Appendix for Domain-Independent Dynamic Programming: Generic State Space Search for Combinatorial Optimization*

Ryo Kuroiwa, J. Christopher Beck

Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada, ON M5S 3G8
ryo.kuroiwa@mail.utoronto.ca, jcb@mie.utoronto.ca

CP Models

We present the new CP models used in the experimental evaluation.

TSPTW

We adapt a CP model for a single machine scheduling problem with time windows (Booth et al. 2016) to TSPTW. Let x_i be an interval variable in a range $[a_i, b_i]$ with the length of 0, representing visiting customer i.

$$\begin{aligned}
\min \sum_{i \in N} & c_{i,\mathsf{Next}(x_i)} \\
\text{s.t. } & \mathsf{NoOverlap}([x_0, ..., x_{n-1}], \{c_{ij} \mid i, j \in N\}) \\
& \mathsf{First}(x_0) \\
& x_i : \mathsf{intervalVar}(0, [a_i, b_i]) \qquad \forall i \in N.
\end{aligned}$$

The first constraint ensures that interval variables are ordered in a sequence, and for two consecutive variables x_i and x_j, the start of x_j must be at least c_{ij} greater than the end of x_i. In the objective, $\mathsf{Next}(x_i)$ is the interval variable next to x_i in the sequence. For the last variable, we let $\mathsf{Next}(x_i) = x_0$. The second constraint ensures that the depot is visited first.

SALBP-1

For SALBP-1, we implement the CP model proposed by Bukchin and Raviv (2018) with the addition of the Pack global constraint (Shaw 2004). For an upper bound on the number of stations, instead of using a heuristic to compute it, we use $\bar{m} = \min\{n, 2\lceil \sum_{i \in N} t_i / c \rceil\}$ following the MIP model (Ritt and Costa 2018). Let $M = \{0, ..., \bar{m} - 1\}$ be the set of stations. Let m be a decision variable representing the number of stations, x_i be a decision variable representing the index of the station of task i, and y_j be the sum of the processing times of tasks scheduled in station j. The set of all direct and indirect predecessors of task i is

$$\tilde{P}_i = \{j \in N \mid j \in P_i \lor \exists k \in \tilde{P}_j, j \in \tilde{P}_k\}.$$

The set of all direct and indirect successors of task i is

$$\tilde{S}_i = \{j \in N \mid i \in P_j \lor \exists k \in \tilde{S}_i, j \in \tilde{S}_k\}.$$

Thus,

$$e_i = \left\lceil \frac{t_i + \sum_{k \in \tilde{P}_i} t_k}{c} \right\rceil$$

is a lower bound on the number of stations required to schedule task i,

$$l_i = \left\lfloor \frac{t_i - 1 + \sum_{k \in \tilde{S}_i} t_k}{c} \right\rfloor$$

is a lower bound on the number of stations between the station of task i and the last station, and

$$d_{ij} = \left\lfloor \frac{t_i + t_j - 1 + \sum_{k \in \tilde{S}_i \cap \tilde{P}_j} t_k}{c} \right\rfloor$$

is a lower bound on the number of stations between the stations of tasks i and j.

$$\begin{aligned}
\min \quad & m \\
\text{s.t. } & \mathsf{Pack}(\{y_j \mid j \in M\}, \{x_i \mid i \in N\}, \{t_i \mid i \in N\}) \\
& 0 \leq y_j \leq c && \forall j \in M \\
& e_i - 1 \leq x_i \leq m - 1 - l_i && \forall i \in N \\
& x_i + d_{ij} \leq x_j \qquad \forall j \in N, \forall i \in \tilde{P}_j, \\
& && \nexists k \in \tilde{S}_i \cap \tilde{P}_j : d_{ij} \leq d_{ik} + d_{kj} \\
& m \in \mathbb{Z} \\
& y_j \in \mathbb{Z} && \forall j \in M \\
& x_i \in \mathbb{Z} && \forall i \in N.
\end{aligned}$$

The first constraint ensures $x_i \in M$ and $\sum_{i \in N : x_i = j} t_i = y_j$. The second constraint ensures that the sum of the processing times does not exceed the cycle time. The third constraint states the lower and upper bounds on the index of the station of i. The fourth constraint is an enhanced version of the precedence constraint using d_{ij}.

