Submitted to *INFORMS Journal on Computing*
manuscript (Please, provide the manuscript number!)

Decremental State-Space Relaxations for the Basic Traveling Salesman Problem with a Drone: Online Supplement

Marcos Blufstein, Gonzalo Lera-Romero, Francisco J. Soulignac
Universidad de Buenos Aires. Facultad de Ciencias Exactas y Naturales. Departamento de Computación. Buenos Aires, Argentina. CONICET-Universidad de Buenos Aires. Instituto de Investigación en Ciencias de la Computación (ICC). Buenos Aires, Argentina. mblufstein@dc.uba.ar, gleraromero@dc.uba.ar, fsoulign@dc.uba.ar

Appendix A: Proof of Dominance Rules

To satisfy the journal's length policy (JLP), we only prove Rule 4. The proofs of the other rules are similar.

Rule 4 *Let p and q be partial routes that are ng-feasible and forked, and $w \in V_c \setminus F^{\mathrm{D}}(q)$. If $|p| = |q|$, $t(p) = t(q)$, $F^{\mathrm{T}}(p) \subseteq F^{\mathrm{T}}(q)$, $w \notin F^{\mathrm{D}}(p)$, $\bar{\tau}(p) \leq \bar{\tau}(q)$, and $\bar{c}(p) + c^{\mathrm{F}}(d(p), w) \leq \bar{c}(q) + c^{\mathrm{F}}(d(q), w)$, then p dominates q via w.*

Proof. Suppose $r_q = q + q_1 + q_2$ is an *ng*-feasible route for some sequence of moves $q_1 = (v_1, \mathrm{t}), \ldots, (v_k, \mathrm{t})$, (w, d). To prove that p dominates q via w (Definition 4), we must see that: $r_p = p + q_1 + q_2$ is a route, r_p is *ng*-feasible, and $\bar{c}(r_p) \leq \bar{c}(r_q)$. Route r_p starts at the depot because so does p. Moreover, r_p is a well-defined partial route because r_q is a route, $t(p) = t(q)$, and $w \neq d(p)$ because $w \notin F^{\mathrm{D}}(p)$. By definition, both $q + q_1$ and $p + q_1$ are combined. Hence, by induction, r_p ends at the depot because so does r_q. Moreover, since $|p| = |q|$, we obtain that $|r_p| = |r_q| = n + 2$. Summing up, r_p is a route.

Regarding the *ng*-feasibility of r_p, recall that $F^{\mathrm{T}}(p) \subseteq F^{\mathrm{T}}(q)$ by hypothesis. Then, by induction on $|q'|$, (a) $F^{\mathrm{T}}(p + q') \subseteq F^{\mathrm{T}}(q + q')$ for every sequence of moves q'. In particular, as both $p + q_1$ and $q + q_1$ are combined, we obtain that $F^{\mathrm{D}}(p + q_1) = F^{\mathrm{T}}(p + q_1) \cup \{v_k\} \subseteq F^{\mathrm{T}}(q + q_1) \cup \{v_k\} = F^{\mathrm{D}}(q + q_1)$. Then, another induction on $|q'|$ is enough to conclude that (b) $F^{\mathrm{D}}(p + q_1 + q') \subseteq F^{\mathrm{D}}(q + q_1 + q')$ for every sequence of moves q'. Finally,

observe that $w \notin \{v_1, \dots, v_k\}$ because $q + q_1$ is *ng*-feasible, while $w \notin F^{\mathrm{D}}(p)$ by hypothesis. Therefore, (c) $w \notin F^{\mathrm{D}}(p + q_1)$. Altogether, (a)–(c) imply that r_p is *ng*-feasible.[1]

To end the proof, we must check that $\bar{c}(r_p) \leq \bar{c}(r_q)$. Clearly, $\tau = \sum_{i=0}^{k-1} c^{\mathrm{T}}(v_i v_{i+1})$ is the time consumed by the truck to go from $v_0 = t(p) = t(q)$ to v_k in q_1. Similarly, $u = u_w + \sum_{i=1}^{k} u_{v_i}$ is the sum of the dual values corresponding to the vertices traversed by q_1, while $\bar{c}_2 = \bar{c}((v_k, \mathrm{d}) + q_2)$ is the u-cost of the partial route traversed after both $p + q_1$ and $q + q_1$. Therefore

$$\begin{aligned}
\bar{c}(r_p) &= \bar{c}(p) + \max\left\{\tau(p) + \tau, c^{\mathrm{F}}(d(p), w) + c^{\mathrm{J}}(w, v_k)\right\} - u + \bar{c}_2 \\
&= \max\left\{\bar{\tau}(p) + \tau, \bar{c}(p) + c^{\mathrm{F}}(d(p), w) + c^{\mathrm{J}}(w, v_k)\right\} - u + \bar{c}_2 \\
\text{(hypothesis)} &\leq \max\left\{\bar{\tau}(q) + \tau, \bar{c}(q) + c^{\mathrm{F}}(d(q), w) + c^{\mathrm{J}}(w, v_k)\right\} - u + \bar{c}_2 \\
&= \bar{c}(q) + \max\left\{\tau(q) + \tau, c^{\mathrm{F}}(d(q), w) + c^{\mathrm{J}}(w, v_k)\right\} - u + \bar{c}_2 = \bar{c}(r_q).
\end{aligned}$$

□

Appendix B: Correctness of the Bidirectional Search

We prove Theorems 1 and 2, omitting some details due to the JLP. For simplicity, we restate Theorem 1.

Theorem 1 *The following conditions are equivalent for a route r of D.*

1. *r is ng-feasible,*
2. *r is ng-feasible at some $1 \leq m \leq n + 2$, and*
3. *r is ng-feasible at every $1 \leq m \leq n + 2$.*

Proof. $1 \Rightarrow 2$ and $3 \Rightarrow 1$ are trivial. Regarding $2 \Rightarrow 3$, we first prove that 2 implies 4: if $m > 1$, then r is *ng*-feasible at $m - 1$. Suppose $r = p \oplus (v, x) \oplus q$ for $|p| = m$, and consider the following cases.

