The Directed Minimum Latency Problem

Viswanath Nagarajan
Carnegie Mellon University

R. Ravi
Carnegie Mellon University, ravi@cmu.edu

Follow this and additional works at: http://repository.cmu.edu/tepper
Part of the Economic Policy Commons, and the Industrial Organization Commons

Published In
A. Goel et al. (Eds.): APPROX and RANDOM 2008, LNCS 5171, 193-206.

This Conference Proceeding is brought to you for free and open access by Research Showcase @ CMU. It has been accepted for inclusion in Tepper School of Business by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.
The Directed Minimum Latency Problem

Viswanath Nagarajan\*       R. Ravi\*

Abstract

We study the directed minimum latency problem: given an \(n\)-vertex asymmetric metric \((V, d)\) with a root vertex \(r \in V\), find a spanning path originating at \(r\) that minimizes the sum of latencies at all vertices (the latency of any vertex \(v \in V\) is the distance from \(r\) to \(v\) along the path). This problem has been well-studied on symmetric metrics, and the best known approximation guarantee is 3.59 [3]. For any \(\frac{1}{\log n} < \epsilon < 1\), we give an \(n^{O(1/\epsilon)}\) time algorithm for directed latency that achieves an approximation ratio of \(O(\rho \cdot \frac{n}{\epsilon})\), where \(\rho\) is the integrality gap of an LP relaxation for the asymmetric traveling salesman path problem [13, 5]. We prove an upper bound \(\rho = O(\sqrt{n})\), which implies (for any fixed \(\epsilon > 0\)) a polynomial time \(O(n^{1/2+\epsilon})\)-approximation algorithm for directed latency.

In the special case of metrics induced by shortest-paths in an unweighted directed graph, we give an \(O(\log^2 n)\) approximation algorithm. As a consequence, we also obtain an \(O(\log^2 n)\) approximation algorithm for minimizing the weighted completion time in no-wait permutation flowshop scheduling. We note that even in unweighted directed graphs, the directed latency problem is at least as hard to approximate as the well-studied asymmetric traveling salesman problem, for which the best known approximation guarantee is \(O(\log n)\).

1 Introduction

The minimum latency problem [17, 6, 14, 2] is a variant of the basic traveling salesman problem, where there is a metric with a specified root vertex \(r\), and the goal is to find a spanning path starting from \(r\) that minimizes the sum of arrival times at all vertices (it is also known as the deliveryman problem or traveling repairman problem). This problem can model the traveling salesman problem, and hence is NP-complete. To the best of our knowledge, all previous work has focused on symmetric metrics—the first constant-factor approximation algorithm was in Blum et al. [2], and the currently best known approximation ratio is 3.59 due to Chaudhuri et al. [3]. In this paper, we consider the minimum latency problem on asymmetric metrics.

Network design problems on directed graphs are often much harder to approximate than their undirected counterparts—the traveling salesman and Steiner tree problems are well known examples. The currently best known approximation ratio for the asymmetric traveling salesman problem (ATSP) is \(O(\log n)\) [9, 7], and improving this bound is an important open question. On the other hand, there is a 1.5-approximation algorithm for the symmetric TSP.

The orienteering problem is closely related to the minimum latency problem that we consider—given a metric with a length bound, the goal is to find a bounded-length path between two specified vertices that visits the maximum number of vertices. Blum et al. [1] gave the first constant factor approximation for the undirected version of this problem. Recently, Chekuri et al. [4] and the

\*Tepper School of Business, Carnegie Mellon University, Pittsburgh USA. Supported by NSF grant CCF-0728841.
When dealing with asymmetric metrics, the edge set $E$ is assumed to be
otherwise. Given an asymmetric metric and a special vertex $r$ as the ATSP (tour-version, where
that visits all the vertices. It is easy to see that this problem is at least as hard to approximate
$V, d$ evaluated to the directed latency problem. In ATSP-path, we are given a directed metric ($V, d$
Asymmetric Traveling Salesman Path (ATSP-path).

Theorem 1 The approximability of the path-version and tour-version of directed latency are within
a factor 4 of each other.

In this paper, we work with the path version of directed latency.

For a directed graph $G = (V, E)$ and any $S \subseteq V$, we denote by $\delta^+(S) = \{(u, v) \in E \mid u \in S, v \notin S\}$ the arcs leaving set $S$, and $\delta^-(S) = \{(u, v) \in E \mid u \notin S, v \in S\}$ the arcs entering set $S$. When dealing with asymmetric metrics, the edge set $E$ is assumed to be $V \times V$ unless mentioned otherwise. Given an asymmetric metric and a special vertex $r$, an $r$-path (resp. $r$-tour) is any directed path (resp. tour) originating at $r$.

Asymmetric Traveling Salesman Path (ATSP-path). The following problem is closely related to the directed latency problem. In ATSP-path, we are given a directed metric $(V, d)$ and specified start and end vertices $s, t \in V$. The goal is to compute the minimum length $s - t$ path that visits all the vertices. It is easy to see that this problem is at least as hard to approximate as the ATSP (tour-version, where $s = t$). Lam and Newmann [13] were the first to consider this problem, and they gave an $O(\sqrt{n})$ approximation based on the Frieze et al. [9] algorithm for ATSP. This was improved to $O(\log n)$ in Chekuri and Pal [5], which extended the algorithm of Kleinberg and Williamson [12] for ATSP. Subsequently Feige and Singh [7] showed that the approximability of ATSP-tour and ATSP-path are within a constant factor of each other. We are concerned with the following LP relaxation of the ATSP-path problem.

$$\min \sum_e d_e \cdot x_e$$

$$s.t.$$

$$x(\delta^+(u)) = x(\delta^-(u)) \quad \forall u \in V - \{s, t\}$$

$$x(\delta^+(s)) = x(\delta^-(t)) = 1$$

$$(ATSP - path)$$

$$x(\delta^-(s)) = x(\delta^+(t)) = 0$$

$$x(\delta^-(S)) \geq \frac{2}{3} \quad \forall \{u\} \subseteq S \subseteq V \setminus \{s\}, \forall u \in V$$

$$x_e \geq 0 \quad \forall \text{ arcs } e$$

The most natural LP relaxation for ATSP-path would have a 1 in the right-hand-side of the cut constraints, instead of $\frac{2}{3}$ as above. The above LP further relaxes the cut-constraints, and is still a valid relaxation of the problem. The precise value in the right-hand-side of the cut constraints is not important: we only require it to be some constant strictly between $\frac{1}{2}$ and 1.
1.2 Results and Paper Outline

Our main result is a reduction from the directed latency problem to the asymmetric traveling salesman path problem (ATSP-path) [13, 5], where the approximation ratio for directed latency depends on the integrality gap of an LP relaxation for ATSP-path. We give an \( n^{O(1/\epsilon)} \) time algorithm for the directed latency problem that achieves an approximation ratio of \( O(\rho \cdot n^{1/\epsilon}) \) (for any \( \frac{1}{\log n} < \epsilon < 1 \)), where \( \rho \) is the integrality gap of an LP relaxation for the ATSP-path problem. The best upper bound we obtain is \( \rho = O(\sqrt{n}) \) (Section 3); however we conjecture that \( \rho = O(\log n) \). In particular, our result implies a polynomial time \( O(n^{1/2+\epsilon}) \)-approximation algorithm (any fixed \( \epsilon > 0 \)) for directed latency. We study the LP relaxation for ATSP-path in Section 3, and present the algorithm for latency in Section 2. Our algorithm for latency first guesses a sequence of break-points (based on distances along the optimal path) and uses a linear program to obtain an assignment of vertices to segments (the portions between consecutive break-points), then it obtains local paths servicing each segment, and finally stitches these paths across all segments.