Bin Packing

For the CP model for bin packing, we also use Pack. In addition, we ensure that item i is packed in the i-th or an earlier

*This document is an appendix for a paper published at the 33rd International Conference on Automated Planning and Scheduling (ICAPS 2023) (Kuroiwa and Beck 2023) bin.

$$\begin{aligned}
&\min \max_{i \in N} x_i + 1 \\
&\text{s.t. } \mathsf{Pack}(\{y_j \mid j \in M\}, \{x_i \mid i \in N\}, \{t_i \mid i \in N\}) \\
&\quad 0 \leq y_j \leq c && \forall j \in M \\
&\quad 0 \leq x_i \leq i && \forall i \in N \\
&\quad y_j \in \mathbb{Z} && \forall j \in M \\
&\quad x_i \in \mathbb{Z} && \forall i \in N.
\end{aligned}$$

We compute the upper bound $\bar{m}$ using the first-fit decreasing heuristic.

DP Models

We prove the properties of the DP models assumed in the paper and provide the DyPDL representations of the models.

Lower Bounds for SALBP-1 and Bin Packing

We show that the lower bounds used in the DP models for SALBP-1 and bin packing are valid. These lower bounds, LB1, LB2, and LB3 were originally proposed by Scholl and Klein (1997). The first lower bound, LB1, is originally defined as $\lceil \sum_{i \in N} t_i / c \rceil$. This bound relaxes the problem by allowing to split a task across multiple stations. In a state $\langle U, r \rangle$ ($\langle U, r, k \rangle$ for bin packing), we only need to schedule tasks in U, and we can schedule tasks in the current station, which has the remaining time of r. Therefore, we use $\lceil (\sum_{i \in U} t_i - r)/c \rceil$ as a lower bound.

The second lower bound, LB2, is originally defined as $\sum_{i \in N} w_i^2 + \lceil \sum_{i \in N} w_i'^2 \rceil$, where $w_i^2 = 1$ if $t_i > c/2$ and $w_i'^2 = 1/2$ if $t_i = c/2$. This bound only considers tasks i with $t_i \geq 2/c$. The first term, the number of tasks i with $t_i > 2/c$, is a lower bound because other tasks cannot be scheduled in the same station as i. For the remaining tasks, which have $t_i = 2/c$, two tasks can be scheduled in the same station, which results in the second term. In our model, in a state $\langle U, r \rangle$, we use the bound $\sum_{i \in U} w_i^2 + \lceil \sum_{i \in U} w_i'^2 \rceil$ if $r < c/2$ because we cannot schedule any tasks with $t_i \geq 2/c$ in the current station. If $r \geq c/2$, since we may use the current station, we subtract 1 from the bound.

The second lower bound, LB3, is based on a similar idea to LB2. It is originally defined as $\lceil \sum_{i \in N} w_i'^3 \rceil$ and only considers tasks i with $t_i \geq 3/c$. Therefore, we use the bound $\lceil \sum_{i \in U} w_i^3 \rceil$ if $r < c/3$ and subtract 1 from it otherwise.

Cost-Algebra for MOSP and Graph-Clear

We show that the cost expressions in the DP models for MOSP and graph-clear satisfy the property of cost-algebra. In these models, the cost expressions are in the form of $\max\{e_\tau(S), V(S[\![\tau]\!])\}$ instead of $e_\tau(S) + V(S[\![\tau]\!])$.

A cost-algebra (Edelkamp, Jabbar, and Lafuente 2005) is defined as a 6-tuple $\langle A, \sqcup, \times, \preceq, \mathbf{0}, \mathbf{1} \rangle$ where A is a set, $\times : A \times A \to A$ is a binary operator, $\preceq \ \in A \times A$ is a binary relation, $\mathbf{0}, \mathbf{1} \in A$, and $\sqcup : 2^A \to A$ is an operator to select one element from a subset of A. It must satisfy the following conditions.

1. $\forall a, b \in A, a \times b \in A$
2. $\forall a, b, c \in A, a \times (b \times c) = (a \times b) \times c$
3. $\forall a \in A, a \times \mathbf{1} = \mathbf{1} \times a = a$
4. $\forall a \in A, a \preceq a$
5. $\forall a, b \in A, a \preceq b \wedge b \preceq a \Rightarrow a = b$
6. $\forall a, b, c \in A, a \preceq b \wedge b \preceq c \Rightarrow a \preceq c$
7. $\forall a, b \in A, a \preceq b \vee b \preceq a$
8. $\forall B \subseteq A, \forall b \in B, \sqcup B \preceq b$
9. $\forall a \in A, a \preceq \mathbf{0}$ and $\mathbf{1} \preceq a$
10. $\forall a, b, c \in A, a \preceq b \Rightarrow a \times c \preceq b \times c$ and $c \times a \preceq c \times b$

Conditions 1-3 ensure that $\langle A, \times, \mathbf{1} \rangle$ is a monoid. Conditions 4-7 ensure that $\preceq$ is a total order. Condition 10 is called isotonicity.

