Supplemental material for Bureerat and Sleesongsom, “Constraint Handling Technique for Four-Bar Linkage Path Generation Using Self-Adaptive Teaching-Learning Based Optimization with a Diversity Archive”, Engineering Optimization, 2020.

Appendix

A. Algorithm 1. Function evaluation (with/without prescribed timing)

| Input | x = \{ r_1, r_2, r_3, r_4, r_{px}, r_{py}, \theta_0, x_{O2}, y_{O2}, \theta_i^2 \} |
|-------|------------------------------------------------------------------------|
| Output| f, x and constraints

Evaluate constraints

Step 1 If the optimization problem is the without-prescribe-timing problem and if \( \theta_i^2 \) cannot fulfil constraint (3.2), activate Algorithm 2 to reassign the angle values.

Step 2 If constraints (3.3)-(3.4) are infeasible, activate the technique to reassign link lengths.

Position analysis and function evaluation

Step 1 Otherwise, solving Equations (2-3) for all values of \( \theta_2 \) and solving Equation (1) for \( r_p \) at each \( \theta_2 \).

Step 2 Compute the objective function values and constraints according to Equation (3.1)-(3.5).

B. Algorithm 2: A reassigning technique for timing constraint

| Input | infeasible x at elements \( \theta_i^2, \theta_i^{i+1}, ..., \theta_i^N \) |
|-------|------------------------------------------------------------------|
| Output| feasible x at elements \( \theta_i^2, \theta_i^{i+1}, ..., \theta_i^N \) |

Generate a set of uniform random numbers \( \{ \alpha_i, ..., \alpha_N \} \), \( \alpha_i \in (0, 1) \).

If \( \alpha_2 + ... + \alpha_N \geq 2\pi \), scale them down and modify their values as:
For $i = 2$ to $N$

Generate $\alpha_i = 1.99\pi\alpha_i/(\alpha_2 + \ldots + \alpha_N)$

End

For $i = 1$ to $N$

Step 1 If $i = 1$, $\theta_{i2} = \alpha_1$.

Step 2 Otherwise, $\theta_{i2} = \theta_{i-1}^2 + \alpha_i$.

End

C. **Algorithm 3**: Repairing the Grashof's criterion constraint.

```
Input infeasible \{r_1, r_2, r_3, r_4\}

Output feasible \{r_1, r_2, r_3, r_4\}

Step 1 Generate a set of uniform random numbers \{\delta_1, \delta_2, \delta_3\}, \delta_i \in (0,1).

Step 2 If $2\delta_3 \geq \delta_1$

Step 3 Assign values

$S_1 = \delta_1$

$S_2 = 2\delta_3$

$S_3 = 2\delta_1 + \delta_2$

$S_4 = 2\delta_3 + \delta_2 + \delta_1$

Step 3 If $\max(S_1, S_2, S_3, S_4) > 1$, compute $S_i = S_i/2$ and compute step (2) until $\max(S_1, S_2, S_3, S_4) < 1$

Step 4 Compute $r_i = r_{min} + (r_{max} - r_{min})S_i$ for $i = 1, \ldots, 4$.
```
D. Algorithm 4 Procedure of ATLBO-DA

**Input:** maximum number of generations \((n_{it})\), population size \((n_P)\)

**Output:** \(x^{\text{best}}, f^{\text{best}}\)

**Initialization:**

Step 0.1 Generate \(n_P\) initial students \(\{x^i\}\) and perform function evaluations \(\{f^i\}\).

Step 0.2 Initiate four \(1\times2\) matrices, \(TRR\_Success, TRR\_Fail, LRR\_Success\) and \(LRR\_Fail\), whose all elements are set to be ones.

**Main procedure**

Step 1 While (the termination conditions are not met) do

\{Teacher Phase\}

Step 2 Calculate the mean position of solutions \(\{x^i\}\) written as \(M_{avg}\).

Step 3 Calculate the probabilities of selecting the intervals for \(T_{RR}\) using (18).

Step 4 For \(i=1\) to \(n_P\)

Step 4.1 Perform roulette wheel selection with \(P_{TRR_j}\).

\hspace{1cm} Step 4.1.1 If \(j = 1\) is selected, \(T_{RR} = 0.4 + 0.1\ rand\) is sampled.

\hspace{1cm} Step 4.1.2 Else, if \(j = 2\) is selected, \(T_{RR} = 0.5 + 0.1\ rand\) is sampled.

Step 4.2 Generate \(P_r = \ rand\) and select a teacher.

\hspace{1cm} Step 4.2.1 If \(P_r \leq T_{RR}\), set the best solution as a teacher \(M_{\text{best}}\).

\hspace{1cm} Step 4.2.2 Else, if \(P_r > T_{RR}\), randomly select a solution in \(A_D\) and set it as a teacher \(M_{\text{best}}\).

Step 4.3 Create \(x^{i\_\text{new}}\) using \((10-12)\) and perform function evaluation.

\hspace{1cm} Step 4.3.1 If \(x^{i\_\text{new}}\) is better than \(x^i\), add 1 point to the \(j\)-th element of \(TRR\_Success\).

\hspace{1cm} Step 4.3.2 Else, add 1 point to the \(j\)-th element of \(TRR\_Fail\).
Step 5 Replace \( \{x^i\} \) by \( n_P \) best solutions from \( \{x^i\} \cup \{x^i_{\text{new}}\} \).

\( \{\text{Learning Phase}\} \)

Step 6 Calculate the probabilities of selecting the intervals for \( L_{RR} \) similar to that for \( T_{RR} \) in step 3. 
\[
P_{TRR} = \frac{L_{RR} \_ \text{Success}_j}{L_{RR} \_ \text{Success}_j + L_{RR} \_ \text{Fail}_j}
\]

Step 7 For \( i=1 \) to \( n_P \)

Step 7.1 Perform roulette wheel selection with \( PLRR_j \).

Step 7.1.1 If \( j = 1 \) is selected, \( L_{RR} = 0.4 + 0.1 \cdot \text{rand} \) is sampled.

Step 7.1.2 Else if \( j = 2 \) is selected, \( T_{RR} = 0.5 + 0.1 \cdot \text{rand} \) is sampled.

Step 7.2 Generate \( P_r = \text{rand} \).

Step 7.2.1 If \( P_r \leq L_{RR} \), create \( x^i_{\text{new}} \) using two-student learning and perform function evaluation.

Step 7.2.2 Else, create \( x^i_{\text{new}} \) using three-student learning and perform function evaluation.

Step 7.3 Update \( L_{RR} \_ \text{Success} \) and \( L_{RR} \_ \text{Fail} \).

Step 7.3.1 If \( x^i_{\text{new}} \) is better than \( x^i \), add 1 point to the \( j \)-th element of \( L_{RR} \_ \text{Success} \).

Step 7.3.2 Else, add 1 point to the \( j \)-th element of \( L_{RR} \_ \text{Fail} \).

Step 8 Replace \( \{x^i\} \) by \( n_P \) best solutions from \( \{x^i\} \cup \{x^i_{\text{new}}\} \). Calculate the objective function value of \( x^i_{\text{new}} \).

Step 9 Update the diversity archive with the non-dominated solutions obtained from \( \{x^i\}_{\text{old}} \cup \{x^i\}_{\text{new}} \).

Step 10 End While