We also consider the special case of metrics given by shortest paths in an underlying unweighted directed graph, and obtain an \( O(\log^2 n) \) approximation for minimum latency in this case (Section 4). This algorithm is essentially based on using the directed orienteering algorithm [15, 4] within the framework for undirected latency [10]. On the hardness side, we observe that the directed latency problem (even in this ‘unweighted’ special case) is at least as hard to approximate as ATSP, for which the best known ratio is \( O(\log n) \).

We note that ideas from the ‘unweighted’ case, also imply an \( O(\log^2 n) \) approximation algorithm for minimizing weighted completion time in the no-wait permutation flowshop scheduling problem [20, 18]—this can be cast as the latency problem in a special directed metric. We are not aware of any previous results on this problem.

2 The Directed Latency Algorithm

For a given instance of directed latency, let \( \pi \) denote an optimal latency path, \( L = d(\pi) \) its length, and \( \text{Opt} \) its total latency. For any two vertices \( u, v \in V \), recall that \( d^\pi(u, v) \) denotes the length along path \( \pi \) from \( u \) to \( v \); note that \( d^\pi(u, v) \) is finite only if \( u \) appears before \( v \) on path \( \pi \). The algorithm first guesses the length \( L \) (within factor 2) and \( l = \lceil \frac{L}{\epsilon} \rceil \) vertices as follows: for each \( i = 1, \ldots, l \), \( v_i \) is the last vertex on \( \pi \) with \( d^\pi(r, v_i) \leq n^{i\epsilon} \frac{L}{n} \). We set \( v_0 = r \) and note that \( v_l \) is the last vertex visited by \( \pi \). Let \( F = \{v_0, v_1, \cdots, v_l\} \). Consider the linear program (MLP) in Figure 1.

Basically this LP requires one unit of flow to be sent from \( v_i \) to \( v_{i+1} \) (for all \( 0 \leq i < l - 1 \)) such that the total extent to which each vertex \( u \) is covered (over all these flows) is at least 1. In addition, the \( i \)-th flow is required to have total cost (under the length function \( d \)) at most \( n^{i+1}\epsilon \cdot \frac{L}{n} \). It is easy to see that this LP can be solved in polynomial time for any guess \( \{v_i\}_{i=1}^l \). Furthermore the number of possible guesses is \( O(n^{1/\epsilon}) \), hence we can obtain the optimal solution of (MLP) over all guesses, in \( n^{O(1/\epsilon)} \) time.

Claim 1 The minimum value of (MLP) over all possible guesses of \( \{v_i\}_{i=0}^l \) is at most \( 2n^\epsilon \cdot \text{Opt} \).

Proof: This claim is straightforward, based on the guesses from an optimal path. Recall that \( \pi \) is the optimal latency path for the given instance. One of the guesses of the vertices \( \{v_i\}_{i=0}^l \) satisfies the condition desired of them, namely each \( v_i \) (for \( i = 1, \ldots, l \)) is the last vertex on \( \pi \) with \( d^\pi(s, v_i) \leq n^{i\epsilon} \frac{L}{n} \). For each \( i = 0, \cdots, l - 1 \), define \( O_i \) to be the set of vertices that are visited between
Splitting off: A directed graph is called Eulerian if the in-degree equals the out-degree at each vertex. In our proofs, we make use of the following ‘splitting-off’ theorem for Eulerian digraphs.

**Theorem 2 (Frank [8] (Theorem 4.3) and Jackson [11])** Let $D = (U + r, A)$ be an Eulerian directed multi-graph. For each arc $f = (r, v) \in A$ there exists an arc $e = (u, r) \in A$ so that after replacing arcs $e$ and $f$ by arc $(u, v)$, the directed connectivity between every pair of vertices in $U$ is preserved.

Note that any vector $\tilde{x}$ of rational edge-capacities that is Eulerian (namely $\tilde{x}(\delta^-(v)) = \tilde{x}(\delta^+(v))$ at all vertices $v$) corresponds to an Eulerian multi-graph by means of a (sufficiently large) uniform scaling of all arcs. Based on this correspondence, one can use the above splitting-off theorem directly on fractional edge-capacities that are Eulerian.
2.1 Servicing vertices $V_1$

We partition $V_1$ into $l$ parts as follows: $U_i$ (for $i = 0, \ldots, l - 1$) consists of those vertices of $V_1$ that are well-covered by $z^i$ but not well-covered by any flow $z^j$ for $j > i$. Each set $U_i$ is serviced separately by means of a suitable ATSP solution on $U_i \cup \{v_i\}$ (see Lemma 1): this step requires a bound on the length of back-arcs from $U_i$-vertices to $v_i$, which is ensured by the next claim.

Claim 2 For each vertex $w \in U_i$, $d(w, v_i) \leq 8l \cdot n^i L_z n$. 

Proof: Let $j \leq i - 1$ be such that $y^i_w \geq \frac{1}{2^n}$; such an index exists by the definition of $V_1$ and $U_i$. In other words, arc-capacities $z^j$ support at least $\frac{1}{2^n}$ flow from $w$ to $v_{j+1}$; so $4l \cdot z^j$ supports a unit flow from $w$ to $v_{j+1}$. Thus $d(w, v_{j+1}) \leq 4l(d(z^j)) \leq 4l \cdot n^{(j+1)i} L_z n$. Note that for any $0 \leq k \leq l$, $z^k$ supports a unit flow from $v_k$ to $v_{k+1}$; hence $d(v_k, v_{k+1}) \leq d \cdot z^k \leq n^{(k+1)i} L_z n$. Now, $d(w, v_i) \leq d(w, v_{j+1}) + \sum_{k=j+1}^{i-1} d(v_k, v_{k+1}) \leq 4l \frac{L_z n}{n} \sum_{k=j}^{i-1} n^{(k+1)i} \leq 8l \cdot n^{i} L_z n$.

We now show how all vertices in $U_i$ can be covered by a $v_i$-tour.

Lemma 1 For each $i = 0, \ldots, l - 1$, there is a poly-time computable $v_i$-tour covering vertices $U_i$, of length $O\left(\frac{1}{n^i} n^{(i+1)i} \log n \cdot \frac{L_z}{n}\right)$.