Edelkamp, Jabbar, and Lafuente (2005) proved that $\langle \mathbb{R}^+ \cup \{+\infty\}, \min, +, \leq, +\infty, 0 \rangle$, which corresponds to the shortest path problem, satisfies the conditions. We show that a tuple $\langle \mathbb{R}^+ \cup \{+\infty\}, \min, \max, \leq, +\infty, 0 \rangle$ satisfies the conditions, which corresponds to the cost expresssions in the DP models for MOSP and graph-clear. The tuple $\langle \mathbb{R}^+ \cup \{+\infty\}, \max, 0 \rangle$ is a monoid since $\max\{a, b\} \in \mathbb{R}$, $\max\{a, \max\{b, c\}\} = \max\{\max\{a, b\}, c\}$, and $\max\{a, 0\} = \max\{0, a\} = a$. Conditions 4-9 hold since $\langle \mathbb{R}^+ \cup \{+\infty\}, \min, +, \leq, +\infty, 0 \rangle$ is a cost-algebra. For the isotonicity, $\max\{a, c\} \leq \max\{b, c\}$ and $\max\{c, a\} \leq \max\{c, b\}$ for $a \leq b$.

DyPDL Representations

We present DyPDL representations and YAML-DyPDL domain files of the DP models for CVRP in Table 1 and Listing 1, for SALBP-1 in Table 2 and Listing 2, for bin packing in Table 3 and Listing 3, for MOSP in Table 4 and Listing 4, and for graph-clear in Table 5 and Listing 5.

In SALBP-1, `open-station` is a forced transition in the YAML-DyPDL domain file. While it is not necessarily in theory because the other transitions are not applicable when `open-station` is applicable, it can be beneficial since a solver does not need to evaluate the preconditions of the other transitions.

$\mathcal{V}$	Type	Objects	Preference
U	set	customers N	
i	element	customers N	
l	numeric		less
k	numeric		less
$\mathcal{K}$	Type	Indices	
q	numeric		
m	numeric		
d_j	numeric	$j \in N$	
c_{jp}	numeric	$j, p \in N$	
c'_{jp}	numeric	$j, p \in N$	
S^0	$\langle U = N \setminus \{0\}, i = 0, l = 0, k = 1 \rangle$		
$\mathcal{B}$	$\{\{U = \emptyset, i = 0\}\}$		
$\mathcal{C}$	$\emptyset$		
h	0		
$\mathcal{T}$	eff	cost	pre
visit j	$U \leftarrow U \setminus \{j\}$ $i \leftarrow j$ $l \leftarrow l + d_j$	$c_{ij} + V(S[\![\tau]\!])$	$j \in U$ $l + d_j \leq q$
visit j via the depot	$U \leftarrow U \setminus \{j\}$ $i \leftarrow j$ $l \leftarrow d_j$ $k \leftarrow k + 1$	$c'_{ij} + V(S[\![\tau]\!])$	$j \in U$ $k < m$
return	$i \leftarrow 0$	$c_{i0} + V(S[\![\tau]\!])$	$U = \emptyset$ $i \neq 0$

Table 1: DyPDL representation of the DP model for CVRP. No forced transition exists in this model.

$\mathcal{V}$	Type	Objects	Preference
U	set	tasks N	
r	numeric		more
$\mathcal{K}$	Type	Objects	Indices
c	numeric		
t_i	numeric		$i \in N$
P_i	set	tasks N	$i \in N$
w^2_i	numeric		$i \in N$
w'^2_i	numeric		$i \in N$
w^3_i	numeric		$i \in N$
S^0	$\langle U = N, r = 0 \rangle$		
$\mathcal{B}$	$\{\{U = \emptyset\}\}$		
$\mathcal{C}$	$\emptyset$		
h	$\max \left\{ \begin{array}{l} \lceil (\sum_{i \in U} t_i - r)/c \rceil \\ \sum_{u \in U} w^2_i + \lceil \sum_{i \in U} w'^2_i \rceil - l^2 \\ \lceil \sum_{i \in U} w^3_i \rceil - l^3 \end{array} \right.$		
$\mathcal{T}$	eff	cost	pre
assign i	$U \leftarrow U \setminus \{i\}$ $r \leftarrow r - t_i$	$V(S[\![\tau]\!])$	$i \in U$ $P_i \cap U = \emptyset$ $r \geq t_i$
open a station	$r \leftarrow c$	$1 + V(S[\![\tau]\!])$	$U' = \emptyset$