Case 1: $x = \mathrm{t}$, so $t(p) = t(q)$, and p and q are combined. Hence, $p = p' + (w, \mathrm{d})$ for some partial route p' and vertex w. We divide this case into two alternatives.

Case 1.1: p' is combined, i.e., the truck carries the drone from $v' = t(p') = d(p')$ to $v = w$ in the last move of p. In this case, $r = p' \oplus q'$ for the partial route $q' = q + (v', \mathrm{d})$ of D^{-1}. Certainly, p' is *ng*-feasible by definition. Moreover, either $v' \notin N(v)$ or $v' \in F^{\mathrm{T}}(p) = (F^{\mathrm{T}}(p') \cup \{v'\}) \cap N(v)$. In the former case, $v' \notin F^{\mathrm{T}}(q)$ because $F^{\mathrm{T}}(q) \subseteq N(v)$; in the latter case, $v' \notin F^{\mathrm{T}}(q)$ because r is *ng*-feasible at m, thus $F^{\mathrm{T}}(p) \cap F^{\mathrm{T}}(q) = \emptyset$. Consequently, q' is *ng*-feasible. Finally, suppose $z \in F^{\mathrm{T}}(q') \subseteq N(v')$ and observe that $z \neq v'$ because $v' \notin F^{\mathrm{T}}(q) \subseteq F^{\mathrm{T}}(q') \cup \{v\}$. If $z = v$, then $z \notin F^{\mathrm{T}}(p')$ because $p = p' + (v, \mathrm{d})$ is *ng*-feasible. Otherwise, $z \in F^{\mathrm{T}}(q) \supseteq F^{\mathrm{T}}(q') \setminus \{v\}$ and, therefore, $z \notin F^{\mathrm{T}}(p) \subseteq F^{\mathrm{T}}(p') \cup \{v\}$ because r is *ng*-feasible at m. In both cases, $z \notin F^{\mathrm{T}}(p')$, thus $F^{\mathrm{T}}(p') \cap F^{\mathrm{T}}(q') = \emptyset$. Altogether, r is *ng*-feasible at $m - 1$.

Case 1.2: p' is forked. Similar to Case 1.1 and omitted.

Case 2: $x = \mathrm{d}$, so p is forked and, therefore, $p = p' + (w, \mathrm{t})$ for some partial route p' and vertex w. Consider the following possibilities.

[1] Observe that the rule is false when Definition 1 is restricted to those routes in which the truck never moves alone to the vertex of the drone, because nothing prevents the truck to visit $d(p)$ within $q + q_1$ when $d(q) \neq d(p)$.

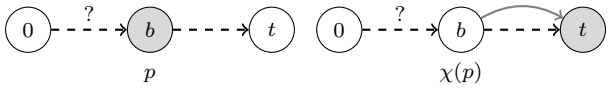

Figure 1 **The partial route $\chi(p)$ associated to a partial route p.**

Case 2.1: p' is combined, thus the truck splits from the drone at $w' = t(p')$, and it travels alone to w in the last move of p. In this case, $r = p' \oplus q'$ for the partial route $q' = q + (w', \mathrm{t}) + (v, \mathrm{d})$. The partial route p' is *ng*-feasible by hypothesis. If $w' \in N(w)$, then $w' \in F^{\mathrm{T}}(p) = (F^{\mathrm{T}}(p') \cup \{w'\}) \cap N(w)$, and $w' \notin F^{\mathrm{T}}(q) \subseteq N(w)$ because r is *ng*-feasible at m. Consequently, $q + (w', \mathrm{t})$ is *ng*-feasible. Similarly, $v \notin F^{\mathrm{D}}(q)$ because r is *ng*-feasible at m and, consequently, $v \notin F^{\mathrm{D}}(q + (w', \mathrm{t})) = F^{\mathrm{D}}(q) \cup \{w'\}$ because $v \neq w' \in F^{\mathrm{D}}(p)$. This means that q' is also *ng*-feasible. Suppose $z \in F^{\mathrm{T}}(q')$. If $z = v$, then $z \notin F^{\mathrm{T}}(p') \subseteq F^{\mathrm{D}}(p') \cup \{w\} = F^{\mathrm{D}}(p)$ because r is *ng*-feasible at m. Also, $z \neq w'$ because $w' \notin F^{\mathrm{T}}(q')$. If $z = w$, then $z \notin F^{\mathrm{T}}(p')$ because $p = p' + (w, \mathrm{t})$ is *ng*-feasible. Finally, if $z \notin \{v, w, w'\}$, then $z \in F^{\mathrm{T}}(q) \subseteq N(w)$. Hence, as r is *ng*-feasible at m, it follows that $z \notin F^{\mathrm{T}}(p) = (F^{\mathrm{T}}(p') \cup \{w'\}) \cap N(w)$ and, therefore, $z \notin F^{\mathrm{T}}(p')$. Summing up, $F^{\mathrm{T}}(p') \cap F^{\mathrm{T}}(q') = \emptyset$ and, altogether, r is *ng*-feasible at $m - 1$.

Case 2.2: p' is forked. Similar to the previous cases, and omitted.

To prove $2 \Rightarrow 3$, we apply $2 \Rightarrow 4$ several times. Details are omitted. □

Corollary 1 *A route r of D is ng-feasible if and only if its corresponding backward route $(0, \mathrm{d}) \oplus r$ of D^{-1} is ng-feasible.*

Before dealing with Theorem 2, we prove two technical results that explain why the labeling algorithm discards a route. These results are the basis for completion bounds and the variable fixing method. Is for the latter that we consider an additional configuration of the labeling algorithm in which some dominance rules are disabled. Let $\chi(p)$ be the partial route with the same truck and drone paths as p, except the truck carries the drone after $d(p)$ (Figure 1). That is, $\chi(p)$ is obtained from p by replacing (v, t) with (v, d) for every move (v, t) that appears after the last move of the form $(\bullet, \mathrm{d})$; certainly, $p = \chi(p)$ when p is combined.

