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)

<u>Input</u> **x** = { r_1, r_2, r_3, r_4, r_{px}, r_{py}, θ_0, x_{O2}, y_{O2}, $\theta^i{}_2$}

<u>Output</u> 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 θ_2 and solving Equation (1) for $\mathbf{r}_\mathrm{p}$ at each θ_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

<u>Input</u> infeasible **x** at elements $\theta_2^i, \theta_2^{i+1}, \ldots, \theta_2^N$

<u>Output</u> feasible **x** at elements $\theta_2^i, \theta_2^{i+1}, \ldots, \theta_2^N$

Generate a set of uniform random numbers $\{\alpha_1, \ldots, \alpha_N\}$, $\alpha_i \in (0, 1)$.

If $(\alpha_2 + \ldots + \alpha_N) \geq 2\pi$, 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 $i = 1$, $\theta^i_2 = \alpha_1$.

Step 2 Otherwise, $\theta^i_2 = \theta_2^{i-1} + \alpha_i$.

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

$S_1 = \delta_1$

$S_2 = 2\delta_3$

$S_3 = 2\delta_3 + \delta_2$

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

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 :$\mathbf{x}^{\text{best}}$, f^{best}

Initialization:

Step 0.1 Generate n_p initial students {$\mathbf{x}^i$} and perform function evaluations {f^i}.

Step 0.2 Initiate four 1×2 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 {$\mathbf{x}^i$} written as $\mathbf{M}_{avg}$.

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 $PTRR_j$.

Step 4.1.1 If $j = 1$ is selected, $T_{RR} = 0.4 + 0.1rand$ is sampled.

Step 4.1.2 Else, if $j = 2$ is selected, $T_{RR} = 0.5 + 0.1rand$ is sampled.

Step 4.2 Generate $P_r = rand$ and select a teacher.

Step 4.2.1 If $P_r \leq T_{RR}$, set the best solution as a teacher $\mathbf{M}_{\text{best}}$.

Step 4.2.2 Else, if $P_r > T_{RR}$, randomly select a solution in $\mathbf{A}_D$ and set it as a teacher $\mathbf{M}_{\text{best}}$.

Step 4.3 Create $\mathbf{x}^i_{\text{new}}$ using (10 – 12) and perform function evaluation.

Step 4.3.1 If $\mathbf{x}^i_{\text{new}}$ is better than $\mathbf{x}^i$, add 1 point to the j-th element of *TRR_Success*.

Step 4.3.2 Else, add 1 point to the j-th element of *TRR_Fail*.

Step 5 Replace $\{\mathbf{x}^i\}$ by n_P best solutions from $\{\mathbf{x}^i\}\cup\{\mathbf{x}^i_{\text{new}}\}$.

{*Learning Phase*}

Step 6 Calculate the probabilities of selecting the intervals for L_{RR} similar to that for T_{RR} in step 3. $PTRR_j = \dfrac{LRR\_Success_j}{LRR\_Success_j + LRR\_Fail_j}$

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*rand* is sampled.

Step 7.1.2 Else, if j = 2 is selected, T_{RR} = 0.5 + 0.1*rand* is sampled.

Step 7.2 Generate P_r = *rand*.

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

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

Step 7.3 Update *LRR_Success* and *LRR_Fail*.

Step 7.3.1 If $\mathbf{x}^i_{\text{new}}$ is better than $\mathbf{x}^i$, add 1 point to the j-th element of *LRR_Success*.

Step 7.3.2 Else, add 1 point to the j-th element of *LRR_Fail*.

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

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

Step 10 End While