Table 2: DyPDL representation of the DP model for SALBP-1, where $U' = \{i \in U \mid P_i \cap U = \emptyset \land r \geq t_i\}$, $l^2 = 1$ if $r \geq c/2$ and $l^2 = 0$ otherwise, and $l^3 = 1$ if $r \geq c/3$ and $l^3 = 0$ otherwise.

$\mathcal{V}$	Type	Objects	Preference	
U	set	items N		
r	numeric		more	
k	element	items N	less	
$\mathcal{K}$	Type	Indices		
c	numeric			
t_i	numeric	$i \in N$		
w^2_i	numeric	$i \in N$		
w'^2_i	numeric	$i \in N$		
w^3_i	numeric	$i \in N$		
S^0	$\langle U = N, r = 0, k = 0 \rangle$			
$\mathcal{B}$	$\{\{U = \emptyset\}\}$			
$\mathcal{C}$	$\emptyset$			
h	$\max \left\{ \begin{array}{l} \lceil (\sum_{i \in U} t_i - r)/c \rceil \\ \sum_{u \in U} w^2_i + \lceil \sum_{i \in U} w'^2_i \rceil - l^2 \\ \lceil \sum_{i \in U} w^3_i \rceil - l^3 \end{array} \right.$			
$\mathcal{T}$	eff	cost	pre	forced
pack i	$U \leftarrow U \setminus \{i\}$ $r \leftarrow r - t_i$	$V(S[\![\tau]\!])$	$i \in U$ $r \geq t_i$ $i + 1 \geq k$	$\bot$
open with i	$U \leftarrow U \setminus \{i\}$ $r \leftarrow c - t_i$ $k \leftarrow k + 1$	$1 + V(S[\![\tau]\!])$	$U^1 = \emptyset$ $i \in U$ $i \geq k$	$\top$

Table 3: DyPDL representation of the DP model for bin packing, where $U^1 = \{i \in U \mid r \geq t_i \land i + 1 \geq k\}$, $l^2 = 1$ if $r \geq c/2$ and $l^2 = 0$ otherwise, and $l^3 = 1$ if $r \geq c/3$ and $l^3 = 0$ otherwise.

$\mathcal{V}$	Type	Objects	
R	set	customers C	
O	set	customers C	
$\mathcal{K}$	Type	Objects	Indices
N_c	set	customers C	$c \in C$
S^0	$\langle R = C, O = \emptyset \rangle$		
$\mathcal{B}$	$\{\{R = \emptyset\}\}$		
$\mathcal{C}$	$\emptyset$		
h	0		
$\mathcal{T}$	eff	cost	pre
close c	$R \leftarrow R \setminus \{c\}$ $O \leftarrow O \cup N_c$	$\max \left\{ \begin{array}{l} V(S[\![\tau]\!]) \\ \lvert (O \cap R) \cup (N_c \setminus O) \rvert \end{array} \right.$	$c \in R$

Table 4: DyPDL representation of the DP model for MOSP. No forced transition exists in this model.

$\mathcal{V}$	Type	Objects	
C	set	nodes N	
$\mathcal{K}$	Type	Objects	Indices
N	set	nodes N	
a_c	numeric		$c \in N$
b_{ij}	numeric		$i, j \in N$
S^0	$\langle C = \emptyset \rangle$		
$\mathcal{B}$	$\{\{C = N\}\}$		
$\mathcal{C}$	$\emptyset$		
h	0		
$\mathcal{T}$	eff	cost	pre
sweep c	$C \leftarrow C \cup \{c\}$	$\max\{V(S[\![\tau]\!]), e(c, S)\}$	$c \notin C$

Table 5: DyPDL representation of the DP model for graph-clear, where $e(c, S) = a_c + \sum_{i \in N} b_{ci} + \sum_{i \in C} \sum_{j \in \overline{C} \setminus \{c\}} b_{ij}$. No forced transition exists in this model.

