Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Alternative Filtering for the Weighted Circuit Constraint: Comparing Lower Bounds for the TSP and Solving TSPTW

Sylvain Ducomman and **Hadrien Cambazard** and **Bernard Penz**
Univ. Grenoble Alpes, G-SCOP, F-38000 Grenoble, France
CNRS, G-SCOP, F-38000 Grenoble, France
{firstname.name}@g-scop.grenoble-inp.fr

Abstract

Many problems, and in particular routing problems, require to find one or many circuits in a weighted graph. The weights often express the distance or the travel time between vertices. We propose in this paper various filtering algorithms for the weighted circuit constraint which maintain a circuit in a weighted graph. The filtering algorithms are typical cost based filtering algorithms relying on relaxations of the Traveling Salesman Problem. We investigate three bounds and show that they are incomparable. In particular we design a filtering algorithm based on a lower bound introduced in 1981 by Christophides *et al.*. This bound can provide stronger filtering than the classical Held and Karp's approach when additional information, such as the possible positions of the clients in the tour, is available. This is particularly suited for problems with side constraints such as time windows.

Baseline CP for TSP/TSPTW

Many problems, and in particular routing problems, require to find one or many circuits in a weighted graph. For example, the Traveling Salesman Problem (TSP) consists in finding a circuit of minimum total weight that visits all vertices of the graph. Time Windows are often added to the formulation to express the fact that a vertex or 'a client' can be visited only when available (TSPTW). To the best of our knowledge, the state-of-the-art to solve this problem is based on dynamic programming combined with column generation (Baldacci, Mingozzi, and Roberti 2012).

Related work: In Constraint Programming (CP), two approaches have been developed to tackle the TSPTW. The first approach (Pesant et al. 1998) uses redundant constraints in order to reduce the search space. In particular, a constraint to reduce the domains of the time windows is added as well as an arc elimination constraint for filtering possible direct successors of a vertex. But the objective function is propagated independently of the circuit constraint which leads to poor global lower bounds. The second approach (Focacci, Lodi, and Milano 2002) adds a lower bound computed with an assignment relaxation. Additionally, an effective filtering for weighted circuit was proposed in (Benchimol et al. 2012) based on the 1-tree relaxation by Held and Karp (Held and Karp 1970; 1971) but has not been used for the TSPTW. Our paper is based on the work of (Benchimol et al. 2012). We propose different filtering algorithms, all based on a relaxation of the TSP taking into account different aspects that are relevant for asymmetric or time-constrained cases such as the TSPTW. A second contribution is to show that the three bounds are incomparable and in particular the two bounds based on Lagrangian relaxation.

Problem definition: Let $G = (N, E)$ be a complete directed graph without loops with vertex set $N = \{0, \dots, n+1\}$, the vertex 0 (resp. $n+1$) corresponds to the start (resp. the end) of the route ($N^s = \{0, \dots, n\}$ and $N^e = \{1, \dots, n+1\}$). We denote by d_{ij} the distance (or cost) of arc (i, j). A salesman tour is defined as a path in G from vertex 0 to vertex $n+1$, visiting each vertex $1, \dots, n$ exactly once. The Traveling Salesman Problem (TSP) consists in finding a salesman tour of minimum distance in G. In presence of time constraints, each vertex i is associated to a time windows $[a_i, b_i]$, and each arc (i, j) to a traveling time t_{ij}. The service time of customer i is included in t_{ij}. The Traveling Salesman Problem with Time Windows (TSPTW) consists in identifying a shortest salesman tour visiting each vertex within its time window. Note that the problem has $n+1$ cities: n clients plus one depot where the tour starts and ends. In the following, we denote by $D(x)$, the domain of variable x and by $\overline{x}$ (resp. $\underline{x}$) the upper (resp. lower) bound of x.

Model

This section presents a Constraint Programming (CP) model for the TSPTW. The model is based on a set of variables $next_i \in N$ for each vertex $i \in N$ representing the immediate successor of the vertex i in the tour (for convenience $next_{n+1} = 0$). We also add the opposite variable $pred_i \in N$ for each vertex $i \in N$ representing the immediate predecessor of i ($pred_0 = n+1$). Let $dist_i$ for each vertex $i \in N$ be the cumulated distance of the tour when reaching vertex i. For time windows constraints, we add the variable $start_i \in [a_i, b_i]$ representing the cumulated traveling time along the route to reach vertex i. So $start_i$ can be seen as the starting time of the service for client i and its initial domain is defined by the corresponding time window.