Proof: Fix an $i \in \{0, \ldots, l - 1\}$; note that the arc capacities $z^i$ are Eulerian at all vertices except $v_i$ and $v_{i+1}$. Although applying splitting-off (Theorem 2) requires an Eulerian graph, we can apply it to $z^i$ after adding a dummy $(v_{i+1}, v_i)$ arc of capacity 1, and observing that flows from $v_i$ or flows into $v_{i+1}$ do not use the dummy arc. So using Theorem 2 on vertices $V \setminus (U_i \cup \{v_i, v_{i+1}\})$ and triangle inequality, we obtain arc capacities $\alpha$ on the arcs induced by $U_i \cup \{v_i, v_{i+1}\}$ such that: $\;d \cdot \alpha \leq d \cdot z^i \leq n^{(i+1)i} \cdot \frac{L_z n}{n}$ and $\alpha$ supports $\frac{y^i_u}{n} \geq \frac{1}{2^n} \frac{L_z n}{n}$ from $v_i$ to $u$ and from $u$ to $v_{i+1}$, for every $u \in U_i$. Below we use $B$ to denote the quantity $n^{(i+1)i} \cdot \frac{L_z n}{n}$. Consider adding a dummy arc from $v_{i+1}$ to $v_i$ of length $B$ in the modified metric on $U_i \cup \{v_i, v_{i+1}\}$, and set the arc capacity $\alpha(v_{i+1}, v_i)$ on this arc to be 1. Note that $\alpha$ is Eulerian, has total cost at most $2B$, and every non-trivial cut has value at least $min\{y^i_u : u \in U_i\} \geq \frac{1}{2^n}$. So scaling $\alpha$ uniformly by $4l$, we obtain a fractional feasible solution to ATSP on the vertices $U_i \cup \{v_i, v_{i+1}\}$ (in the modified metric), having cost at most $8l \cdot B$. Since the Frieze et al. [9] algorithm computes an integral tour of length at most $O(\log n)$ times any fractional feasible solution (see Williamson [19]), we obtain a $v_i$-tour $\tau$ on the modified metric of length at most $O(l \log n) \cdot B$. Since the dummy $(v_{i+1}, v_i)$ arc has length $B$, it may be used at most $O(l \log n)$ times in $\tau$. So removing all occurrences of this dummy arc gives a set of $O(l \log n)$ $v_i - v_{i+1}$ paths in the original metric, that together cover $U_i$. Ignoring vertex $v_{i+1}$ and inserting the direct arc to $v_i$ from the last $U_i$ vertex in each of these paths gives us the desired $v_i$-tour covering $U_i$. Finally note that each of the arcs to $v_i$ inserted above has length $O(l \cdot n^i \cdot \frac{L_z n}{n} = O(l) \cdot B$ (from Claim 2), and the number of arcs inserted is $O(l \log n)$. So the length of this $v_i$-tour is at most $O(l \log n) \cdot B + O(l^2 \log n) \cdot B = O\left(\frac{1}{n} n^{(i+1)i} \log n \cdot \frac{L_z n}{n}\right)$.

2.2 Servicing vertices $V_2$

We partition vertices in $V_2$ into $W_0, \ldots, W_{l-1}$, where each $W_i$ contains the vertices in $V_2$ that are well-covered by $z^i$. As noted earlier, each vertex $u \in W_i$ in fact has $y^i_u \geq \frac{1}{2^n} > \frac{3}{4}$. We now consider any particular $W_i$ and obtain a $v_i - v_{i+1}$ path covering the vertices of $W_i$. Vertices in $W_i$ are covered by a fractional $v_i - v_{i+1}$ path as follows. Splitting off vertices $V \setminus (W_i \cup \{v_i, v_{i+1}\})$ in the fractional solution $z^i$ gives us edge capacities $\beta$ in the metric induced on $W_i \cup \{v_i, v_{i+1}\}$, such that:
\( \beta \) supports at least \( \frac{3}{4} \) flow from \( v_i \) to \( u \) and from \( u \) to \( v_{i+1} \) for all \( u \in W_i \), and \( d \cdot \beta \leq d \cdot z_i \) (this is similar to how arc-capacities \( \alpha \) were obtained in Lemma 2.1). Note that \( \beta \) is a fractional feasible solution to the LP relaxation (ATSP – path) for the ATSP-path instance on the metric induced by \( W_i \cup \{v_i, v_{i+1}\} \) with start-vertex \( v_i \) and end-vertex \( v_{i+1} \). So if \( \rho \) denotes the (constructive) integrality gap of (ATSP – LP), we can obtain an integral \( v_i-v_{i+1} \) path that spans \( W_i \) of length at most \( \rho (d \cdot \beta) \leq \rho (d \cdot z_i) \leq \rho n^{(i+1)\epsilon} \frac{L}{n} \). This requires a polynomial time algorithm that computes an integral path of length at most \( \rho \) times the LP value; However even a non-constructive proof of integrality gap \( \rho' \) implies a constructive integrality gap \( \rho = O(\rho' \log n) \), using the algorithm in Chekuri and Pal \[5\]. So we obtain:

**Lemma 2** For each \( i = 0, \cdots, l - 1 \), there is a poly-time computable \( v_i-v_{i+1} \) path covering \( W_i \) of length at most \( \rho \cdot n^{(i+1)\epsilon} \frac{L}{n} \).

### 2.3 Stitching the local paths

We now stitch the \( v_i \)-tours that service \( V_1 \) (Lemma 1) and the \( v_i - v_{i+1} \) paths that service \( V_2 \) (Lemma 2), to obtain a single \( r \)-path that covers all vertices. For each \( i = 0, \cdots, l - 1 \), let \( \pi_i \) denote the \( v_i \)-tour servicing \( U_i \), and let \( \tau_i \) denote the \( v_i - v_{i+1} \) path servicing \( W_i \). The final \( r \)-path that the algorithm outputs is the concatenation \( \tau^* = \pi_0 \cdot \pi_1 \cdots \pi_{l-1} \cdot \tau_{l-1} \). From Lemmas 1 and 2, it follows that for all \( 0 \leq i \leq l - 1 \), \( d(\pi_i) \leq O(\frac{1}{\epsilon^2} \log n) \cdot n^{(i+1)\epsilon} \frac{L}{n} \) and \( d(\tau_i) \leq O(\rho) \cdot n^{(i+1)\epsilon} \frac{L}{n} \). So the length of \( \tau^* \) from \( r \) until all vertices of \( U_i \cup W_i \) are covered is at most \( O(\rho + \frac{1}{\epsilon^2} \log n) \cdot n^{(i+1)\epsilon} \frac{L}{n} \) (as \( \epsilon \geq \Omega(\frac{1}{\log n}) \)). This implies that the total latency of vertices in \( U_i \cup W_i \) along path \( \tau^* \) is at most

\[
O(\rho + \frac{1}{\epsilon^2} \log n) \cdot n^{(i+1)\epsilon} \frac{L}{n} \cdot (|W_i| + |U_i|).
\]

Moreover, the contribution of each vertex in \( U_i \) (resp., \( W_i \)) to the LP objective is at least \( \frac{1}{4} \cdot n^{(i+1)\epsilon} \frac{L}{n} \) (resp., \( \frac{3}{4} \cdot n^{(i+1)\epsilon} \frac{L}{n} \)). Thus the contribution of \( U_i \cup W_i \) to the LP objective is at least \( \frac{1}{4} \cdot n^{(i+1)\epsilon} \frac{L}{n} \cdot (|W_i| + |U_i|) \). Using the upper bound on the latency along \( \tau^* \) for \( U_i \cup W_i \), and summing over all \( i \), we obtain that the total latency along \( \tau^* \) is at most \( O(\frac{1}{\epsilon^2} \rho + \frac{1}{\epsilon^2} \log n) \) times the optimal value of (MLP). From Claim 1, it now follows that the latency of \( \tau^* \) is \( O(\frac{1}{\epsilon^2} \rho + \frac{1}{\epsilon^2} \log n)n^\epsilon \cdot \text{Opt} \).

**Theorem 3** For any \( \Omega(\frac{1}{\log n}) < \epsilon < 1 \), there is an \( O(\frac{\rho + \log n}{\epsilon^2}) \cdot n^\epsilon \)-approximation algorithm for directed latency, that runs in time \( n^{O(1/\epsilon)} \), where \( \rho \) is the integrality gap of the LP (ATSP – path). Using \( \rho = O(\sqrt{n}) \), we have a polynomial time \( O(n^{\frac{1}{2} + \epsilon}) \) approximation algorithm for any fixed \( \epsilon > 0 \).