Listing 1: YAML-DyPDL domain file for CVRP.

```
objects:
  - customer
state_variables:
  - name: U
    type: set
    object: customer
  - name: i
    type: element
    object: customer
  - name: l
    type: integer
    preference: less
  - name: k
    type: integer
    preference: less
tables:
  - name: q
    type: integer
  - name: m
    type: integer
  - name: d
    type: integer
    args: [customer]
  - name: c
    type: integer
    args: [customer, customer]
    default: 0
  - name: c-via-depot
    type: integer
    args: [customer, customer]
base_cases:
  - [(is_empty U), (= i 0)]
reduce: min
cost_type: integer
transitions:
  - name: visit
    parameters: [{ name: j, object: U }]
    preconditions: [(<= (+ l (d j)) q)]
    effect:
      U: (remove j U)
      i: j
      l: (+ l (d j))
    cost: (+ cost (c i j))
  - name: visit-via-depot
    parameters: [{ name: j, object: U }]
    preconditions: [(< k m)]
    effect:
      U: (remove j U)
      i: j
      l: (d j)
      k: (+ k 1)
    cost: (+ cost (+ c-via-depot i j))
  - name: return
    preconditions:
      - (is_empty U)
      - (!= i 0)
    effect:
      i: 0
    cost: (+ cost (c i 0))
dual_bounds:
  - 0
```

Listing 2: YAML-DyPDL domain file for SALBP-1.

```
objects:
  - task
state_variables:
  - name: U
    type: set
    object: task
  - name: r
    type: integer
    preference: greater
tables:
  - { name: c, type: integer }
  - name: t
    type: integer
    args: [task]
  - name: P
    type: set
    object: task
    args: [task]
  - name: w2_1
    type: integer
    args: [task]
  - name: w2_2
    type: continuous
    args: [task]
  - name: w3
    type: continuous
    args: [task]
base_cases:
  - - (is_empty U)
reduce: min
cost_type: integer
transitions:
  - name: assign
    parameters: [{ name: i, object: U }]
    preconditions:
      - (is_empty
          (intersection U (P i)))
      - (<= (t i) r)
    effect:
      U: (remove i U)
      r: (- r (t i))
    cost: cost
  - name: open-station
    forced: true
    preconditions:
      - forall: [{ name: i, object: U }]
        condition: >
          (or
            (> (t i) r)
            (> |(intersection U (P i))|
               0))
    effect:
      r: c
    cost: (+ cost 1)
dual_bounds:
  - (ceil (/ (- (sum t U) r) c))
  - (- (+ (sum w2_1 U)
          (ceil (sum w2_2 U)))
       (if (>= r (/ c 2.0)) 1 0))
  - (- (ceil (sum w3 U))
       (if (>= r (/ c 3.0)) 1 0))
```

Listing 3: YAML-DyPDL domain file for bin packing.

```
objects:
  - item
state_variables:
  - name: U
    type: set
    object: item
  - name: r
    type: integer
    preference: greater
  - name: k
    type: element
    object: item
    preference: less
tables:
  - name: c
    type: integer
  - name: t
    type: integer
    args: [item]
  - name: w2_1
    type: integer
    args: [item]
  - name: w2_2
    type: continuous
    args: [item]
  - name: w3
    type: continuous
    args: [item]
base_cases:
  - - (is_empty U)
reduce: min
cost_type: integer
transitions:
  - name: pack
    parameters: [{name: i, object: U}]
    preconditions:
      - (<= (t i) r)
      - (>= (+ i 1) k)
    effect:
      U: (remove i U)
      r: (- r (t i))
    cost: cost
  - name: open-with
    forced: true
    parameters: [{name: i, object: U}]
    preconditions:
      - (>= i k)
      - forall: [{name: j, object: U}]
        condition: (> (t j) r)
    effect:
      U: (remove i U)
      r: (- c (t i))
      k: (+ 1 k)
    cost: (+ cost 1)
dual_bounds:
  - (ceil (/ (- (sum t U) r) c))
  - (- (+ (sum w2_1 U)
          (ceil (sum w2_2 U))
       (if (>= r (/ c 2.0)) 1 0))
  - (- (ceil (sum w3 U))
       (if (>= r (/ c 3.0)) 1 0))
```