The model is given by equations (1)-(6):

$$\textbf{Minimize } z \tag{1}$$

$$z = \sum_{i=0}^{n}(d_{i,next_i}) = \sum_{i=1}^{n+1}(d_{pred_i,i}) \tag{2}$$

$$\text{WEIGHTEDCIRCUIT}(next_0, ..., next_{n+1}, z) \tag{3}$$

$$dist_{next_i} = dist_i + d_{i,next_i} \quad \forall i \in N^s$$

$$dist_0 = 0 \tag{4}$$

$$start_{next_i} \geq start_i + t_{i,next_i} \quad \forall i \in N^s$$

$$start_0 = 0 \tag{5}$$

$$\text{INVERSE}([next_0, \ldots, next_{n+1}], [pred_0, \ldots, pred_{n+1}]) \tag{6}$$

Predecessors and successors are used to state the objective function (2) because $\sum_{i=0}^{n}(d_{i,next_i})$ and $\sum_{i=1}^{n+1}(d_{pred_i,i})$ can be different when all variables are not instantiated and thus provide incomparable lower bounds for z. Stating one of the two equalities would be correct but the use of both strengthen propagation. Constraints (4) and (5) represent quantities accumulated along the route and use the ELEMENT constraint (Van Hentenryck and Carillon 1988). Variables $next$ and $pred$ are linked so that: $next_i = j \Leftrightarrow pred_j = i, \forall (i,j) \in E$. This is expressed by the INVERSE constraint (6). The WEIHTEDCIRCUIT constraint (3) presented in (Benchimol et al. 2012) maintains a circuit in a weighted graph and provides a strong lower bound on z.
Redundant constraints (7)-(14) are then added and make use of variables *pos* and *b* to express filtering related to positions and precedences. For each arc (i,j), $b_{ij} \in \{0,1\}$ indicates if i is visited before j ($b_{ij} = 1$) or after j ($b_{ij} = 0$). Variables $pos_i \in \{0, \ldots, n+1\}$ for each $i \in N$ represent the position of vertex i in the tour ($pos_0 = 0$ and $pos_{n+1} = n+1$).

$$b_{ij} + b_{ji} = 1 \quad \forall (i,j) \in E \tag{7}$$

$$(b_{ij} = 1) \Rightarrow next_j \neq i \quad \forall (i,j) \in E \tag{8}$$

$$pos_j > pos_i + 1 \Rightarrow next_i \neq j \quad \forall (i,j) \in E \tag{9}$$

$$pos_j > pos_i \Leftrightarrow b_{ij} = 1 \quad \forall (i,j) \in E \tag{10}$$

$$\text{ALLDIFFERENT}(pos_0, \ldots, pos_{n+1}) \tag{11}$$

$$(b_{ij} = 1) \Rightarrow start_j \geq start_i + t_{ij} \quad \forall (i,j) \in E \tag{12}$$

$$pos_i = \sum_{j \in N} b_{ji} \quad \forall i \in N \tag{13}$$

$$start_i + t_{ij} > start_j \Rightarrow next_i \neq j \quad \forall (i,j) \in E \tag{14}$$

Note the redundant constraint (14) introduced in (Langevin et al. 1993) which is the arc elimination constraint.

Filtering algorithms

This section presents three different filtering algorithms for the weighted circuit constraint. The first one relies on the Held and Karp's relaxation for the TSP and is presented in (Benchimol et al. 2012). The second one is based on the assignment problem and was used by (Focacci, Lodi, and Milano 2002). We extend this algorithm with exact reduced cost computation. The last algorithm takes into account the time constraints of the TSPTW by using a relaxation of the TSP initially introduced in (Christofides, Mingozzi, and Toth 1981). All algorithms use a relaxation of the TSP to perform cost based filtering.

Filtering with Held and Karp's relaxation

We use the "1-tree relaxation" introduced by Held and Karp (Held and Karp 1970) which relaxes the constraint of the TSP enforcing each vertex to have a degree of two. A minimum 1-tree is a minimum spanning tree on vertices N^e plus the edge $(0, n+1)$ and the edge connected to 0 with the minimum weight. The bound provided by the "1-tree relaxation" is improved by adding Lagrangian multipliers (potentials) to each vertex penalizing the violation of the degree (Held and Karp 1970; 1971). The Lagrangian dual is solved with a subgradient procedure.
Therefore each vertex has a potential π_i and the cost of edge (i,j) includes the potentials of the two adjacent vertices $(d_{ij} + \pi_i + \pi_j)$. At each step of the subgradient, a valid relaxation of the TSP is available with the current potentials and a filtering algorithm can be applied. The cost of the minimum 1-tree that must (resp. do not) contain a given edge can be evaluated in order to detect whether the corresponding edge is forbidden (resp. mandatory) with respect to the current best known upper-bound. The efficient computation of the marginal and replacement costs of each edge is presented in (Benchimol et al. 2012) with a time complexity of $\mathcal{O}(n + m + nlog(n))$ for the calculation of forbidden edges and $\mathcal{O}(m\alpha(mn))$[1] for the mandatory edges. This relaxation is defined for undirected graph. In order to handle the directed case, two options have been investigated: using 1-arborescence instead of 1-tree or applying the Jonker and Volgenant's transformation (Jonker and Volgenant 1983). We implemented the latter following the advice of (Benchimol et al. 2012).