Proposition 1 *For $1 \leq m \leq n + 2$, let $\mathcal{P}_m$ be the family of ng-feasible partial routes of length m that the labeling algorithm obtains when applied on a transport network D. If $r = p \oplus (v, x) \oplus q$ is an ng-feasible route of D and $|p| = m$, then D has an ng-feasible route $r' = p' \oplus (v', x') \oplus q'$ such that $p' \in \mathcal{P}_m$, $\bar{c}(r') \leq \bar{c}(r)$, and:*

1. *if $x = \mathrm{t}$, then $x' = \mathrm{t}$ and $q' = q$,*
2. *if $x = \mathrm{d}$ and $x' = \mathrm{t}$, then $q' = \chi(q) + (v', \mathrm{d})$,*
3. *if $x = x' = \mathrm{d}$, then $q' \in \{q, \chi(q)\}$ and $v' \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p')$,*
4. *if Rules 5–6 are disabled and $x = \mathrm{d}$, then $x' = \mathrm{d}$, $v' = v$, and $q' = q$.*

Proof. The proof is by induction on m. The base case $m = 1$ is true because $\mathcal{P}_1 = \{(0, \mathrm{d})\}$ and $r = (0, \mathrm{d}) \oplus q$ for every *ng*-feasible route q of D^{-1}. For the inductive step $m > 1$, we consider the following alternatives.

Case 1: $x = \mathrm{t}$ and $p = \phi + (v, \mathrm{d})$ for some partial route ϕ. In this case, the truck carries the drone from $w = t(\phi)$ to $t(p) = v$ in the last move of p, thus $r = \phi \oplus (q + (w, \mathrm{d}))$. By the inductive hypothesis, D has an

ng-feasible route $r_0 = \phi' \oplus (q + (w, \mathrm{d}))$ such that $\phi' \in \mathcal{P}_{m-1}$ and $\bar{c}(r_0) \leq \bar{c}(r)$. Note that $v \notin F^{\mathrm{T}}(\phi')$ because $p_0 = \phi' + (v, \mathrm{d})$ is a partial route of the *ng*-feasible route $r_0 = p_0 \oplus q$. Then, at the $(m-1)$-th iteration of the labeling algorithm, ϕ' will get extended into p_0. Certainly, $\bar{c}(p_0) \leq \bar{c}(p)$ because $\bar{c}(\phi') \leq \bar{c}(\phi)$. Consider the longest sequence $p_0, \ldots, p_k$ such that the labeling algorithm detects that p_{j+1} dominates p_j according to Rule 3, for $0 \leq j < k$. This sequence is well defined because the labeling algorithm stops after a finite number of steps. By Rule 3, each p_j is combined, $j \geq 0$, thus Rule 3 is the only reason why the labeling algorithm may discard p_k. Hence, $p_k \in \mathcal{P}_m$ by the maximality of k. Moreover, by Definition 2, $r_j = p_j \oplus q$ is an *ng*-feasible route with $\bar{c}(r_j) \leq \bar{c}(r_{j-1})$ for every $j > 0$, thus $r' = r_k$ satisfies the proposition.

Case 2: $x = \mathrm{t}$ and $p = \phi + (w, \mathrm{d})$ for some partial route ϕ and $w \neq v$. In other words, in the last move of p, the drone leaves $d(\phi)$ to visit w, and then it travels to v where it meets the truck again. Thus, $r = \phi \oplus (w, \mathrm{d}) \oplus q$. Since $x = \mathrm{t}$, q is combined. This means that $\chi(q) = q$, hence either Condition 2 or 3 holds for ϕ. In both cases, the existence of the desired route r' follows as Case 1; details are omitted.

Case 3: $x = \mathrm{d}$, thus $p = \phi + (w, \mathrm{t})$ for some partial route ϕ and $w \neq v$. This time, the truck travels alone from some vertex $z = t(\phi)$ to $t(p) = w$. Consequently, $r = \phi \oplus (v, \mathrm{d}) \oplus (q + (z, \mathrm{t}))$ and we have three possibilities by the inductive hypothesis for $m - 1$.

Case 3.1: Condition 2 is true for ϕ. We omit the details.

Case 3.2: Condition 3 is true for ϕ and $q' = \chi(q + (z, \mathrm{t})) = \chi(q) + (z, \mathrm{d})$. We omit the details.

Case 3.3: Condition 3 is true for ϕ and $q' = q + (z, \mathrm{t})$, thus D has a route $r_0 = \phi' \oplus (v_0, \mathrm{d}) \oplus (q + (z, \mathrm{t}))$ such that $\phi' \in \mathcal{P}_{m-1}$, $\bar{c}(r_0) \leq \bar{c}(r)$ and $v_0 \notin F^{\mathrm{D}}_{\mathrm{lbl}}(\phi')$. Moreover, $v_0 = v$ if Rules 5–6 are disabled. At the $(m-1)$-th iteration of the labeling algorithm, ϕ' will be extended into $p_0 = \phi' + (w, \mathrm{t})$ because $w \notin F^{\mathrm{T}}(\phi')$ by the *ng*-feasibility of $r_0 = p_0 \oplus (v_0, \mathrm{d}) \oplus q$. As before, $\bar{c}(p_0) \leq \bar{c}(p)$ follows because $\bar{c}(\phi') \leq \bar{c}(\phi)$. The *ng*-feasibility of r_0 also implies that $v_0 \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p_0)$ the moment p_0 is created. Later, v_0 may be inserted into $F^{\mathrm{D}}_{\mathrm{lbl}}(p_0)$ because of a dominance relation. By Definition 6, if p_0 is detected to be pre-dominated by a partial route p_1 via v_0 according to Rule 6, then $r_1 = p_1 \oplus (\chi(q) + (z, \mathrm{d}))$ is an *ng*-feasible route with $\bar{c}(r_1) \leq \bar{c}(r_0) \leq \bar{c}(r)$. In this case, we can obtain the desired route r' from r_1 as in Case 1. Suppose, then, that p_0 is not pre-dominated by Rule 6, and consider the longest sequence $p_0, \ldots, p_k$ such that the labeling algorithm detects that either p_{j+1} dominates p_j via $v_{j+1} = v_j$ according to Rule 4 or p_{j+1} forking to v_{j+1} dominates p_j via v_j according to Rules 5. Again, this sequence is well defined because the labeling algorithm runs for a finite amount of time. Certainly, p_j is not pre-dominated according to Rule 6 for $j > 0$; otherwise, the algorithm never compares p_{j-1} against p_j. Moreover, when p_{j-1} and p_j are compared, we know that $v_j \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p_j)$, and $v_j = v_{j-1}$ if Rule 5 is disabled. Altogether, $p_k \in \mathcal{P}_m$ and $v_k \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p_k)$ by the maximality of k. Moreover, by Definitions 4 and 5, $r_j = p_j \oplus (v_j, \mathrm{d}) \oplus q_j$ is an *ng*-feasible route with $\bar{c}(r_j) \leq \bar{c}(r_{j-1})$ for $j > 0$, where $q_j \in \{q_{j-1}, \chi(q_{j-1})\}$, and $q_{j+1} = q_j$ if Rule 5 is disabled. Summing up, $r' = r_k$ satisfies the proposition. □