We prove the bound \( \rho = O(\sqrt{n}) \) in the next section. A bound of \( \rho = O(\log n) \) on the integrality gap of (ATSP – path) would imply that this algorithm is a quasi-polynomial time \( O(\log^4 n) \) approximation for directed latency.

**Remark:** The (ATSP – path) rounding algorithm in Section 3 can be modified slightly to obtain (for any \( 0 < \delta < 1 \)), an \( (O(n^{d} \log n), \frac{1}{\delta^2}) \) bi-criteria approximation for ATSP-path. This implies the following generalization of Theorem 3.

**Corollary 1** For any \( \Omega(\frac{1}{\log n}) < \epsilon < 1 \) and \( 0 < \delta < 1 \), there is a \( n^{O(1/\epsilon)} \) time algorithm for directed latency, that computes \( \lfloor \frac{1}{3} \rfloor \) \( \pi \) paths covering all vertices, having a total latency of \( O(\frac{\log n}{\epsilon^2} \cdot n^{\epsilon + \delta}) \cdot \text{Opt} \), where \( \text{Opt} \) is the minimum latency of a single path covering all the vertices.
3 Bounding the integrality gap of ATSP-path

We prove an upper bound of $O(\sqrt{n})$ on the integrality gap $\rho$ of the linear relaxation ($ATSP-path$) (c.f. Section 1.1). Even for the seemingly stronger LP with 1 in the right-hand-side of the cut constraints, the best bound on the integrality gap we can obtain is $O(\sqrt{n})$: this follows from the cycle-cover based algorithm of Lam and Newmann [13]. As mentioned in Chekuri and Pal [5], it is unclear whether their $O(\log n)$-approximation can be used to bound the integrality gap of such a linear program. In this section, we present a rounding algorithm for the weaker LP ($ATSP-path$), which shows $\rho = O(\sqrt{n})$. Our algorithm is similar to the ATSP-path algorithm of Lam and Newmann [13] and the ATSP algorithm of Frieze et al. [9]; but it needs some more work as we compare the algorithm’s solution against a fractional solution to ($ATSP-path$).

Let $x$ be any feasible solution to ($ATSP-path$). We now show how $x$ can be rounded to obtain an integral path spanning all vertices, of total length $O(\sqrt{n})/(d \cdot x)$. Let $N$ denote the network corresponding to the directed metric with the cost of each arc equal to its metric length, and an extra $(t, s)$ arc of cost 0. The capacity of this extra $(t, s)$ arc is set to 3, and all other capacities are set to $\infty$. The rounding algorithm for $x$ is as follows.

1. Initialize the set of representatives $R \leftarrow V \setminus \{s, t\}$, and the current integral solution $\sigma = \emptyset$.

2. While $R \neq \emptyset$, do:
   
   (a) Compute a minimum cost circulation $C$ in $N[R \cup \{s, t\}]$ that sends at least 2 units of flow through each vertex in $R$ (note: $C$ can be expressed as a sum of cycles).
   
   (b) Repeatedly extract from $C$ all cycles that do not use the extra arc $(t, s)$, to obtain circulation $A \subseteq C$. Let $R' \subseteq R$ be the set of $R$-vertices that have degree at least 1 in $A$.
   
   (c) Let $B = C \setminus A$; note that $B$ is Eulerian and each cycle in it uses arc $(t, s)$.
   
   (d) If $|R'| \geq \sqrt{n}$, do:
      
      i. Set $\sigma \leftarrow \sigma \cup A$.
      
      ii. Modify $R$ by dropping all but one $R'$-vertex from each strong component of $A$.
   
   (e) If $|R'| < \sqrt{n}$, do:
      
      i. Take an Euler tour on $B$ and remove all (at most 3) occurrences of arc $(t, s)$ to obtain $s$-$t$ paths $P_1, P_2, P_3$.
      
      ii. Restrict each path $P_1, P_2, P_3$ to vertices in $R \setminus R'$ by short-cutting over $R'$-vertices, to obtain paths $\tilde{P}_1, \tilde{P}_2, \tilde{P}_3$.
      
      iii. Take a topological ordering $s = w_1, w_2, \ldots, w_h = t$ of vertices $(R \setminus R') \cup \{s, t\}$ relative to the arcs $\tilde{P}_1 \cup \tilde{P}_2 \cup \tilde{P}_3$.
      
      iv. Set $\sigma \leftarrow \sigma \cup \{(w_j, w_{j+1}) : 1 \leq j \leq h - 1\}$.
      
      v. Repeat for each vertex $u \in R'$: find an arc $(w, w') \in \sigma$ such that $x$ supports $\frac{1}{6}$ flow from $w$ to $u$ and from $u$ to $w'$, and modify $\sigma \leftarrow (\sigma \setminus (w, w')) \cup \{(w, u), (u, w')\}$.
      
      vi. Set $R \leftarrow \emptyset$.

3. Output any spanning $s$-$t$ walk in $\sigma$.

We now show the correctness and performance guarantee of the rounding algorithm. We first bound the cost of the circulation obtained in Step 2a during any iteration.
Claim 3 For any \( R \subseteq V \setminus \{s, t\} \), the minimum cost circulation \( C \) computed in step 2a has cost at most \( 3(d \cdot x) \).

Proof: The arc values \( x \) define a fractional \( s-t \) path in network \( N \). Extend \( x \) to be a (fractional) circulation by setting \( x(t, s) = 1 \). We can now apply splitting-off (Theorem 2) on each vertex in \( V \setminus R \), to obtain capacities \( x' \) in network \( N[R \cup \{s, t\}] \), such that every pairwise connectivity is preserved and (by triangle inequality) \( d \cdot x' \leq d \cdot x \). Note that the extra \( (t, s) \) arc is not modified in the splitting-off steps. So \( x' \) supports \( \frac{2}{3} \) flow from \( s \) to each vertex in \( R \); this implies that \( 3x' \) is a feasible fractional solution to the circulation instance solved in step 2a (note that \( x'(t, s) \) remains 1, so solution \( 3x' \) satisfies the capacity of arc \( (t, s) \)). Finally, note that the linear relaxation for circulation is integral (c.f. Nemhauser and Wolsey [16]). So the minimum cost (integral) circulation computed in step 2a has cost at most \( 3d \cdot x \).

Note that each time step 2d is executed, \( |R| \) decreases by at least \( \sqrt{n}/2 \) (each strong component in \( A \) has at least 2 vertices); so there are at most \( O(\sqrt{n}) \) such iterations and the cost of \( \sigma \) due to additions in this step is \( O(\sqrt{n}) \) (using Claim 3). Step 2e is executed at most once (at the end); the next claim shows that this step is well defined and bounds the cost incurred.

Claim 4 In step 2(e)iii, there exists a topological ordering \( w_1, \ldots, w_h \) of \( (R \setminus R') \cup \{s, t\} \) w.r.t. arcs \( P_1 \cup P_2 \cup P_3 \). Furthermore, \( \{(w_j, w_{j+1}) : 1 \leq j \leq h - 1\} \subseteq P_1 \cup P_2 \cup P_3 \).