Filtering with Assignment Problem

The Assignment Problem (AP) is to find a perfect matching of minimum cost in a bipartite weighted graph. It provides a relaxation of the TSP by allowing sub-circuits. Let N^s and $V = \{n+i, \forall i \in N^e\}$ be the two disjoint sets of vertices of the bipartite graph for the assignment problem. Let $B = \{(i, n+j), \forall (i,j) \in E\}$ be the set of arcs, and the cost of the arc $d^B_{i,n+j} = d_{ij}, \forall (i,j) \in E$. The assignment problem is defined over the bipartite graph $G^B = (N^s, V, B, d^B)$. A lower bound is obtained by computing a maximum matching with minimum cost c^B in the graph G^B using the hungarian algorithm (Kuhn 2010) with a time complexity of $\mathcal{O}(n^3)$. We denote by P the set of arcs corresponding to a perfect matching of minimum cost c^B. The reduced cost for an arc (i,j) not in P, *i.e.* the minimum increase of the overall cost for setting $next_i$ to j. It is computed in practice by (Focacci, Lodi, and Milano 2002) using the linear formulation and thus represent a lower bound of the exact increase. However, exact reduced costs for the arcs can be obtained from shortest paths in the residual graph as in (Régin 2002). We apply to this purpose Johnson's algorithm for all pairs of shortest paths on the residual graph $G^R = (N^s, V, R, d^R)$ with $R = (B \backslash P) \cup P'$ and $P' = \{(j,i), \forall (i,j) \in P\}$. Johnson's algorithm has a time complexity of $\mathcal{O}(n^2 log(n) + nm)$. Let's denote by $sp_{ij}, \forall (i,j) \in R$ the value of the alternating shortest path in

[1] α is a functional inverse of Ackermann's function

the residual graph between i and j. An arc $(i,j) \in B \backslash P$ is a forbidden arc if its insertion cost, *i.e.* the cost of an optimal solution of the assignment problem that includes arc (i,j), exceeds the upper bound. Let Δ_{ij} be the reduced cost of the arc $(i,j) \in B \backslash P$, so that value j is filtered from $D(next_i)$ if $c^B + \Delta_{ij} > \overline{z}$. Let $j' \in V$ and $i' \in N^s$ be respectively the vertices currently matched to i and j in P. Δ_{ij} is computed as follow:

$$\Delta_{ij} = (d_{ij} + sp_{i'j'}) - (d_{ij'} + d_{i'j})$$

The first term represents the cost of the minimum maching including (i,j) while the second term is due to the removal of the two existing arcs in the current matching. In this case an arc (i,j) can only be mandatory if all other (i,k) are forbidden. Thus it is enough to compute forbidden arcs to detect the mandatory ones.

Filtering with n-path and Lagrangian relaxation

Alternative relaxations have been proposed for the TSP and we investigate filtering algorithms based on the path relaxations of (Christofides, Mingozzi, and Toth 1981).

n-path relaxation. Recall that the problem has $n+1$ cities (the depot and n clients) and that the depot is represented by vertex 0 (start of the tour) and vertex $n+1$ (end of the tour). A solution is an elementary path in G of exactly $n+1$ arcs. The idea is simply to relax the constraint that a vertex has to be visited exactly once but requiring the path to use exaclty $n+1$ arcs. A solution of this relaxation is thus a non-elementary path of $n+1$ arcs. $f^*(k,i)$ denotes the value of the optimal path of exactly k arcs reaching vertex i from vertex 0. We consider $f^*(1,i) = d_{0i}$, $\forall i \in \{1,\dots,n\}$ if $1 \in D(pos_i)$ and $f^*(k,i) = +\infty \, \forall i \in \{1,\dots,n+1\}, \forall k \notin D(pos_i)$. Values of $f^*(k,i)$ can be obtained by applying the recursive formula:

$$f^*(k,i) = \min_{j \in D(pred_i)} (f^*(k-1,j) + d_{ji})$$
$$\forall k \in \{2,\dots,n+1\}, \forall i \text{ s.t. } k \in D(pos_i) \quad (15)$$

$f^*(n+1,n+1)$ provides a lower bound of z^*. Similarly to $f^*(k,i)$ we can compute $f^*_{rev}(k,i)$ which is the value of the shortest path leaving vertex i and reaching vertex $n+1$ in exactly k arcs. The space complexity of this dynamic program is $O(n^2)$ but time complexity is $O(n^2 m)$ where m is the maximum domain size of the $pred$ variables. Note that the domains of the predecessor ($pred$) and position variables (pos) are used in (15) to restrict the state space. The aim of this algorithm is to take into account the reasonings on positions to inform the relaxation. This is not directly possible in the spanning tree relaxation and make sense for time constrained problems such as the TSPTW where reasonings about precedences and positions are often stronger than reasonings about direct predecessors/successors.

ln-path relaxation. In the n-path relaxation, the degree constraint of each vertex is relaxed similarly to the Held and Karp's bound. We can thus apply a similar Lagrangian relaxation of the degree constraints to penalize their violation and we refer to this relaxation as the **ln-path**. Let's consider a real multiplier ($\lambda_i \in \mathbb{R}$) associated to each vertex i and boolean variables $x_{ij} \in \{0,1\}$ to indicate whether $next_i = j$. The Lagrangian subproblem can be stated as:

$$\textbf{Minimize} \sum_{(i,j) \in E} (d_{ij} - \lambda_i - \lambda_j) x_{ij} + 2 \sum_{j \in N} \lambda_j \quad (16)$$

$$\text{N-PATH}(\{x_{ij} | \forall i \in N, j \in D(next_i)\}) \quad (17)$$

Note that this formulation is obtained by relaxing $\sum_{i \in D(pred_j)} x_{ij} + \sum_{i \in D(next_j)} x_{ji} = 2$ for each vertex $j \in N$. This degree constraint along with the n-path requirement would indeed ensure a valid tour. Finally, as mentioned by (Christofides, Mingozzi, and Toth 1981), the relaxation can be strengthened without increasing the time complexity by forbidding circuits of two arcs *i.e.* with three consecutive vertices such as x,y,x. This requires to store the value of the best path, $\phi^*(k,i)$, reaching vertex i in k arcs with a different predecessor for vertex i than the path supporting $f^*(k,i)$. We thus consider in the following that the n-path relaxation is a non-elementary shortest path of n+1 arcs without such circuits.

ln-path filtering algorithm: Since the relaxation involves the $next$ and pos variables, we can filter them at any iteration of the subgradient algorithm *i.e* for any given values of the multipliers. It is a generic methodology once the dynamic program is properly defined. We denote by C the constant term of the objective so that $C = 2 \sum_{j \in N} \lambda_j$.