Proposition 2 *For $1 \leq m \leq n + 2$, let $\mathcal{P}_m$ be the family of ng-feasible partial routes of length m that the labeling algorithm obtains when applied on the reverse D^{-1} of a transport network D. If $r = p \oplus (v, x) \oplus q$ is an ng-feasible route of D and $|q| = m$, then D has an ng-feasible route $r' = p' \oplus (v', x') \oplus q'$ such that $q' \in \mathcal{P}_m$, $\bar{c}(r') \leq \bar{c}(r)$, and:*

1. *if* $x = \mathrm{t}$*, then* $x' = \mathrm{t}$ *and* $p' = p$*,*
2. *if* $x = \mathrm{d}$ *and* $x' = \mathrm{t}$*, then* $p' = \chi(p) + (v', \mathrm{d})$*.*
3. *if* $x = x' = \mathrm{d}$*, then* $p' \in \{p, \chi(p)\}$*,* $v' \notin F^{\mathrm{D}}_{\mathrm{lbl}}(q')$*, and either* $p' = \chi(p)$ *or* $v' = v$ *or* q' *is forked.*

Proof. Omited due to the JLP. □

Theorem 2 *Algorithm 1 (bidirectional search) outputs at least one ng-feasible route of the minimum u-cost.*

Proof. Let $\bar{c}$ be the minimum among the u-costs of the ng-feasible routes, and $\mathcal{F}$ and $\mathcal{B}$ be the families of partial routes the labeling algorithm computes at Steps 1 and 2 of Algorithm 1, respectively. By Theorem 1, it suffices to see that $\mathcal{R}$ has route with u-cost $\bar{c}$ that is ng-feasible at m. We consider three cases in which $|p| = m$, where Case i assumes that Case $(i-1)$ does not hold for $i \in \{2, 3\}$.

Case 1: some route $r = p \oplus q$ has $\bar{c}(r) = \bar{c}$. By Proposition 1, D has an ng-feasible route $r' = p' \oplus q$ with $p' \in \mathcal{F}$ and $\bar{c}(r') = \bar{c}$. Then, by Proposition 2, D has an ng-feasible route $r^* = p' \oplus q'$ such that $q' \in \mathcal{B}$ and $\bar{c}(r^*) = \bar{c}$. By Theorem 1, r^* is ng-feasible at m, thus Step 5 of Algorithm 1 outputs r^*.

Case 2: some route $r = p \oplus (v, \mathrm{d}) \oplus q$ such that q is combined has $\bar{c}(r) = \bar{c}$. Note that $\chi(q) = q$ by definition. Then, as Case 1 does not hold, Proposition 1 implies that D has an ng-feasible route $r' = p' \oplus (v', \mathrm{d}) \oplus q$ such that $p' \in \mathcal{F}$, $\bar{c}(r') = \bar{c}$, and $v' \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p')$. Certainly, $r' = (p' + (v', \mathrm{d})) \oplus q$. Then, by Proposition 2, D has an ng-feasible route $r^* = (p' + (v', \mathrm{d})) \oplus q'$ such that $\bar{c}(r^*) = \bar{c}$ and $q' \in \mathcal{B}$. Certainly, $v' \notin F^{\mathrm{D}}(q') = F^{\mathrm{D}}_{\mathrm{lbl}}(q')$ because $r^* = p' \oplus (v', \mathrm{d}) \oplus q'$ is ng-feasible. By Theorem 1, r^* is ng-feasible at m, thus Step 5 of Algorithm 1 outputs r^*.

Case 3: some route $r = p \oplus (v, \mathrm{d}) \oplus q$ such that q is forked has $\bar{c}(r) = \bar{c}$. As Cases 1 and 2 do not hold, Proposition 1 implies that D has an ng-feasible route $r' = p' \oplus (w, \mathrm{d}) \oplus q$ such that $p' \in \mathcal{F}$ and $\bar{c}(r') = \bar{c}$. Similarly, as Case 1 does not hold, Proposition 2 implies that D has an ng-feasible route $r^* = p' \oplus (z, \mathrm{d}) \oplus q'$ such that $q' \in \mathcal{B}$, $\bar{c}(r') \leq \bar{c}(r)$, and $z \notin F^{\mathrm{D}}_{\mathrm{lbl}}(q')$. As before, $z \notin F^{\mathrm{D}}(p')$ because $r^* = p' \oplus (z, \mathrm{d}) \oplus q'$ is ng-feasible. Moreover, by Theorem 1, r^* is ng-feasible at m, thus Step 5 of Algorithm 1 outputs r^*. □

Appendix C: Correctness of the Variable Fixing Methods

In this section, we prove the correctness of the variable fixing methods in Section 4. For this, we have to prove Proposition 1, and we must show that:

G1. We can generate a family $\mathcal{B}$ of partial routes of D^{-1} that is ub-consistent for D, as required to invoke Algorithms 2–4,

G2. The family $\mathcal{F}$ of partial routes of D that Step 1 of Algorithm 2 computes is ub-consistent for D^{-1}, as this fact is used by Algorithms 3 and 4, and

G3. Step 1 of Algorithm 2 solves a pricing problem, i.e., the route $r^* \in \mathcal{F}$ minimizing $\bar{c}(r^*)$ solves the pricing problem defined by the dual vector u. This fact is exploited by Step 5 of Algorithm 4.