Proof: Note that any cycle in \( P_1 \cup P_2 \cup P_3 \) is a cycle in \( B \) that does not use arc \( (t, s) \), which is not possible by the definition of \( B \) (every cycle in \( B \) uses arc \( (t, s) \)); so \( P_1 \cup P_2 \cup P_3 \) is acyclic. It is clear that if \( \tilde{P}_1 \cup \tilde{P}_2 \cup \tilde{P}_3 \) contains a cycle, so does \( P_1 \cup P_2 \cup P_3 \) (each path \( \tilde{P}_i \) is obtained by short-cutting the corresponding path \( P_i \)). Hence \( \tilde{P}_1 \cup \tilde{P}_2 \cup \tilde{P}_3 \) is also acyclic, and there is a topological ordering of \( (R \setminus R') \cup \{s, t\} \) relative to arcs \( \tilde{P}_1 \cup \tilde{P}_2 \cup \tilde{P}_3 \). We now prove the second part of the claim. In circulation \( C \), each vertex of \( R \) has at least 2 units of flow through it; but vertices \( R \setminus R' \) are not covered (even to an extent 1) in the circulation \( A \). So each vertex of \( R \setminus R' \) is covered to extent at least 2 in circulation \( B \), and hence in \( P_1 \cup P_2 \cup P_3 \). In other words, each vertex of \( R \setminus R' \) appears on at least two of the three \( s-t \) paths \( P_1, P_2, P_3 \). This also implies that (after the short-cutting) each \( R \setminus R' \) vertex appears on at least two of the three \( s-t \) paths \( \tilde{P}_1, \tilde{P}_2, \tilde{P}_3 \). Now observe that for each consecutive pair \( (w_j, w_{j+1}) \) (\( 1 \leq j \leq h - 1 \)) in the topological order, there is a common path \( \tilde{P}_k \) (for some \( k = 1, 2, 3 \)) that contains both \( w_j \) and \( w_{j+1} \). Furthermore, in \( \tilde{P}_k \), \( w_j \) and \( w_{j+1} \) are consecutive in that order (otherwise, the topological order would contain a back arc!). Thus each arc \( (w_j, w_{j+1}) \) (\( 1 \leq j \leq h - 1 \)) is present in \( \tilde{P}_1 \cup \tilde{P}_2 \cup \tilde{P}_3 \), and we obtain the claim.

We also need the following claim to bound the cost of insertions in step 2(e)v.

Claim 5 For any two vertices \( u', u'' \) in \( V \), if \( \lambda(u', u'' ; x) \) (resp. \( \lambda(u'', u' ; x) \)) denotes the maximum flow supported by \( x \) from \( u' \) to \( u'' \) (resp. \( u'' \) to \( u' \)), then \( \lambda(u', u'' ; x) + \lambda(u'', u' ; x) \geq \frac{1}{3} \).

Proof: If either \( u' \) or \( u'' \) is in \( \{s, t\} \), the claim is obvious since for every vertex \( v \), \( x \) supports \( \frac{2}{3} \) flow from \( s \) to \( v \) and from \( v \) to \( t \). Otherwise \( \{s, t, u', u''\} \) are distinct, and define capacities \( \hat{x} \) as:

\[
\hat{x}(v_1, v_2) = \begin{cases} 
  x(v_1, v_2) & \text{for arcs } (v_1, v_2) \neq (t, s) \\
  1 & \text{for arc } (v_1, v_2) = (t, s) 
\end{cases}
\]

Observe that \( \hat{x} \) is Eulerian; now apply Theorem 2 to \( \hat{x} \) and split-off all vertices of \( V \) except \( T = \{s, t, u', u''\} \), and obtain capacities \( y \) on the arcs induced on \( T \). We have \( \lambda(t_1, t_2; y) = \lambda(t_1, t_2; \hat{x}) \) for all \( t_1, t_2 \in T \). Note that since neither \( t \) nor \( s \) is split-off, their degrees in \( y \) are unchanged.
from \( \hat{x} \), and also \( y(t, s) \geq \hat{x}(t, s) = 1 \). Since the out-degree of \( t \) in \( \hat{x} \) (hence in \( y \)) is 1 and \( y_{t,s} \geq 1 \), we have \( y(t, u^{'}) = y(t, u^{''}) = 0 \) and \( y(t, s) = 1 \). The capacities \( y \) support at least \( \frac{y}{3} \) flow from \( s \) to \( u^{'} \); so \( y(s, u^{'}) + y(u^{''}, u^{'}) \geq \frac{2}{3} \). Similarly for \( u^{''} \), we have \( y(s, u^{''}) + y(u^{''}, u^{'}) \geq \frac{2}{3} \), and adding these two inequalities we get \( y(u^{'}, u^{''}) + y(u^{''}, u^{'}) + (y(s, u^{'}) + y(s, u^{''})) \geq \frac{4}{3} \). Note that \( y(s, u^{'}) + y(s, u^{''}) \leq y(\delta^{+}(s)) = \hat{x}(\delta^{+}(s)) = 1 \) (the degree of \( s \) is unchanged in the splitting-off). So \( y(u^{'}, u^{''}) + y(u^{''}, u^{'}) \geq \frac{1}{3} \). Since \( y \) is obtained from \( \hat{x} \) by a sequence of splitting-off operations, it follows that \( \hat{x} \) supports flows corresponding to all edges in \( y \) simultaneously. In particular, the following flows are supported disjointly in \( \hat{x} \): \( F_{1} \) that sends \( y(u^{'}, u^{''}) \) units from \( u^{'} \) to \( u^{''} \), \( F_{2} \) that sends \( y(u^{''}, u^{'}) \) units from \( u^{''} \) to \( u^{'} \), and \( F_{3} \) that sends \( y(t, s) = 1 \) unit from \( t \) to \( s \). Hence the flows \( F_{1} \) and \( F_{2} \) are each supported by \( \hat{x} \) and do not use the extra \((t, s)\) arc (since \( \hat{x}(\delta^{+}(t)) = \hat{x}(t, s) = 1 \). This implies that the flows \( F_{1} \) and \( F_{2} \) are both supported by the original capacities \( x \) (where \( x(t, s) = 0 \). Hence \( \lambda(u^{'}, u^{''}; x) + \lambda(u^{''}, u^{'}; x) \geq y(u^{'}, u^{''}) + y(u^{''}, u^{'}) \geq \frac{1}{3} \).

From Claim 4, we obtain that the cost addition in step 2(e)iv is at most \( d(\hat{P}_{1}) + d(\hat{P}_{2}) + d(\hat{P}_{3}) \leq d(P_{1}) + d(P_{2}) + d(P_{3}) \leq 3(d \cdot x) \) (from Claim 3). We now consider the cost addition to \( \sigma \) in step 2(e)v. Claim 5 implies that for any pair of vertices \( u^{'}, u^{''} \in V \), \( x \) supports \( \frac{1}{6} \) flow either from \( u^{'} \) to \( u^{''} \) or from \( u^{''} \) to \( u^{'} \). Also for every vertex \( u \), \( x \) supports \( \frac{y}{3} \) flow from \( s \) to \( u \) and from \( u \) to \( t \). Since \( \sigma \) always contains an \( s-t \) path in step 2(e)v, there is always some position along this \( s-t \) path to insert any vertex \( u \in R \) as required in step 2(e)v. Furthermore, the cost increase in any such insertion step is at most \( 12(d \cdot x) \). Hence the total cost for inserting all the vertices \( R \) into \( \sigma \) is at most \( 12|R|[(d \cdot x) = O(\sqrt{n})(d \cdot x)] \). Thus the total cost of \( \sigma \) at the end of the algorithm is \( O(\sqrt{n})(d \cdot x) \). Finally note that \( \sigma \) is connected (in the undirected sense), Eulerian at all vertices in \( V \setminus \{s, t\} \) and has outdegree 1 at \( s \). This implies that \( \sigma \) corresponds to a spanning \( s-t \) walk. This completes the proof of the following.