- Forbidden arcs: $\forall i \in \{0,\dots,n\}, \forall j \in D(next_i)$
  $(\forall k \in D(pos_i), f^*(k,i) + d_{ij} + f^*_{rev}(n-k,j) - C > \overline{z})$
  $\Rightarrow next_i \neq j,$
  The expression $f^*(k,i) + d_{ij} + f^*_{rev}(n-k,j) - C$ is a lower of the cost of any solution using arc (i,j) as the k-th arc. This relies on the fact that $f^*(k,i)$ is a lower bound to reach vertex i from vertex 0 with k arcs, and $f^*_{rev}(n-k,j)$ is a lower bound to reach vertex $n+1$ from j using $n-k$ arcs. Thus the expression relates to a path of $k+1+n-k$ i.e $n+1$ arcs. By taking the minimum cost over all possible positions for arc (i,j) ($\forall k \in D(pos_i)$) we obtain a lower bound of any solution using arc (i,j).
- Forbidden positions: $\forall i \in \{1,\dots,n\}, \forall k \in D(pos_i)$
  $f^*(k,i) + f^*_{rev}(n-k+1,i) - C > \overline{z}$
  $\Rightarrow pos_i \neq k$

Any values of the multipliers can be used for filtering. The filtering requires the computation of f^*_{rev} which can be done in the same time complexity than f^* by a second call to the dynamic program.

n-path for filtering the time: The n-path relaxation itself can be applied to the time (t_{ij}) dimension to provide filtering related to timing. The dynamic program (15) can be written according to the time dimension and becomes:

$$f^*(k,i) = \min_{j \in D(pred_i)} (f^*(k-1,j) + t_{ji})$$
$$\forall k \in \{2,\dots,n+1\}, \forall i \text{ s.t. } k \in D(pos_i) \quad (18)$$

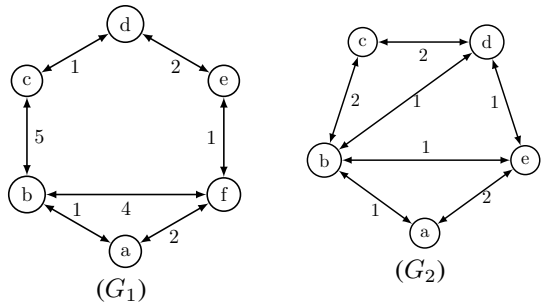

Figure 1: Graph G_1 satisfying $z_{np} > z_{oa} > z_{ap}$; Graph G_2 satisfying $z_{np} < z_{oa} < z_{ap}$

This relaxation alone can be used to filter the bound of variables pos and $start$:

- Minimum/maximum positions: $\forall i \in \{1, \ldots, n\}, \forall k \in D(pos_i)$, we can state that:

$$\underline{start_i} + f^*_{rev}(n-k+1, i) > \overline{start_{n+1}} \Rightarrow pos_i > k$$
$$f^*(k, i) > \overline{start_i} \Rightarrow pos_i < k$$

- Minimum/maximum starting time: the domains of the positions can also be used to strenghten the minimum and maximum starting time of each vertex. So $\forall i \in N$, we have:

$$start_i \leq \overline{start_{n+1}} - f^*_{rev}(n - \overline{pos_i} + 1, i)$$
$$start_i \geq f^*(\underline{pos_i}, i)$$

A subgradient approach similar to the one used for the Held and Karp's relaxation is used to solve the Lagrangian dual.

Comparison of the bounds

We now show that the relaxations are incomparable even for symmetric distances. It thus make sense to eventually use them together in order to improve the filtering. We first focus on the 1-arborescence, the assignment and the n-path relaxations without the use of Lagrangian multipliers. We then show that the directed Held and Karp's bound is also incomparable with the ln-path relaxation, both methods using Lagrangian relaxation.

1-arboresence, n-path, assignment. The values of the 1-arborescence, the n-path and the assignment relaxations are respectively denoted z_{oa}, z_{np} and z_{ap}. A 1-arborescence on G (as defined in the problem definition) is a graph with vertices $0, 1, \ldots, n+1$ consisting in an arborescence on the vertices $0, 1, \ldots, n+1$ rooted in 0 together with one in-going arc incident to 0. So every vertex has an in-degree of exactly one and can be reached from the root vertex 0.