The variable fixing methods rely upon computing the completion bound of partial routes. As anticipated, the completion bound $cb_{\mathcal{B}}(p)$ of a partial route p of D depends on a family $\mathcal{B}$ of partial routes of D^{-1}.

Definition 1 *If p is combined, then*

$$cb_{\mathcal{B}}(p) = \sum_{v \in V} u_v + \min\{\bar{c}(r) \mid r = p \oplus q \text{ is ng-feasible for } q \in \mathcal{B}\}$$

. Otherwise, $cb_{\mathcal{B}}(p) = \sum_{v \in V} u_v + \min\{cb_1, cb_2, cb_{\mathcal{B}}(\chi(p))\}$, where

$$cb_1 = \min\{\bar{c}(r) \mid r = p \oplus (v, \mathrm{d}) \oplus q \text{ is ng-feasible for some } q \in \mathcal{B} \text{ forked, and } v \notin F^{\mathrm{D}}(p) \cup F^{\mathrm{D}}_{\mathrm{lbl}}(q)\}, \text{ and}$$

$$cb_2 = \min\{\bar{c}(r) \mid r = p \oplus (v, \mathrm{d}) \oplus q \text{ is ng-feasible for some } q \in \mathcal{B} \text{ combined, and } v \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p) \cup F^{\mathrm{D}}_{\mathrm{lbl}}(q)\}.$$

Of course, $cb_{\mathcal{B}}(p) = \infty$ when the minimum is undefined.

Observation 1 *For every ng-feasible partial route p of D, $cb_{\mathcal{B}}(p) \geq \overline{lb}$.*

Recall that we omit $\mathcal{B}$ when it is clear by context. Definition 1 adds $\sum_{v \in V} u_v$ to provide a meaningful comparison against $\overline{lb}$ and $c(r)$. However, for the purposes of computing cb within a labeling algorithm, it is convenient to consider the bound without this term, to compare it against the u-costs. Let $\overline{cb}(p) = cb(p) - \sum_{v \in V} u_v$ and $\overline{ub} = ub - \sum_{v \in V} u_v$. Certainly, $cb(p) < ub$ if and only if $\overline{cb}(p) < \overline{ub}$, and $c(r) < ub$ if and only if $\bar{c}(r) < \overline{ub}$ for every elementary route r. Note that the latter does not hold for non-elementary routes.

It is worth mentioning that, in many routing problems (e.g. Baldacci et al. 2012), $\overline{cb}(p)$ is defined as the **minimum** u-cost obtainable when p continues its journey through some partial route q of $\mathcal{B}$. Instead, we are considering only a **lower bound** of this u-cost. The reason for our definition is that the latter is flawed when p is forked: due to the dominance rules, the merge of p and q could have a u-cost greater than what the labeling algorithm can find. This is why Definition 1 takes $\chi(p)$ into account. Definition 1 also considers $F^{\mathrm{D}}(p)$ when q is forked because, as explained in Section 3.1, the forward and backward algorithms can restrict the drone from visiting different customers in their subsequent moves.

As we state in Section 4.1, $\overline{cb}(p)$ should be a lower bound of the minimum u-cost achievable by extending p. This hypothesis is not satisfied by every family of routes $\mathcal{B}$ of D^{-1}. For instance, if $\mathcal{B} = \emptyset$, then $\overline{cb}_{\mathcal{B}}(p) = \infty > \bar{c}(r)$ for every *ng*-feasible route r extending p. To provide meaningful lower bounds, with $\overline{cb}(p) \leq \bar{c}(r)$, we need $\mathcal{B}$ to be consistent. Moreover, as we compute completion bounds within labeling algorithms that discard routes with $\bar{c}(r) \geq \overline{ub}$ (e.g., Step 1 of Algorithm 2), $\mathcal{B}$ has to be *ub*-consistent.

Definition 2 *Let $ub \in \mathbb{R}$, and $\mathcal{B}$ be a family of ng-feasible partial routes of D^{-1}. We say that $\mathcal{B}$ is ub-*consistent *for D when $\overline{cb}_{\mathcal{B}}(p) \leq \bar{c}(r)$ for every ng-feasible route $r = p \oplus (v, x) \oplus q$ of D with $\bar{c}(r) < \overline{ub}$ such that either $v \notin F^{\mathrm{D}}_{\mathrm{lbl}}(p)$ or $x = \mathrm{t}$.*

Proposition 3 *Let $ub \in \mathbb{R}$, and $\mathcal{B}$ be a family of partial routes of D^{-1}. If for every route $r = p \oplus (v, x) \oplus q$ with $\bar{c}(r) < \overline{ub}$ such that either $v \in F^{\mathrm{D}}_{\mathrm{lbl}}(p)$ or $x = \mathrm{t}$, there exists an ng-feasible route $r' = p' \oplus (v', x') \oplus q'$ in D such that $\bar{c}(r') \leq \bar{c}(r)$, $q' \in \mathcal{B}$, and r' satisfies Conditions 1–3 of Proposition 2, then $\mathcal{B}$ is ub-consistent for D.*

Proof. Omitted to satisfy the JLP. □

Corollary 2 *Let $\mathcal{B}$ be the family of ng-feasible partial routes the labeling algorithm obtains when applied on D^{-1}. For every $ub \in \mathbb{R}$, $\mathcal{B}$ is ub-consistent for D.*

Algorithm 1 is an alternative to Corollary 2 when obtaining an *ub*-consistent family $\mathcal{B}$. Algorithm 1 is the same method that Algorithm 3 executes at Step 1 to compute the *ub*-consistent family it requires at Step 4. The same is true for Algorithm 4 and its Steps 1 and 8. Before proving the *ub*-consistency of $\mathcal{B}$, we relate the family of partial routes computed by Algorithm 1 to the one the labeling algorithm finds.

