the $\textsc{Hurdle}$ section is $\mathcal{O}(n\tau)$.

GRG with $H = \{\textbf{RLS}_1, \textbf{RLS}_2, \ldots, \textbf{RLS}_n\}$ Second OneMax Part

Lemma

The expected runtime of the Generalised Random Gradient Hyper-heuristic equipped with $H := \{\mathrm{RLS}_1, \ldots, \mathrm{RLS}_n\}$ and $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$, starting from a random heuristic choice at $|x|_1 \geq 3n/4$ on TwoRates *is $\mathcal{O}(n\tau \log n)$.*

Proof idea:

- We pessimistically assume that once an operator makes any improvement, the fitness improves only by 1 and the number of iterations required to make the improvement is exactly τ.

GRG with $H = \{\textbf{RLS}_1, \textbf{RLS}_2, \ldots, \textbf{RLS}_n\}$ Second OneMax Part

Lemma

The expected runtime of the Generalised Random Gradient Hyper-heuristic equipped with $H := \{\mathrm{RLS}_1, \ldots, \mathrm{RLS}_n\}$ and $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$, starting from a random heuristic choice at $|x|_1 \geq 3n/4$ on TwoRates *is $\mathcal{O}(n\tau \log n)$.*

Proof idea:

- We pessimistically assume that once an operator makes any improvement, the fitness improves only by 1 and the number of iterations required to make the improvement is exactly τ.
- Divide H into three sets H_1, H_2, H_3 where,
  - H_1 contains only RLS_1;
  - H_3 contains the operators that have a probability to find an improvement in the second OneMax part that is always smaller that $1/n^2$ (i.e. if p is the probability that the current operator will make an improvement in a single step, then $1-(1-p)^\tau < 1/n^2$);
  - H_2 contains the remaining operators.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \ldots, \mathbf{RLS}_n\}$ Second OneMax Part

- Let $T \mid \mathrm{OM} = k$ be the runtime before reaching the optimum, starting at k one bits.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \dots, \mathbf{RLS}_n\}$ Second OneMax Part

- Let $T \mid \mathrm{OM} = k$ be the runtime before reaching the optimum, starting at k one bits.
- Let $k^* = \arg\max_{k \in [3n/4..n]} \mathbb{E}[T \mid \mathrm{OM} = k]$.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \ldots, \mathbf{RLS}_n\}$ Second OneMax Part

- Let $T \mid \mathrm{OM} = k$ be the runtime before reaching the optimum, starting at k one bits.
- Let $k^* = \arg\max_{k \in [3n/4..n]} \mathbb{E}[T \mid \mathrm{OM} = k]$.
- We have,

$$
\begin{aligned}
\mathbb{E}[T \mid \mathrm{OM} = k^*] = \frac{1}{n} \cdot \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_1] \\
+ \frac{1}{n} \cdot \sum_{i \in H_2} \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_i] \\
+ \frac{1}{n} \cdot \sum_{i \in H_3} \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_i].
\end{aligned}
$$

- For RLS_1, it has overwhelming probability to consecutively succeed until reaching the optimum $\Longrightarrow\ \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_1] = \mathcal{O}(n\tau)$

- For RLS_1, it has overwhelming probability to consecutively succeed until reaching the optimum $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_1] = \mathcal{O}(n\tau)$
- For RLS_i where $i \in H_2$, either,
  (a) It successfully goes all the way to the end $\implies$ the time can be bounded by $\mathcal{O}(n\tau)$,
  (b) It fails before reaching the optimum $\implies$ the time can be bounded by $\tau\theta + \tau + \mathbb{E}[T \mid \mathrm{OM} = k^* + \theta] < \mathcal{O}(n\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$.

  $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_{i \in H_2}] < \mathcal{O}(n\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$

- For RLS_1, it has overwhelming probability to consecutively succeed until reaching the optimum $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_1] = \mathcal{O}(n\tau)$
- For RLS_i where $i \in H_2$, either,
  (a) It successfully goes all the way to the end $\implies$ the time can be bounded by $\mathcal{O}(n\tau)$,
  (b) It fails before reaching the optimum $\implies$ the time can be bounded by $\tau\theta + \tau + \mathbb{E}[T \mid \mathrm{OM} = k^* + \theta] < \mathcal{O}(n\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$.

  $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_{i \in H_2}] < \mathcal{O}(n\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$
- For RLS_i where $i \in H_3$, either,
  (a) It fails to make a single improvement $\implies$ the runtime is $\tau + \mathbb{E}[T \mid \mathrm{OM} = k^*]$
  (b) It succeeds in making a single improvement $\implies$ the runtime together with the probability can be bounded by $\mathcal{O}(n^2\tau) \cdot 1/n^2$ by definition of H_3.