Proposition 1 *z_{oa}, z_{np} and z_{ap} are incomparable.*

Proof: We proceed in two steps by highlighting two cases (G_1 and G_2) where two opposite rankings of the bounds hold. Let's start with the graph G_1 (Figure 1) and satisfying $z_{np} > z_{oa} > z_{ap}$. We check that:

- The n-path relaxation has two symmetric feasible solutions with a cost of 12 and one is shown Figure 2.b. Suppose the path starts with arc (ab). We are then facing two choices: (bf) or (bc). If one takes (bf), it is impossible to reach back vertex (a) in 6 arcs without performing a xyx cycle. (bc) is thus the only option and a tour must be done to reach (a) in 6 arcs. Due to the symmetry of G_1, the same reasoning holds if the path starts with (af).
- The cost of the minimum 1-arborescence is 10 (Figure 2.c). A minimum arborescence rooted in (a) is computed as (ab), (bf), (fe), (ed), (dc). The arc of minimum cost entering vertex (a) (which is the arc (ba)) must be added to to obtain the minimum 1-arborescence.
- A solution of the assignment problem can use only the arcs of minimum cost (*i.e.* of value 1) by cycling between the vertices (a)-(b), (f)-(e) and (c)-(d). This solution with a cost of 6 is thus a minimum cost assignment (Figure 2.a).

So we have $z_{np} = 12$, $z_{oa} = 10$ and $z_{ap} = 6$. Let's now

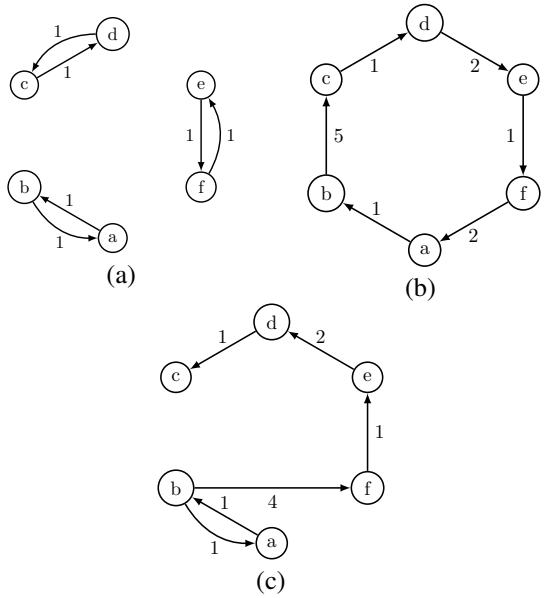

Figure 2: Solutions of the relaxations when applied to G_1.

turn our attention to G_2 satisfying $z_{np} < z_{oa} < z_{ap}$.

- The sequence (a)-(b)-(e)-(d)-(b)-(a) is a feasible solution of the n-path using only the minimum costs (*i.e.* of value 1) so that $z_{np} = 5$ (Figure 3.b).
- The minimum 1-arborescence has a cost of 6 (Figure 3.c). Note that an arc of cost 2 is mandatory to reach (*i.e* to span) vertex (c). So the feasible solution shown in the figure is optimal (since it is using only arcs of minimum cost 1 and one arc of cost 2) and $z_{oa} = 6$.
- The minimum assignment has a cost of 8. Suppose (ab) is in the assignment. This causes the suppression of arc (cb), leaving vertex (c) with a single match so that (cd) must be in the assignment too. Similarly (ea) is forced into the assignment. At this stage two perfect matchings

Instances	Assignment+LP filtering		Assignment+exact filtering			Held and Karp			ln-path		
	time	search	gap z_{ap}	time	search	gap z_{hk}	time	search	gap z_{lnp}	time	search
br17.atsp	199,3	5716951	100	17,6	21986	0	0,5	12	28,21	125,1	43113
ftv33.atsp	5,7	33936	7,85	0,1	15	0	0,2	4	4,82	2,3	27
ftv35.atsp	2,4	14613	6,25	0,2	99	1,09	0,4	14	4,28	3,3	114
ftv38.atsp	9,8	48260	6,01	0,3	121	1,05	0,5	19	3,27	6,8	162
ftv44.atsp	14,5	63362	5,7	0,4	178	1,86	2,2	108	2,79	13,2	224
ftv47.atsp	>1800	9094310	6,98	4,3	1612	1,69	4,1	181	3,15	158,5	5700
ry48p.atsp	>1800	3905158	13,21	73,2	15650	1,05	4,5	47	4,18	406	1859
ft53.atsp	>1800	7225825	14,11	2,2	248	0,17	4,7	7	11,66	>7200	515
ftv55.atsp	>1800	5502314	10,76	25,5	7307	1,55	14,6	507	6,59	>7200	125773
ftv64.atsp	>1800	4590304	6,42	10,5	2012	1,96	9,2	172	3,92	3066,1	32699
ft70.atsp	>1800	3761251	1,8	3,4	130	0,05	16,6	18	0,84	61,4	311
ftv70.atsp	>1800	3798786	9,44	69,8	8870	2,26	48,3	599	4,77	>7200	40428
kro124p.atsp	>1800	1418885	6,22	>7200	356027	0,67	227,5	1370	3,41	>7200	5644
mean	46,3	1175424,4	15,0	17,3	4852,3	1,0	25,7	235,2	6,3	426,9	9356,6
median	9,8	48260,0	7,0	3,8	930,0	1,1	4,5	47,0	4,2	61,4	311,0