Algorithm 1 Bounded labeling

Input: $ub \in \mathbb{R}$, a family of partial routes $\mathcal{F}$ of D that is *ub*-consistent for D^{-1}, and a set of dominance rules Γ.
Output: a family of partial routes $\mathcal{B}$ of D^{-1} that is *ub*-consistent for D.
1: **output** the family $\mathcal{B}$ of partial routes of D^{-1} that is obtained by running a labeling algorithm in which:
- the dominance rules in Γ are deactivated.
- no partial route p with $cb_{\mathcal{F}}(p) \geq ub$ is ever created.

Proposition 4 *Let $ub \in \mathbb{R}$, $\mathcal{F}$ be a family of partial routes of D that is ub-consistent for D^{-1}, and $\mathcal{B}'$ be the family of ng-feasible partial routes of D^{-1} the labeling algorithm in Section 3 obtains when executed on D^{-1}. If $\mathcal{B}$ is the family of ng-feasible partial routes that Algorithm 1 outputs when applied to D with input ub and $\mathcal{F}$, then $\mathcal{B} = \{p \in \mathcal{B}' \mid cb_{\mathcal{F}}(p) < ub\}$.*

Proof. For $1 \leq m \leq n+2$, let $\mathcal{B}_m$ and $\mathcal{B}'_m$ be the families of partial routes of $\mathcal{B}$ and $\mathcal{B}'$ having length m, respectively. We prove by induction on m that $\mathcal{B}_m = \{p \in \mathcal{B}'_m \mid cb_{\mathcal{F}}(p) < ub\}$. The base case $m = 1$ is trivial because $(0, \mathrm{d})$ is the unique partial route in $\mathcal{B}'_1$, and it belongs to $\mathcal{B}_1$ if and only if $cb_{\mathcal{F}}((0, \mathrm{d})) < ub$. For the inductive step, let p_{m+1} be a partial route of D^{-1} with $|p| = m+1$, and consider the following cases.

Case 1: $cb_{\mathcal{F}}(p_{m+1}) \geq ub$, so $p_{m+1} \notin \mathcal{B}_{m+1}$ by Step 1 of Algorithm 1.

Case 2: $cb_{\mathcal{F}}(p_{m+1}) < ub$. Let p_m be the parent of p_{m+1} in the tree generated by the labeling algorithm (i.e., p_m is the partial route with the first m moves of p_{m+1}). By definition of cb, D^{-1} has a route r of the form $r = p_{m+1} \oplus (\bullet, \bullet) \oplus \bullet$ with $\bar{c}(r) < ub$. Certainly, r is also of the form $r = p_m \oplus (\bullet, \bullet) \oplus \bullet$, thus $cb_{\mathcal{F}}(p_m) < ub$ follows because of the *ub*-consistency of $\mathcal{F}$. Consider the subcases below.

Case 2.1: $p_{m+1} \notin \mathcal{B}'_{m+1}$, so either $p_m \notin \mathcal{B}'_m$ or the labeling algorithm forbids the extension from p_m to p_{m+1} after it applies a dominance rule between p_m and a partial route q_m. In the former case, $p_m \notin \mathcal{B}_m$ by the inductive hypothesis, thus $p_{m+1} \notin \mathcal{B}_{m+1}$. In the latter case, Rules 3–6 imply that $|q_m| = m$, hence $q_m \in \mathcal{B}'_m$. Moreover, by Definitions 2–6, q_m is extendable into a route r_q with $\bar{c}(r_q) \leq \bar{c}(r) < ub$. Therefore, by the inductive hypothesis, $q_m \in \mathcal{B}_m$ and, consequently, the dominance rule that prevents the extension from p' to p in the labeling algorithm applies in Algorithm 1 as well. Summing up, $p \notin \mathcal{B}_{m+1}$.

Case 2.2: $p_{m+1} \in \mathcal{B}'_{m+1}$. Omitted due to the JLP. □

Corollary 3 *Let $ub \in \mathbb{R}$, and $\mathcal{F}$ be a family of partial routes of D that is ub-consistent for D^{-1}. If Algorithm 1 obtains the family of ng-feasible partial routes $\mathcal{B}$ when applied on D^{-1} with input ub and $\mathcal{F}$, then $\mathcal{B}$ is ub-consistent for D.*

Corollary 4 *Let $ub \in \mathbb{R}$, and $\mathcal{F}$ be a family of partial routes of D that is ub-consistent for D^{-1}. Recall that c^* is the optimum value of the TSP-D. If $ub > c^*$, then $\mathcal{B}$ contains an ng-feasible route r^* of D^{-1} that minimizes $\bar{c}(r^*)$.*

Steps 1 and 8 of Algorithm 4 are equivalent to Algorithm 1 with Rules 5–6 disabled. By Corollary 4, these steps find a solution to the pricing problem as a by-product, i.e., G3 is true. Regarding G2, observe that *ub*-consistency implicitly depends on the neighborhoods that define *ng*-feasibility, as Definition 2 must hold for every *ng*-feasible route r of D. Let Ω be the family of *ng*-feasible routes of D and suppose a family of partial routes $\mathcal{B}$ is *ub*-consistent for D. If we consider a family of *ng*-feasible routes $\Omega' \subseteq \Omega$, then $\mathcal{B}$ is also *ub*-feasible for D with respect to Ω'. Then, by Corollary 3, the family $\mathcal{F}$ computed in Step 1 of Algorithm 2 is *ub*-compatible for D when Algorithm 3 executes Step 4. Corollary 3 also implies that $\mathcal{B}$ is *ub*-consistent for D each time Algorithm 4 runs Step 8. Altogether, G2 holds as well.