Listing 4: YAML-DyPDL domain file for MOSP.

```
objects:
  - customer
state_variables:
  - name: R
    type: set
    object: customer
  - name: O
    type: set
    object: customer
tables:
  - name: "N"
    type: set
    object: customer
    args:
      - customer
base_cases:
  - - (is_empty R)
reduce: min
cost_type: integer
transitions:
  - name: close
    parameters:
      - name: c
        object: R
    effect:
      R: (remove c R)
      O: (union O (N c))
    cost: >
      (max cost
        |(union (intersection O R)
                (difference (N c) O))|)
dual_bounds:
  - 0
```

Listing 5: YAML-DyPDL domain file for graph-clear.

```
objects:
  - node
state_variables:
  - name: C
    type: set
    object: node
tables:
  - name: "N"
    type: set
    object: node
  - name: a
    type: integer
    args:
      - node
  - name: b
    type: integer
    args:
      - node
      - node
    default: 0
base_cases:
  - - (is_subset N C)
reduce: min
cost_type: integer
transitions:
  - name: sweep
    parameters:
      - name: c
        object: node
    preconditions:
      - (not (is_in c C))
    effect:
      C: (add c C)
    cost: >
      (max cost
           (+ (a c)
              (+ (sum b c N)
              (sum b C (remove c ~C)))))
dual_bounds:
  - 0
```

Experimental Results

In addition to the number of instances solved to optimality, we also evaluate the computational time to find an optimal solution. We take the average time over instances solved by all methods. Furthermore, we evaluate the best lower bound found by the algorithm. In CAASDy, the minimum f-value of states in the open list is a lower bound on the optimal solution. For each instance, we compute the ratio of the lower bound found by a method to the best lower bound found by all the methods. Concretely, if there are methods $1..., n$, and method i finds a lower bound L_i, then, the lower bound ratio is defined as $\frac{L_i}{\max_{j=1,...,n} L_j}$. Thus, higher is better and 1.0 is the maximum.

For TSPTW, we also evaluate the decision diagram-based solver, ddo (Gillard, Schaus, and Coppé 2020) since it was previously used in TSPTW. We use the 'barrier' solver of ddo[1] (Coppé, Gillard, and Schaus 2022). Since the original version is used to solve problems to minimize the makespan objective, we modified the code so that it minimizes the total travel time.

References

Booth, K. E.; Tran, T. T.; Nejat, G.; and Beck, J. C. 2016. Mixed-Integer and Constraint Programming Techniques for Mobile Robot Task Planning. *IEEE Robot. and Autom. Lett.*, 1(1): 500–507.

Bukchin, Y.; and Raviv, T. 2018. Constraint Programming for Solving Various Assembly Line Balancing Problems. *Omega*, 78: 57–68.

Coppé, V.; Gillard, X.; and Schaus, P. 2022. Branch-and-Bound with Barrier: Dominance and Suboptimality Detection for DD-Based Branch-and-Bound. arXiv:2211.13118.

Edelkamp, S.; Jabbar, S.; and Lafuente, A. L. 2005. Cost-Algebraic Heuristic Search. In *Proc. AAAI*, 1362–1367.

Gillard, X.; Schaus, P.; and Coppé, V. 2020. Ddo, a Generic and Efficient Framework for MDD-Based Optimization. In *Proc. IJCAI*, 5243–5245.

Kuroiwa, R.; and Beck, J. C. 2023. Domain-Independent Dynamic Programming: Generic State Space Search for Combinatorial Optimization. In *Proc. ICAPS*.

Ritt, M.; and Costa, A. M. 2018. Improved Integer Programming Models for Simple Assembly Line Balancing and Related Problems. *Int. Trans. Oper. Res.*, 25(4): 1345–1359.

Scholl, A.; and Klein, R. 1997. SALOME: A Bidirectional Branch-and-Bound Procedure for Assembly Line Balancing. *INFORMS J. Comput.*, 9(4): 319–335.

Shaw, P. 2004. A Constraint for Bin Packing. In *Proc. CP*, 648–662.