Table 1: Results on some asymmetric TSP of TSPLIB

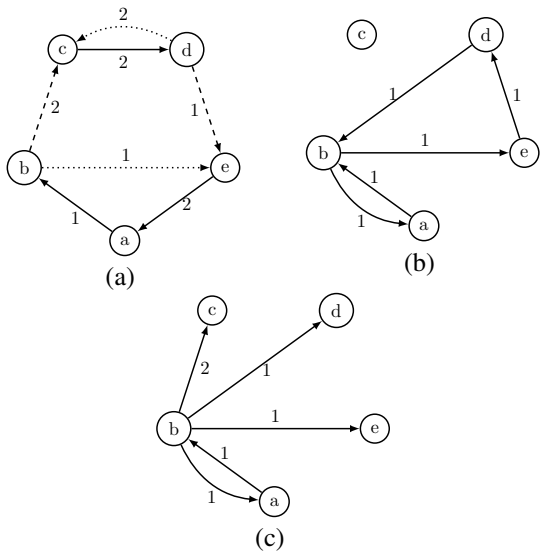

Figure 3: Solution of the relaxations when applied to G_2

are possible, one using (be) and (dc) whereas the other would use the arcs (bc) and (de) (Figure 3.a). Both have the same cost of 8, so that z_{ap} = 8. The same reasoning holds when taking (ae) initially instead of (ab).

Summerizing, we have $z_{np} = 5$, $z_{oa} = 6$ and $z_{ap} = 8$. □

Held and Karp and ln-path. The values of the Held and Karp's and ln-path relaxations are denoted z_{hk} and z_{lnp}. These two methods use Lagrangian relaxation.

Proposition 2 *z_{hk} and z_{lnp} are incomparable.*

Proof: Let's first show that we can have $z_{hk} < z_{lnp}$ by considering the graph G_3 of figure 4. In the graph G_3 presented in (Benoit and Boyd 2008), we know that the Held and Karp's bound has an integrality gap of $10/9$. In other words $z_{hk} = 9$ whereas the value of the optimal tour is 10 ($z^* = 10$). Consider now that all paths of 6 arcs, without xyx cycle, and starting and ending in a in this graph have a cost of 10. Any feasible solution of the n-path relaxation has value 10 so that $z_{hk} < z_{np}$ and therefore $z_{hk} < z_{lnp}$.

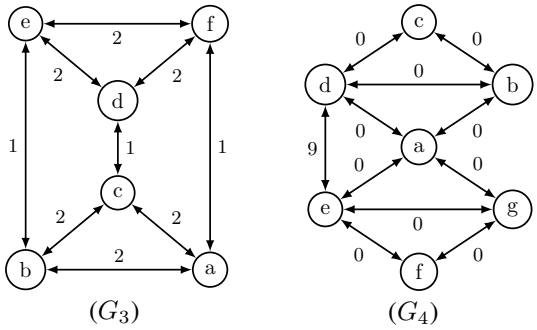

Figure 4: Graph G_3 satisfying $z_{lnp} > z_{hk}$; Graph G_4 satisfying $z_{lnp} < z_{hk}$

To show that $z_{hk} > z_{lnp}$, we consider the graph G_4 of figure 4. Exact integrality gaps for Held and Karp's bound are studied in (Benoit and Boyd 2008) and it is known that the bound is at least $8/9$ of the optimal tour for graphs of 7 vertices. Thus $z_{hk} \geq \frac{8}{9}z^*$. An optimal tour in this graph has a cost of 9 since (de) is mandatory in any hamiltonian tour (for instance: (a)-(b)-(c)-(d)-(e)-(f)-(g)-(a)). Therefore we know that $z_{hk} \geq 8$. Furthermore, the Lagrangian dual of the ln-path contains the following two constraints related to two feasible solutions of the n-path relaxation (see (16)):

$$z_{lnp} = max \;\; w$$
$$w \leq -2\lambda_b - 2\lambda_c - 2\lambda_d + 2\lambda_e + 2\lambda_f + 2\lambda_g \qquad (19)$$
$$w \leq \;\; 2\lambda_b + 2\lambda_c + 2\lambda_d - 2\lambda_e - 2\lambda_f - 2\lambda_g \qquad (20)$$

The constraint (19) corresponds to the feasible path (a)-(b)-(c)-(d)-(b)-(c)-(d)-(a) and constraint (20) for the path (a)-(e)-(f)-(g)-(e)-(f)-(g)-(a). By summing (19) and (20), it is easy to see that $z_{lnp} \leq 0$. We can conclude that $z_{lnp} < z_{hk}$. □