**Theorem 4** The integrality gap of (ATSP – path) is at most \( O(\sqrt{n}) \).

### 4 Unweighted Directed Metrics

In the special case where the metric is induced by shortest paths in an unweighted directed graph, we obtain an improved approximation guarantee for the minimum latency problem. This draws on ideas from the undirected latency problem, and the \( O(\log^{2} n) \) approximation ratio for directed orienteering ([15] and [4]). The directed orienteering problem is as follows: given a starting vertex \( r \) in an asymmetric metric and length bound \( L \), find an \( r \)-path of length at most \( L \) covering the maximum number of vertices. We note that the reduction from ATSP to directed latency also holds in unweighted directed metrics, and the best known approximation ratio for ATSP even on this special class is \( O(\log n) \). Here we show the following.

**Theorem 5** An \( \alpha \)-approximation algorithm for directed orienteering implies an \( O(\alpha + \gamma) \) approximation algorithm for the directed latency problem on unweighted digraphs, where \( \gamma \) is the best approximation ratio for ATSP. In particular there is an \( O(\log^{2} n) \) approximation.

Let \( G = (V, A) \) denote the underlying digraph that induces the given metric, and \( r \) the root vertex. We first argue (Section 4.1) that if \( G \) is strongly connected, then there is an \( O(\alpha) \)-approximation algorithm. Then we show (Section 4.2) how this can be extended to the case when \( G \) is not strongly connected.
4.1 $G$ is strongly connected

In this case, the distance from any vertex to the root $r$ is at most $n = |V|$. The algorithm and analysis for this case are identical to those for the undirected latency problem [2, 10, 3]. Details are given in Appendix B.

Remark: This ‘greedy’ approach does not work in the general directed case since it is unclear how to bound the length of back-arcs to the root $r$ (which is required to stitch the paths that are computed greedily). In the undirected case, back-arcs can be easily bounded by the forward length, and this approach results in a constant approximation algorithm. In the unweighted strongly-connected case (considered above), the total length of back-arcs used by the algorithm could be bounded by roughly $n^2$ (which is also a lower bound for the latency problem). By an identical analysis, it also follows that there is an $O(\alpha)$-approximation for the directed latency problem on metrics $(V, d)$ with the following property: for every vertex $v \in V$, the back-arc length to $r$ is within a constant factor of the forward-arc length from $r$, i.e. $d(v, r) \leq O(1) \cdot d(r, v)$. As a consequence, we obtain an $O(\alpha) = O(\log^2 n)$ approximation for no-wait flowshop scheduling with the weighted completion time objective ($n$ is the number of jobs in the given instance); this seems to be the first approximation ratio for the problem. The no-wait flowshop problem can be modeled as a minimum latency problem in an appropriate directed metric [20, 18], with the property that all back-arcs to the root $r$ have length 0; hence the above greedy approach applies.

4.2 $G$ is not strongly connected

In this case, we show an $O(\gamma + \beta)$-approximation algorithm, where $\gamma$ is the approximation guarantee for ATSP and $\beta$ is the approximation guarantee for the minimum latency problem on unweighted strongly-connected digraphs. From Section 4.1, $\beta = O(\alpha)$, where $\alpha$ is the approximation ratio for directed orienteering. Consider the strong components of $G$, which form a directed acyclic graph. If the instance is feasible, there is a Hamilton path in $G$ from $r$; so we can order the strong components of $G$ as $C_1, \ldots, C_l$ such that $r \in C_1$ and any spanning path from $r$ visits the strong components in that order. For each $1 \leq i \leq l$, let $n_i = |C_i|$, and pick an arbitrary vertex $s_i \in C_i$ as root for each strong component (setting $s_1 = r$).

Lemma 3 There exists a spanning $r$-path $\tau^*$ having latency objective at most $7 \cdot \text{Opt}$ such that $\tau^* = \tau_1 \cdot (s_1, s_2) \cdot \tau_2 \cdot (s_2, s_3) \cdots (s_{l-1}, s_l) \cdot \tau_l$, where each $\tau_i$ (for $1 \leq i \leq l$) is an $s_i$-tour covering all vertices in $C_i$.

Proof: Consider the optimal latency $r$-path $P^*$: this is a concatenation $P_1 \cdot P_2 \cdots P_l$ of paths in each strong component ($P_i$ is a spanning path on $C_i$). For each $1 \leq i \leq l$, let $\text{Lat}(P_i)$ denote the latency of vertices $C_i$ just along path $P_i$, and $D_i = \sum_{j=1}^{i-1} d(P_j)$ be the distance traversed by $P^*$ before $P_i$. Then the total latency along $P^*$ is $\text{Opt} = \sum_{i=1}^{l}(n_i \cdot D_i + \text{Lat}(P_i))$.

For each $1 \leq i \leq l$, let $\tau_i$ denote a spanning tour on $C_i$, obtained by adding to $P_i$ the direct arcs: from $s_i$ to its first vertex and from its last vertex to $s_i$. Each of these extra arcs in $\tau_i$ has length at most $n_i - 1$ (since $C_i$ is strongly connected), and $d(P_i) \geq n_i - 1$ (it is spanning on $C_i$); so $d(\tau_i) \leq 3d(P_i)$. Let $\text{Lat}(\tau_i)$ denote the latency of vertices $C_i$ along $\tau_i$; from the above observation we have $\text{Lat}(\tau_i) \leq n_i \cdot (n_i - 1) + \text{Lat}(P_i)$. Now we obtain $\tau^*$ as the concatenation $\tau_1 \cdot (s_1, s_2) \cdot \tau_2 \cdots (s_{l-1}, s_l) \cdot \tau_l$. Note also that for any $1 \leq i \leq l - 1$, $d(s_i, s_{i+1}) \leq n_i + n_{i+1}$. So the
The latency in $\tau^*$ of vertices $C_i$ is:

\[
\begin{align*}
n_i \cdot \sum_{j=1}^{i-1} (d(\tau_j) + d(s_j,s_{j+1})) + \text{Lat}(\tau_i) \\
\leq n_i \cdot \sum_{j=1}^{i-1} (3d(P_j) + n_j + n_{j+1}) + n_i \cdot (n_i - 1) + \text{Lat}(P_i) \\
\leq n_i \cdot \sum_{j=1}^{i-1} (3d(P_j) + 2n_j) + n_i^2 + n_i \cdot (n_i - 1) + \text{Lat}(P_i) \\
\leq n_i \cdot \sum_{j=1}^{i-1} 7d(P_j) + n_i^2 + n_i \cdot (n_i - 1) + \text{Lat}(P_i) \\
\leq 7n_i \cdot D_i + 2n_i^2 + \text{Lat}(P_i) \\
\leq 7n_i \cdot D_i + 5 \cdot \text{Lat}(P_i)
\end{align*}
\]

The last inequality follows from the fact that $\text{Lat}(P_i) \geq n_i^2/2$ ($P_i$ is a path on $n_i$ vertices in an unweighted metric). So the total latency of $\tau^*$ is at most $7 \sum_{i=1}^{l} (n_i \cdot D_i + \text{Lat}(P_i)) = 7 \cdot \text{Opt}$. The algorithm for directed latency in this case computes an approximately minimum latency $s_i$-path for each $C_i$ separately (using the algorithm in Section 4.1); by adding the direct arc from the last vertex back to $s_i$, we obtain $C_i$-spanning tours $\{\sigma_i\}_{i=1}^l$. We now use the following claim from [2] to bound the length of each tour $\sigma_i$.