  $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_{i \in H_3}] < \mathcal{O}(\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$

- For RLS_1, it has overwhelming probability to consecutively succeed until reaching the optimum $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_1] = \mathcal{O}(n\tau)$
- For RLS_i where $i \in H_2$, either,
  (a) It successfully goes all the way to the end $\implies$ the time can be bounded by $\mathcal{O}(n\tau)$,
  (b) It fails before reaching the optimum $\implies$ the time can be bounded by $\tau\theta + \tau + \mathbb{E}[T \mid \mathrm{OM} = k^* + \theta] < \mathcal{O}(n\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$.

  $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_{i \in H_2}] < \mathcal{O}(n\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$
- For RLS_i where $i \in H_3$, either,
  (a) It fails to make a single improvement $\implies$ the runtime is $\tau + \mathbb{E}[T \mid \mathrm{OM} = k^*]$
  (b) It succeeds in making a single improvement $\implies$ the runtime together with the probability can be bounded by $\mathcal{O}(n^2\tau) \cdot 1/n^2$ by definition of H_3.

  $\implies \mathbb{E}[T \mid \mathrm{OM} = k^*, \mathrm{RLS}_{i \in H_3}] < \mathcal{O}(\tau) + \mathbb{E}[T \mid \mathrm{OM} = k^*]$
- By Hoeffding's bound, we know $|H_3| = n - \mathcal{O}(\log n)$. Using all the inequalities above, we finish the proof.

GRG with $H = \{\mathbf{RLS}_1, \mathbf{RLS}_2, \ldots, \mathbf{RLS}_n\}$ Hurdle Section

Lemma

Starting with a random heuristic choice at the beginning of any hurdle in the $\textsc{Hurdle}$ *region, the expected runtime of the Generalised Random Gradient Hyper-heuristic with* $\tau \geq n^{\log 9+\epsilon}$, $\tau = poly(n)$, *equipped with* $H = \{RLS_1 \ldots RLS_n\}$, *to optimise* $\textsc{CutTwoRates}$ *is* $\mathcal{O}(n\tau)$.

Proof idea: The idea is similar.

Conclusion and Discussion

- The whole analysis relies on the gaps of TwoRates being of even length.
- The following variant of GRG allows to jump gaps of both even and odd length at the expense of a factor 2 on the expected runtime.

Algorithm 2 Extended GRG Hyper-heuristic

```
Choose x ∈ S uniformly at random
while stopping conditions not satisfied do
    Choose h ∈ H according to some distribution
    c_t ← 0
    while c_t < τ do
        c_t ← c_t + 1; x′ ← h(x); x† ← RLS_1(x′)
        if f(x′) > f(x) then
            c_t ← 0; x ← x′
        else if f(x†) > f(x) then
            c_t ← 0; x ← x†
```

Conclusion and Discussion

Conclusion

- Flex-EA on $\textsc{TwoRates}$: $\mathcal{O}(n^{4.5})$
- GRG with $H = \{\mathrm{RLS}_1, \mathrm{RLS}_{4g}\}$ on $\textsc{CutTwoRates}$: $\mathcal{O}\left(n^{3.67}/\sqrt{\log n}\right)$
- GRG with $H = \{\mathrm{RLS}_1, \ldots, \mathrm{RLS}_n\}$ on $\textsc{TwoRates}$: $\mathcal{O}(n^{4.17} \log n)$

Future work

- Flex-EA may also be able to optimise $\textsc{TwoRates}$ by only applying RLS_{4g}.
- Analysis of ARG on $\textsc{TwoRates}$ is needed because in practice we do not know how to set τ.
- Lower bounds and more precise upper bounds are needed.

References I

Doerr, B., Lissovoi, A., Oliveto, P. S., and Warwicker, J. A. (2018).
On the runtime analysis of selection hyper-heuristics with adaptive learning periods.
In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1015–1022. ACM.

Krejca, M. S. and Witt, C. (2024).
A flexible evolutionary algorithm with dynamic mutation rate archive.
In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '24, pages 1578–1586. ACM.

Lissovoi, A., Oliveto, P., and Warwicker, J. A. (2020).
How the duration of the learning period affects the performance of random gradient selection hyper-heuristics.
Proceedings of the AAAI Conference on Artificial Intelligence, 34(03):2376–2383.

References II

Lissovoi, A., Oliveto, P. S., and Warwicker, J. A. (2017).
On the runtime analysis of generalised selection hyper-heuristics for pseudo-boolean optimisation.
In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 849–856. ACM.