Note that G_3 provides an example where ln-path is the optimal tour. Moreover G_4 shows a case for which the 1-arborescence *i.e.* Held and Karp relaxation is optimal but the ln-path remains uninformative (value 0) despite the use

		z_{hk}		z_{ap}		z_{lnp}	
		mean	med	mean	med	mean	med
P	time	0,36	0,20	**0,15**	**0,10**	0,33	0,31
	gap	**4,71**	**3,89**	12,43	13,02	5,06	4,11
	$\|A\|$	**488**	**316**	624	557	527	445
	$\|P\|$	577	525	611	531	**574**	**519**
A	time	3,13	0,33	**1,07**	**0,18**	3,18	0,33
	gap	0,76	0,43	0,84	0,51	**0,50**	**0,29**
	$\|A\|$	1151	437	1132	404	**1116**	**364**
	$\|P\|$	914	388	900	383	**892**	**364**
D	time	1,13	0,81	**0,66**	**0,47**	1,07	0,58
	gap	3,08	2,00	8,43	6,99	**1,85**	**0,72**
	$\|A\|$	764	483	821	551	**624**	**258**
	$\|P\|$	614	423	632	437	**529**	**295**
G	time	1,55	1,40	**0,71**	**0,63**	2,32	1,92
	gap	9,43	9,10	21,19	21,53	**7,40**	**7,27**
	$\|A\|$	1817	1651	1999	1917	**1751**	**1440**
	$\|P\|$	1528	1467	1538	1467	**1516**	**1452**
O	time	10,02	8,93	**4,63**	**4,61**	45,54	34,35
	gap	8,35	7,95	20,21	20,48	**7,29**	**7,03**
	$\|A\|$	**9668**	**9248**	10171	9251	9945	9251
	$\|P\|$	7369	6951	7370	6951	7369	6951

Table 2: Comparing the quality of the lower bounds and the size of the domains after propagation at the root node.

		HK		AP		LNP	
		mean	med	mean	med	mean	med
P	time	191,0	**4,4**	193,9	8,1	**136,9**	4,7
	search	3k	97	28k	422	**809,3**	**45**
	#	23/27		23/27		22/27	
A	time	17,3	0,6	**1,0**	**0,2**	1,9	0,7
	search	718,3	11	**5,6**	**4**	11,7	10
	#	29/50		35/50		35/50	
D w20	time	1,0	0,4	0,6	0,4	**0,5**	**0,4**
	search	11,3	2,5	3,2	2,5	**2,8**	**2,5**
	#	20/20		20/20		20/20	
D w40	time	35,6	1,2	20,8	0,9	**1,0**	**0,2**
	search	1k	5	3k	4	**4,7**	**3**
	#	20/20		19/20		20/20	
D w60	time	17,2	2,6	59,7	1,2	**4,7**	**0,6**
	search	506,2	66	8k	5	**225,8**	**3**
	#	18/20		17/20		18/20	
D w80	time	13,1	1,0	665,6	3,8	**2,1**	**0,5**
	search	350,1	4	135k	355	**15,7**	**3**
	#	18/20		12/20		18/20	

Table 3: Exact resolution for some TSPTW instances. In each class, mean and median are computed on instances solved by all approaches.

of multipliers. Finally we outline that G_1, G_2, G_3 and G_4 have symmetric costs so that the results hold in the more general symmetric case.

Experimental analysis

The experiments ran as a single thread on a Dual Quad Core Xeon CPU, 2.66GHz with 12MB of L2 cache per processor and 16GB of RAM overall, running Linux 2.6.25 x64. The CP solver was Choco 2.1.5. Results are reported on academic benchmarks for TSP and TSPTW. The TSP dataset is taken from TSPLIB. The TSPTW dataset[2] is made of 50 asymmetric real instances (A) from (Ascheuer 1995), 27 symmetric instances (P) from (Pesant et al. 1998), 135 symmetric instances (D) from (Dumas et al. 1995), 130 symmetric instances (G) from (Gendreau et al. 1998), 25 symmetric instances (O) from (Ohlmann and Thomas 2007). The labels **time**, **search**, **gap** and **#** denote the cpu times in seconds, the number of nodes of the search tree, the gap to the best known solution computed as $100\frac{bestknown-bound}{bestknown}$ **at the root node of the search tree** and the number of instances solved (proof of optimality completed). Each run has a 2h time limit. The best known upper bound is enforced at the root node ($z \leq bestknown$) similarly to (Benchimol et al. 2012). Furthermore for Held and Karp and ln-path relaxation approaches, the subgradient process used is similar to (Benchimol et al. 2012). A limit on the number of iterations is used and the multipliers are restored upon backtracking to give a starting point to the subgradient. For the resolution we use the search strategy presented in (Pesant et al. 1998).

TSP: Table 1 reports the results obtained on asymmteric instances from TSPLIB. We compare four approaches using a CP model with the weighted circuit constraint propagated by either the assignment, the Held and Karp and the ln-path relaxation. We also report the results, quoted from (Benchimol et al. 2012), obtained with the assignment bound using linear programming reduced costs (first column). The best approach regarding the number of instances solved and quality of the bound is the Held and Karp's filtering. We have $z_{hk} < z_{lnp} < z_{ap}$ on this dataset in practice. We also note that the use of exact reduced costs for the assignment decreases the search space by an order of magnitude compared to the use of the linear programming ones. Finally, z_{lnp} is better than z_{ap} on this benchmark, but the ln-path is too slow to be competitive at this stage.