**Claim 6 ([2])** Given $C_i$-spanning tours $\sigma_i$ and $\pi_i$, there exists a poly-time computable tour $\sigma'_i$ on $C_i$ of length at most $3 \cdot d(\pi_1)$ and latency at most thrice that of $\sigma_i$.

**Proof:** Tour $\sigma'_i$ is constructed as follows: starting at $s_i$, traverse tour $\sigma_i$ until a length of $d(\pi_i)$, then traverse tour $\pi_i$ from the current vertex to visit all remaining vertices and then return to $s_i$. Note that tour $\pi_i$ will have to be traversed at most twice, and so the length of $\sigma'_i$ is at most $3d(\pi_i)$. Furthermore, the total latency along $\sigma'_i$ for vertices visited in the $\sigma_i$ part is at most $\text{Lat}(\sigma_i)$ (the latency along $\sigma_i$). Also the latency along $\sigma'_i$ of each vertex $v$ visited in the $\pi_i$ part is at most $3d(\pi_i)$, which is at most thrice its latency in $\sigma_i$. Hence the total latency along $\sigma'_i$ is at most $3 \cdot \text{Lat}(\sigma_i)$.

This implies that by truncating $\sigma_i$ with a $\gamma$-approximate TSP on $C_i$, we obtain another spanning tour $\sigma'_i$ of length $3\gamma \cdot L_i$ and latency $3 \cdot \text{Lat}(\sigma_i)$ (where $L_i$ is length of the minimum TSP on $C_i$). The final $r$-path is the concatenation of these local tours, $\pi = \sigma'_1 \cdot (s_1, s_2) \cdot \sigma'_2 \cdot \cdots \cdot (s_{l-1}, s_l) \cdot \sigma'_l$.

**Claim 7** The latency of $r$-path $\pi$ is at most $O(\gamma + \beta) \cdot \text{Opt}$.

**Proof:** Consider the near-optimal $r$-path $\tau^*$ given by Lemma 3. For $1 \leq i \leq l$, let $\text{Opt}_i$ denote the latency of the $C_i$-spanning tour $\tau_i$, and $\tilde{D}_i = \sum_{j=1}^{i-1} (d(\tau_j) + d(s_j,s_{j+1}))$ denote the length of $\tau^*$ before $C_i$. Then the total latency of $\tau^*$ can be written as $\sum_{i=1}^{l} (n_i \cdot D_i + \text{Opt}_i) \leq 7 \cdot \text{Opt}$.

Now consider the $r$-path $\pi$ output by the algorithm. The $s_i$-tour $\tau_i$ is a feasible solution to the minimum latency instance on $C_i$; so the latency of tour $\sigma_i$ is at most $\beta \cdot \text{Opt}_i$, since we use a $\beta$-approximation for each such instance. So for each $1 \leq i \leq l$, the truncated tour $\sigma'_i$ has latency $\text{Lat}(\sigma'_i) \leq 3 \beta \cdot \text{Opt}_i$, and length $d(\sigma'_i) \leq 3\gamma L_i$. Again, the latency of $\pi$ can be written as $\sum_{i=1}^{l} (n_i \cdot D'_i + \text{Lat}(\sigma'_i))$, where $D'_i = \sum_{j=1}^{i-1} (d(\sigma'_j) + d(s_j,s_{j+1}))$ is the length of $\pi$ before $C_i$. So the latency of vertices $C_i$ in $\pi$ is:

\[
\begin{align*}
n_i \cdot \sum_{j=1}^{i-1} (d(\sigma'_j) + d(s_j,s_{j+1})) + \text{Lat}(\sigma'_i) \\
\leq n_i \cdot \sum_{j=1}^{i-1} (3\gamma \cdot L_j + d(s_j,s_{j+1})) + 3\beta \cdot \text{Opt}_i \\
\leq n_i \cdot \sum_{j=1}^{i-1} (3\gamma \cdot d(\tau_j) + d(s_j,s_{j+1})) + 3\beta \text{Opt}_i \\
\leq 3\gamma n_i \cdot \sum_{j=1}^{i-1} (d(\tau_j) + d(s_j,s_{j+1})) + 3\beta \text{Opt}_i \\
= 3\gamma n_i \cdot \tilde{D}_i + 3\beta \text{Opt}_i \\
\leq 3(\gamma + \beta)(n_i \cdot \tilde{D}_i + \text{Opt}_i)
\end{align*}
\]
So the total latency of $\pi$ is at most $3(\gamma + \beta) \sum_{i=1}^{l}(n_i \cdot \tilde{D}_i + \text{Opt}_i) \leq O(\gamma + \beta) \cdot \text{Opt}$. Theorem 5 now follows.

References


Recall the definitions of the two versions of directed latency on asymmetric metrics \((V, d)\) (resp. \(d^r\)).

### Proof of Theorem 1.

We now prove Theorem 1, which shows the equivalence (up to a constant factor) of approximating the path and tour versions of directed latency.

**Proof of Theorem 1.** Recall the definitions of the two versions of directed latency on asymmetric metric \((V, d)\) with root vertex \(r \in V\). _Path latency:_ find a spanning path \(\pi\) originating at \(r\) that minimizes \(\sum_{v \in V} d^T(r, v)\) (here \(d^T(r, r) = 0\)). _Tour latency:_ find a spanning tour \(\tau\) originating at \(r\) that minimizes \(\sum_{v \in V} d^T(r, v)\) (here \(d^T(r, r) = d(\tau)\)).

We first show that any approximation algorithm for path-latency implies the same guarantee for tour-latency. Modify the given metric \((V, d)\) (for tour-latency) by adding a new vertex \(r'\) with distances to vertices \(V\) as follows: \(d(v, r') = d(v, r)\) and \(d(r', v) = \infty\) for all \(v \in V\). It is clear that optimal value of tour-latency on \((V, d)\) equals the optimal value of path-latency on the modified metric, which implies the first direction.

For the reverse direction, let \(\mathcal{A}\) be any \(\alpha\)-approximation algorithm for tour-latency (we assume \(\alpha \leq n\)). Set \(\beta = \lceil 4\alpha \rceil + 1\) and define metric \((\tilde{V}, \tilde{d})\) which is obtained from \((V, d)\) by making \(\beta\) copies of each vertex in \(V \setminus r\). Let \(\tilde{p}\)Lat (resp. \(p\)Lat) denote the optimal value of path-latency on \(\tilde{d}\) (resp. \(d\)); note that \(p\)Lat = \(\beta \cdot p\)Lat. We assume (by scaling) that the smallest non-zero distance in \(\tilde{d}\) is 1, and that there is a finite length spanning \(r\)-path in metric \(d\) (otherwise the latency problem...
is trivial); so $p\text{Lat} \leq n^2 d_{\text{max}}$ where $d_{\text{max}}$ is the maximum finite distance in $d$. We further modify metric $d$ by adding dummy arcs of length $L$ (value to be set later) from each $v \in V \setminus r$ to $r$, and let $l$ denote the shortest path metric on this modified graph. Note that the optimal value of tour-latency on metric $l$ is $t\text{Lat} \leq 2 \cdot p\text{Lat} + L$. The algorithm for path-latency on $d$ is as follows.