Regarding G1, we can always rely upon the labeling algorithm in Section 3 because of Corollary 2. However, as Algorithms 2–4 always run immediately after Algorithm 1 (bidirectional search), we take advantage of the faster method described in Algorithm 2.

Algorithm 2 Initial *ub*-consistent family.

Input: $ub \in \mathbb{R}$, and the families $\mathcal{F}$ and $\mathcal{B}$ computed by Algorithm 1.

Output: $\mathcal{B}$, updated into a family of partial routes that is *ub*-consistent for D.

1: **let** m be the parameter used to compute $\mathcal{F}$ and $\mathcal{B}$ by Algorithm 1.
2: Warm-start Algorithm 1 setting with the set $\mathcal{B}$.
3: Run Algorithm 1 on D^{-1} with input ub, $\mathcal{F}$, and $\Gamma = \emptyset$, updating $\mathcal{B}$ with the partial routes of length $> m$.

Let $\mathcal{F}$ and $\mathcal{F}'$ be the families of *ng*-feasible partial routes of D that Algorithm 1 and labeling algorithm in Section 3 output, respectively. Despite $\mathcal{F} \subseteq \mathcal{F}'$, Algorithm 2 outputs the same family $\mathcal{B}$ of partial routes when its input $\mathcal{F}$ is replaced by $\mathcal{F}'$, given that no partial route in $\mathcal{F}' \setminus \mathcal{F}$ ever participates in a completion bound. As $\mathcal{F}'$ is *ub*-consistent for D^{-1} by Corollary 3, Algorithm 1 with input ub and $\mathcal{F}'$ outputs a family $\mathcal{B}'$ that is *ub*-consistent for D. Certainly, $\mathcal{B}' \subseteq \mathcal{B}$ because the only difference between Step 1 of Algorithm 1 and Step 3 of Algorithm 2 is that the former discards those partial routes with $|p| \leq m$. Consequently, $\mathcal{B}$ is *ub*-consistent for D, and G1 holds as required.

To end this section, we prove Proposition 1, copied below, observing that Step 1 of Algorithm 2 is equivalent to Algorithm 1 with Rules 5–6 disabled.

Proposition 1 *Let $\mathcal{F}$ be the family of partial routes computed at Step 1 of Algorithm 2. If $r = (p + (v, x_v)) \oplus (w, x_w) \oplus q$ is an elementary route with $c(r) < ub$, then there exists $p' \in \mathcal{F}$ such that $|p'| = |p|$, $t(p') = t(p)$, $p' + (v, x_v)$ is ng-feasible, and $cb(p' + (v, x_v)) < ub$.*

Proof. By Proposition 4, we can prove Proposition 1 for the family of *ng*-feasible routes $\mathcal{F}' \supseteq \mathcal{F}$ the labeling algorithm in Section 3 outputs when executed on D. This is done applying Proposition 1; we omit the details.

Appendix D: Neighborhood Augmentation

Recall that, given a non-elementary route r, the NA method computes a new family of *ng*-feasible routes $\Omega' \subseteq \Omega - \{r\}$. As membership to Ω depends on a family of neighborhoods, we obtain Ω' by inserting vertices to $N(v)$, for $v \in V$, in such a way that r is not *ng*-feasible for the updated N.

By definition, r is nothing but a sequence of moves $(v_1, x_1), \ldots, (v_{n+2}, x_{n+2})$. Since r is not elementary, $v_i = v_j$ for some $1 \leq i < j \leq n+2$. Let p be the partial route $(v_1, x_1), \ldots, (v_{j-1}, x_{j-1})$, and b be the maximum such that $x_b = \mathrm{d}$. Let $e = j - 1$ if p is combined, and $e = b$ otherwise. We say that $C = v_i, \ldots, v_e$ is a *cycle* when its vertices are pairwise different. By Definition 1, r is not *ng*-feasible for N when $v_i \in N(v_p)$ for every $i < p \leq e$. To exclude r from Ω', the NA method *separates* C by inserting v_i into $N(v_p)$ for every $i < p \leq e$.

The neighborhoods grow too fast if all the cycles in r are separated, making each subsequent pricing problem intractable. Let $s = 1 + \max\{|N(v)| \text{ s.t. } v \in V\}$. We limit the effects of the NA method by processing the cycles of r in a non-decreasing order of size. We separate $C = v_1, \ldots, v_j$ if $|N(v_i \cup \{v_1\})| \leq s$ for every $1 < i \leq j$ the moment we process C.

Appendix E: Heuristics for the Pricing Problem

We adapt Algorithm 1 (bidirectional labeling) into three faster methods that heuristically search for a route with a negative u-cost. In a nutshell, we strengthen the dominance rules to discard non-promising partial routes. These methods are heuristic because they discard those routes with a minimum u-cost when some of their partial routes are not promising, i.e., these methods do not satisfy Theorem 2. The *cost heuristic* adapts Algorithm 1 by not requiring p and q to be combined in Rule 3. The *cost+ng heuristic* further relaxes Rule 3 by not requiring $F^{\mathrm{T}}(p) \subseteq F^{\mathrm{T}}(q)$. Finally, the *ng heuristic* runs Algorithm 1 without condition $F^{\mathrm{T}}(p) \subseteq F^{\mathrm{T}}(q)$ in Rules 3–6. The `CG` method in Section 5.1 applies the cost+*ng*, cost, and *ng* heuristics in this order.

Appendix F: Handling Side Constraints

In this section, we incorporate some usual constraints of truck-and-drone routing problems by making small changes to our algorithm. The goal is to evaluate how well our algorithm adapts to these new constraints with minimal adaptations. Therefore, we change only the definition of partial routes and their costs, ignoring the opportunities to speed up the algorithm. Also, we keep the same parameters for the experimentation, even though the assumption that the pricing problem is too time-consuming is no longer valid.