[1]https://github.com/vcoppe/ddo-barrier

	MIP			CP			Ddo			DP		
TSPTW	#	time	LB	#	time	LB	#	time	LB	#	time	LB
Dumas (135)	121	0.39	0.97	36	52.61	0.43	114	0.11	0.86	**135**	**0.04**	**1.00**
GDE (130)	71	1.38	0.71	4	289.20	0.16	35	1.53	**0.73**	**77**	**0.07**	0.71
OT (25)	**0**	-	0.00	**0**	-	0.66	**0**	-	0.32	**0**	-	**0.67**
AFG (50)	33	546.92	0.80	7	16.02	0.40	30	0.30	0.81	**45**	**0.04**	**0.95**
Total (340)	225	61.20	0.77	47	69.58	0.34	179	0.26	0.77	**257**	**0.04**	**0.86**
CVRP	#	time	LB	#	time	LB	#	time	LB	#	time	LB
A, B, E, F, P (90)	**26**	-	**0.94**	0	-	0.05	-	-	-	4	-	0.22
SALBP-1	#	time	LB	#	time	LB	#	time	LB	#	time	LB
Small (525)	**525**	0.25	**1.00**	**525**	0.19	**1.00**	-	-	-	**525**	**0.04**	**1.00**
Medium (525)	**518**	35.28	**1.00**	501	15.30	**1.00**	-	-	-	509	**2.38**	0.99
Large (525)	317	103.44	0.75	404	3.37	**0.99**	-	-	-	**414**	**1.20**	0.88
Very large (525)	0	-	0.00	155	-	**0.97**	-	-	-	**204**	-	0.54
Total (2100)	1360	37.31	0.69	1585	6.47	**0.99**	-	-	-	**1652**	**1.17**	0.85
Bin Packing	#	time	LB	#	time	LB	#	time	LB	#	time	LB
Falkenauer U (80)	25	64.49	0.94	**36**	2.29	**1.00**	-	-	-	33	**1.61**	0.47
Falkenauer T (80)	37	147.01	**1.00**	**56**	8.16	**1.00**	-	-	-	27	**7.19**	0.40
Scholl 1 (720)	**605**	16.88	0.95	533	19.13	**1.00**	-	-	-	517	**15.70**	0.85
Scholl 2 (480)	354	34.11	0.97	**445**	**0.37**	**1.00**	-	-	-	335	3.12	0.73
Scholl 3 (10)	0	-	**1.00**	**1**	-	**1.00**	-	-	-	0	-	0.09
Falkenauer U	25	64.49	0.94	**36**	2.29	**1.00**	-	-	-	33	**1.61**	0.47
Wäscher (17)	2	788.96	**1.00**	**10**	3.22	**1.00**	-	-	-	**10**	**1.29**	0.62
Schwerin 1 (100)	80	-	**1.00**	**96**	-	**1.00**	-	-	-	0	-	0.06
Schwerin 2 (100)	54	-	**1.00**	**61**	-	**1.00**	-	-	-	0	-	0.09
Hard 28 (28)	**0**	-	**1.00**	**0**	-	**1.00**	-	-	-	**0**	-	0.32
Total (1615)	1157	30.89	0.97	**1238**	11.09	**1.00**	-	-	-	922	**10.19**	0.66
MOSP	#	time	LB	#	time	LB	#	time	LB	#	time	LB
Constraint Modelling Challenge (46)	41	10.32	0.94	**44**	4.53	0.96	-	-	-	**44**	**0.06**	**1.00**
SCOOP Project (24)	8	144.30	0.64	**23**	0.50	**0.99**	-	-	-	16	**0.04**	0.95
Faggioli and Bentivoglio (300)	130	88.66	0.75	**300**	1.77	**1.00**	-	-	-	298	**0.04**	**1.00**
Chu and Stuckey (200)	44	98.50	0.47	70	58.10	0.49	-	-	-	**125**	**0.07**	**1.00**
Total (570)	223	76.94	0.66	437	10.57	0.82	-	-	-	**483**	**0.05**	**1.00**
Graph-Clear	#	time	LB	#	time	LB	#	time	LB	#	time	LB
Planar (60)	16	447.49	0.98	1	297.12	0.82	-	-	-	**20**	**0.18**	**1.00**
Random (75)	8	65.35	0.85	3	9.44	0.52	-	-	-	**25**	**0.41**	**1.00**
Total (135)	24	160.89	0.91	4	81.36	0.65	-	-	-	**45**	**0.36**	**1.00**
Average ratio	0.48			0.41			-			**0.59**		

Table 6: Number of instances solved to optimality ('#'), the time to solve averaged over instances solved by all the methods ('time'), and the average lower bound ratio to the best lower bound found by all the methods ('LB'). 'Average ratio' is the ratio of optimally solved instances in each problem class averaged over all problem classes.