- For each $0 \leq i \leq \lg(\beta n^2 d_{\text{max}})$ do:
  1. Set $L \leftarrow 2^i$ and $l$ to be the metric as defined above.
  2. Run algorithm $A$ for tour-latency on $l$ to get tour $\tau$. Let $\pi$ denote the portion of $\tau$ until the first usage of a dummy arc.
  3. If $\pi$ visits at least one copy of each vertex in $V \setminus r$, consider this as a feasible solution to path-latency on $d$; otherwise skip this iteration.
- Output the best path-latency solution encountered in Step 3 above.

Note that since $1 \leq p\tilde{\text{Lat}} \leq \beta n^2 d_{\text{max}}$, there is an iteration where $p\tilde{\text{Lat}} \leq 2^i \leq 2 \cdot p\tilde{\text{Lat}}$. We now argue that in this iteration $i$, the above algorithm gets a feasible solution in Step 3 with path-latency value (on metric $d$) at most $4\alpha \cdot p\text{Lat}$. The tour $\tau$ found by algorithm $A$ has latency (in metric $l$) $\text{Lat}(\tau) \leq \alpha \cdot t\text{Lat} \leq \alpha (2 \cdot p\tilde{\text{Lat}} + L) \leq 4\alpha \cdot p\tilde{\text{Lat}}$. If $x$ denotes the number of unvisited vertices of $\tilde{V}$ in $\pi$ (i.e. when a dummy arc is used for the first time in $\tau$) then $\text{Lat}(\tau) \geq L \cdot x$; hence $x \leq \text{Lat}(\tau)/L \leq 4\alpha \cdot p\text{Lat}/L \leq 4\alpha \leq \beta$. Since $\tilde{V}$ has $\beta$ copies of each vertex of $V$, $\pi$ visits at least one copy of each $V$-vertex: so Step 3 records $\pi$ as a potential solution. We modify $\tau$ so that it visits all copies of each $V$-vertex the first time some copy is visited: this only reduces the latency of $\tau$. After this modification, note that dummy arcs are used in $\tau$ only after all vertices $\tilde{V}$ are visited: in other words, $\tau$ (and also $\pi$) induces a path-latency solution on metric $(\tilde{V}, \tilde{d})$ of latency value at most $\text{Lat}(\tau) \leq 4\alpha \cdot p\text{Lat}$. Hence the corresponding solution in metric $(V, d)$ has latency at most $1/4 \cdot 4\alpha \cdot p\text{Lat} = 4\alpha \cdot p\text{Lat}$. Thus the best solution to path-latency on $d$ obtained by the above algorithm has value at most $4\alpha \cdot p\text{Lat}$, and it is a $4\alpha$ approximation algorithm.

### B Directed latency on unweighted strongly connected graphs

The algorithm for the unweighted strongly connected case is as follows.

1. Set current length bound $L = n$, and current tour $\tau = \phi$.

2. While not all vertices covered, do:
   
   (a) Run the directed orienteering algorithm $\alpha$ times, each time obtaining a path of length $\leq L$ from $r$ that covers (approximately) the maximum number of remaining vertices.
   
   (b) Stitch the $\alpha$ paths into an $r$-tour by adding arcs from the end of each path back to $r$, and append this tour to the end of $\tau$.
   
   (c) Set length bound $L \leftarrow 2 \cdot L$.

3. Output tour $\tau$.

In the optimal latency $r$-path, let $N_i$ (for $i = 0, 1, \cdots$) be the number of vertices having latency at most $2^i n$. Then one can verify that the optimal latency $\text{Opt} \geq \frac{1}{2} \sum_{i \geq 0} 2^{-i-1} n \cdot (n - N_i)$. In the
r-path returned by the algorithm, let $R_i$ (for $i \geq 0$) denote the number of vertices left uncovered at the end of iteration $i$ (where length bound is $2^i n$). Note that each back arc to $r$ has length at most $n$; so the increase in the length of the r-path in iteration $i$ is at most $\alpha \cdot (2^i n + n) \leq 2^{i+1} \alpha n$.

**Claim 8** For all $i \geq 1$, either $R_i \leq n - N_i$ or $R_i \leq \frac{1}{\epsilon} R_{i-1} + (1 - \frac{1}{\epsilon})(n - N_i)$. Hence for all $i \geq 1$, $R_i \leq \max\{n - N_i, \frac{1}{\epsilon} R_{i-1} + (1 - \frac{1}{\epsilon})(n - N_i)\} \leq \frac{1}{\epsilon} R_{i-1} + (2 - \frac{1}{\epsilon})(n - N_i)$.

**Proof:** If $R_{i-1} \leq n - N_i$, then $R_i \leq R_{i-1} \leq n - N_i$. Suppose $R_{i-1} > n - N_i$; then there is an $r$-path (of length $2^i n$) in iteration $i$ containing at least $N_i - (n - R_{i-1})$ new vertices (the optimal path restricted to uncovered vertices & this length bound). Since we use an $\alpha$-approximation for directed orienteering in step 2a (repeatedly $\alpha$ times), the resulting set of $r$-paths cover at least $(1 - 1/e)(R_{i-1} - n + N_i)$ new vertices- this follows from a standard maximum-coverage argument. In other words, $R_i \leq R_{i-1} - (1 - 1/e)(R_{i-1} - n + N_i) = \frac{1}{\epsilon} R_{i-1} + (1 - \frac{1}{\epsilon})(n - N_i)$.

If $m_0$ is the number of vertices covered in the first iteration (when length bound is $n$), we can upper bound the latency $\text{Alg}$ of the $r$-tour $\tau$ by $m_0 \cdot 2\alpha n + \sum_{i \geq 0} 2^{i+2} \alpha n \cdot R_i$; recall that the increase in length of $\tau$ in iteration $i$ is at most $2^{i+1} \alpha n$. Since $m_0 \leq n$ and $\text{Opt} \geq \binom{n}{2}$ in any $n$-vertex unweighted metric, it suffices to bound $T = \sum_{i \geq 0} 2^{i+2} \alpha n \cdot R_i$. From Claim 8, we have:

$$T \leq 4\alpha n R_0 + \alpha n \sum_{i \geq 1} 2^{i+2} [\frac{1}{\epsilon} R_{i-1} + (2 - \frac{1}{\epsilon})(n - N_i)]$$
$$= 4\alpha n R_0 + \frac{2}{\epsilon} \alpha n \sum_{i \geq 1} 2^{i+2} R_i + 4(2 - \frac{1}{\epsilon}) \alpha n \sum_{i \geq 1} 2^i (n - N_i)$$
$$= 4\alpha n R_0 + \frac{2}{\epsilon} T + 8(2 - \frac{1}{\epsilon}) \alpha n \sum_{i \geq 1} 2^i (n - N_i)$$
$$\leq 4\alpha n^2 + \frac{2}{\epsilon} T + 8(2 - \frac{1}{\epsilon}) \alpha \text{Opt}$$
$$\leq \frac{2}{\epsilon} T + 8(3 - \frac{1}{\epsilon}) \alpha \text{Opt}$$

So $T \leq O(1) \alpha \cdot \text{Opt}$, and $\text{Alg} \leq O(\alpha) \cdot \text{Opt}$.