F.1. Loops

When loops are allowed, the drone may launch and land at the same vertex. Recall that partial routes of the TSP-D are sequences of moves, each of which is a pair (v, x) with $x \in \{\mathrm{d}, \mathrm{t}\}$. For the TSP-D with loops, we consider a new kind of move (v, ℓ) to indicate the visit to v with a loop. After loops, the recursive computation of the truck and drone locations is as follows. If $p = q + (v, \ell)$, then $v \neq t(q)$, and either q is combined or $t(q) = d(q) = 0$, so the drone flies from $d(q)$ to v and then back to $t(p) = d(p) = t(q)$. Regarding the cost of p, we define the time $c(p)$ required by both vehicles to reach $d(p)$, and the time $\tau(p)$ consumed by the truck after $d(p)$. Certainly, $\tau(p) = 0$ because p is combined. Regarding $c(p)$, observe that $\tau(q) = 0$ if q is combined, and $c(q) = 0$ otherwise because the truck visits every customer other than v. So:

$$c(q + (v, \ell)) = c(q) + \max\{\tau(q), c^{\mathrm{F}}(d(q), v) + c^{\mathrm{J}}(v, t(q))\}$$

Finally, all the algorithms treat each move (v, ℓ) as they treat (v, d).

F.2. Incompatible Customers

If the drone is not allowed to visit *incompatible* customers in a set $V_F \subseteq V_c$, then we set $c^{\mathrm{F}}(w,v,m) = c^{\mathrm{J}}(w,v,m) = \infty$ for every $v \in V_F$, every $w \in V$, and every $1 \leq m \leq n+2$.

F.3. Drone's Flying Range

When the drone has a limited *flying range* of ρ units of time, we change the definition of partial routes: for a forked partial route q, the sequence of moves $p = q + (v,x)$, for $v \in V_c$ and $x \in \{\mathrm{d}, \ell\}$, is a partial route if and only if the drone has to fly no more than ρ time. Regarding implementation: we fix to ∞ the costs of the arcs that the drone cannot take without flying more than ρ time; we enforce the new definition of partial routes throughout the different algorithms; and we adjust the dominance rules to ensure the dominating partial route p extends into an *ng*-feasible route for every extension of the dominated route q (e.g., Rule 4 requires $c^{\mathrm{T}}(d(p),w) \leq c^{\mathrm{T}}(d(q),w)$ to guarantee that the drone can visit w regardless of the future rendezvous vertex).

F.4. Drone Cannot Land and Wait

If the drone can fly for ρ units of time and it must wait in the air when it reaches the rendezvous vertex before the truck, then the truck cannot travel alone for more than ρ time. Thus, for a forked partial route q, the sequence of moves $p = q + (v,x)$ is a partial route if and only if either 1. $x = \mathrm{t}$, and the time that the truck travels alone is $\tau(p) \leq \rho$, or 2. $x \in \{\mathrm{d}, \ell\}$, and the drone has to travel no more than ρ time. We adapt the implementation as before, further restricting Rule 4, asking $\tau(p) \leq \tau(q)$.

F.5. Launch and Rendezvous Times

Suppose that launching the drone from the truck at a vertex v consumes a *launch time* $\tau^{\mathrm{F}}(v)$, while retrieving the drone at v requires a *rendezvous time* $\tau^{\mathrm{J}}(v)$. We adapt the recursive case for the cost $c(p)$ of a partial route $p = q + (v,x)$, as follows. If the truck leaves the drone at $t(q)$, then we add $\tau^{\mathrm{F}}(t(q))$ to the cost. Similarly, if the drone travels alone to meet the truck at $t(q)$, then we add $\tau^{\mathrm{J}}(t(q))$ to the cost.

F.6. The truck cannot visit customers alone

To forbid the truck from visiting customers alone, we exclude $q + (v, \mathrm{t})$ from the set of partial routes, for every forked partial route q and every $v \in V$.

F.7. Computational Results

We evaluate our algorithms in the variants of the TSP-D proposed by Roberti and Ruthmair (2021). Refer to Roberti and Ruthmair (2021) for the details of the experiments, which we omit to satisfy the JLP.

`PFA` solves the 3600 instances with up to $n = 39$ customers, while `RR` fails to solve an instance with 19 customers on INCOMP(40%), at least one instance with 29 customers on each variant other than MHD, and many instances of each variant with 39 customers. In total, `RR` solves 2973 of these instances.

Table 1 depicts the number of optimal solutions found by `PFA` for BASIC, MHD, and POI for the instances with at least 49 customers. `PFA` failed to solve some instances with 49 customers for MHD and POI, even though these problems have a smaller state space than BASIC. At least three facts explain this behavior. First, BASIC runs the exact labeling algorithm on the entire network sporadicly. Second, some dominance rules are weaker in MHD and POI. Third, we tuned the parameters on BASIC only. The second and third

Table 1 Optimal solutions obtained by PFA for variants of the TSP-D.

Variant	α	#inst	n						All
			49	59	69	79	89	99	
BASIC	1	25	25	25	25	23	13	1	112
	2	25	25	25	25	22	19	10	126
	3	25	25	25	23	23	17	8	121
MHD	1	25	25	25	23	20	11	1	105
	2	25	25	25	23	18	8	3	102
	3	25	25	22	16	17	6	1	87
POI	1	25	25	24	23	24	14	7	117
	2	25	25	25	25	24	16	12	127
	3	25	23	25	24	25	17	8	122
All	1350	225	223	221	207	196	121	51	1019

reasons imply that more arcs get fixed in BASIC than in MHD and POI, so the state space of BASIC is seldom larger. The exception is the first execution of CG+F (Step 2 of Algorithm 6), where PFA solves the linear relaxation faster for MHD and POI, explaining why PFA solves more of the "easy" instances with $n = 99$ and $\alpha = 1$.