TSPTW: Table 2 reports the quality of the three bounds at the root node of the CP model *i.e* after the fix point of the propagation has been reached. The resulting graphs are typically sparser than the complete graphs considered for TSP due to the filtering of the time windows. $|A|$ denotes the number of arcs *i.e* $|A| = \sum_{i \in N} |D(next_i)|$ and $|P|$ gives the number of possible remaining positions *i.e* $|P| = \sum_{i \in N} |D(pos_i)|$. Interestingly z_{lnp} is better than z_{hk} on 4 benchmarks out of 5 and sometimes significantly (see for instance the 0,72% median gap versus 2% on the 130 Dumas' instances). As a result, the filtering is stronger. On the Pesant's benchmark where z_{hk} remains the best bound, the filtering on the position variables remains better using ln-path since positions are not filtered using the directed Held and Karp's relaxation. We also note that ln-path can be competitive in cpu time when the graph is sparse enough even for instances with up to a 100 vertices (D and G). This is however not the case for Olhmann's instances with 150/200 vertices.

Table 3 shows the results of the exact resolution for the Pesant, Ascheuer as well as Dumas instances up to 80

[2]http://iridia.ulb.ac.be/ manuel/tsptw-instances

clients. The results for the Dumas instances are reported according to the size of the time windows: 20, 40, 60 and 80 (20 instances per class). The filtering based on ln-path can speed up significantly the resolution for these medium size instances with tight time windows.

Conclusion

We proposed new filtering algorithms for the weighted circuit constraint to provide stronger filtering when the graph is sparse and directed. We revisited an old lower bound for the TSP based on a path relaxation initially introduced by (Christofides, Mingozzi, and Toth 1981). To our knowledge, this relaxation has never been really used for solving TSP as it is believed to be dominated by the Held and Karp's approach based on spanning trees. We show however that this bound can improve over the classical Held and Karp's approach when additional information, such as the possible positions of the clients in the tour, is available. This is particularly suited for problems with side constraints such as time windows. The Lagrangian subproblem of the ln-path is computationnaly heavy as it can reach a $O(n^3)$ complexity when the graph is complete. We expect significant improvements with the use of bundle methods(Lemaréchal 2001) for solving the Lagrangian dual as it will reduce the number of iterations of the sub-gradient procedure. This will also certainly lead to an improvement of the bound itself as convergence is difficult for the subgradient approach when facing problems with 100/200 vertices. Furthermore we proved that the filtering algorithms are incomparable. Combination of these algorithms is an interesting direction for future works.

References

Ascheuer, N. 1995. *Hamiltonian path problems in the online optimization of flexible manufacturing systems*. Ph.D. Dissertation, Technische Universität Berlin.

Baldacci, R.; Mingozzi, A.; and Roberti, R. 2012. New state-space relaxations for solving the traveling salesman problem with time windows. *INFORMS Journal on Computing* 24(3):356–371.

Benchimol, P.; Van Hoeve, W.-J.; Régin, J.-C.; Rousseau, L.-M.; and Rueher, M. 2012. Improved filtering for weighted circuit constraints. *Constraints* 17(3):205–233.

Benoit, G., and Boyd, S. 2008. Finding the exact integrality gap for small traveling salesman problems. *Mathematics of Operations Research* 33(4):921–931.

Christofides, N.; Mingozzi, A.; and Toth, P. 1981. State-space relaxation procedures for the computation of bounds to routing problems. *Networks* 11(2):145–164.

Dumas, Y.; Desrosiers, J.; Gelinas, E.; and Solomon, M. M. 1995. An optimal algorithm for the traveling salesman problem with time windows. *Operations Research* 43(2):367–371.

Focacci, F.; Lodi, A.; and Milano, M. 2002. A hybrid exact algorithm for the TSPTW. *INFORMS Journal on Computing* 14(4):403–417.

Gendreau, M.; Hertz, A.; Laporte, G.; and Stan, M. 1998. A generalized insertion heuristic for the traveling salesman problem with time windows. *Operations Research* 46(3):330–335.

Held, M., and Karp, R. M. 1970. The traveling-salesman problem and minimum spanning trees. *Operations Research* 18(6):1138–1162.

Held, M., and Karp, R. M. 1971. The traveling-salesman problem and minimum spanning trees: Part II. *Mathematical Programming* 1(1):6–25.

Jonker, R., and Volgenant, T. 1983. Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters* 2(4):161–163.

Kuhn, H. W. 2010. The hungarian method for the assignment problem. In *50 Years of Integer Programming 1958-2008*. Springer. 29–47.

Langevin, A.; Desrochers, M.; Desrosiers, J.; Gélinas, S.; and Soumis, F. 1993. A two-commodity flow formulation for the traveling salesman and the makespan problems with time windows. *Networks* 23(7):631–640.

Lemaréchal, C. 2001. Lagrangian relaxation. In *Computational combinatorial optimization*. Springer. 112–156.

Ohlmann, J. W., and Thomas, B. W. 2007. A compressed-annealing heuristic for the traveling salesman problem with time windows. *INFORMS Journal on Computing* 19(1):80–90.

Pesant, G.; Gendreau, M.; Potvin, J.-Y.; and Rousseau, J.-M. 1998. An exact constraint logic programming algorithm for the traveling salesman problem with time windows. *Transportation Science* 32(1):12–29.

Régin, J.-C. 2002. Cost-based arc consistency for global cardinality constraints. *Constraints* 7(3-4):387–405.

Van Hentenryck, P., and Carillon, J.-P. 1988. Generality versus specificity: An experience with AI and OR techniques. In *Proceedings of the Seventh National Conference on Artificial Intelligence – AAAI*, 660–664.