Algorithmica (2013) 67:49–64
DOI 10.1007/s00453-012-9711-x

The School Bus Problem on Trees

Adrian Bock · Elyot Grant · Jochen Könemann · Laura Sanità

Received: 23 May 2012 / Accepted: 30 October 2012 / Published online: 21 November 2012

Abstract The School Bus Problem is an NP-hard vehicle routing problem in which the goal is to route buses that transport children to a school such that for each child, the distance travelled on the bus does not exceed the shortest distance from the child's home to the school by more than a given *regret* threshold. Subject to this constraint and bus capacity limit, the goal is to minimize the number of buses required.

In this paper, we give a polynomial time 4-approximation algorithm when the children and school are located at vertices of a fixed *tree*. As a byproduct of our analysis, we show that the integrality gap of the natural set-cover formulation for this problem is also bounded by 4. We also present a constant factor approximation for the variant where we have a fixed number of buses to use, and the goal is to minimize the maximum regret.

Keywords Vehicle routing · Approximation algorithm · Set-cover formulation

1 Introduction

Vehicle routing is an important and active topic in computer science and operations research. In the literature, the objective is typically to find a minimum-cost set of routes in a network that achieve a certain objective subject to a set of constraints. The constraints and cost are often related to the distance travelled, number of routes or vehicles used, coverage of the network by the routes, and so on. Problems of this kind are frequent and crucial in areas such as logistics, distribution systems, and public transportation (see, e.g., the survey by [14]).

A. Bock (✉)
EPFL, Lausanne, Switzerland
e-mail: adrianaloysius.bock@epfl.ch

E. Grant · J. Könemann · L. Sanità
University of Waterloo, Waterloo, Canada

In vehicle routing problems relevant to public transportation, a secondary objective often must be taken into account beyond minimizing operation cost: namely, it is crucial to design routes so as to optimize *customer satisfaction* in order to motivate customers to use the service. This requirement is essential in the so-called *School Bus Problem* (SBP)—the focus of this paper.

In the SBP, we must route buses that pick up children and bring them from their homes to a school. However, parents do not want their children to spend too much time on the bus relative to the time required to transport them to school by car along a shortest path. In fact, if the additional distance travelled by the bus exceeds a certain *regret* threshold, the parents would rather drive their children to school by themselves, which is unacceptable. Subject to this, the goal is to cover all of the children using a minimum number of buses.

Formally, we are given an undirected network $G(V,E)$ with distances on the edges $d: E \rightarrow \mathbb{Z}_+$, a node $s \in V$ representing the school, and a set $W \subseteq V$ representing the houses of children. Additionally, we are given a bus capacity $C \in \mathbb{Z}_+$, and a regret bound $R \in \mathbb{Z}_+$. The aim is to construct a minimum cardinality set $\mathcal{P}$ of walks ending at s (bus routes) and assign each child $w \in W$ to be the responsibility of some bus $p(w) \in \mathcal{P}$ such that (i) for each walk $P \in \mathcal{P}$, the total number of children w with $p(w) = P$ is at most the capacity C; (ii) for every child the regret bound is respected, that is: $d^P(w,s) \leq d(w,s) + R$, where $d^P(w,s)$ is the distance from the child w to the school s on the walk $p(w)$, and $d(w,s)$ is the shortest distance from w to s in the graph G.

In an additional variation of the problem, we have a fixed number N of buses we can use, and the goal is to minimize the maximum regret R. We call this variant the *School Bus Problem with Regret minimization* (SBP-R).

Like many vehicle routing problems, both SBP and SBP-R are strongly NP-hard, even on the simplest of graphs. To see this, consider a star centered at the school s with children located at all other vertices. If k of the edges are very long, say longer than $\frac{R}{2}$, then in any feasible solution we need at least k buses: one bus starting at each of these k endpoints. Then, determining if such k buses are sufficient with a regret bound R is precisely equivalent to solving the bin-packing decision problem with k bins of capacity $\frac{R}{2}$ and objects sized according to the distances from the remaining children to the school. It follows that bin-packing can be reduced to both SBP and SBP-R on stars.

Many variants of vehicle routing have been studied in the context of exact, approximate, and heuristic algorithms. For SBP and SBP-R, heuristic methods for practical applications have been examined (see the survey of [12]), but there is no literature concerning formal approximability and inapproximability results. Our goal is to advance the state of the art in this perspective.

1.1 Our Results

As we will show, the SBP can be formulated as a set covering problem. With this observation, we will easily derive a $\mathcal{O}(\log C)$-approximation to SBP in general graphs, as well as a $\mathcal{O}(\log C)$ upper bound on the integrality gap of the natural set-cover formulation applied to the SBP.

If we restrict the input to *trees*, we can achieve a constant factor approximation. Specifically, we will give a simple combinatorial 4-approximation for the SBP on trees. The approximation factor reduces to 3 in the case of unlimited capacity.

Our algorithm for SBP immediately yields an integrality gap bound matching the approximation factor. Namely, we will show that the integrality gap of the natural set-cover formulation of the SBP on trees is at most 4. In case of unlimited capacity, the gap is at most 3.

On the negative side, we can prove an inapproximability factor of $\frac{3}{2}$ for the SBP on trees.

Finally, we give a combinatorial 13.5-approximation for the SBP-R on trees, in the case of unlimited capacity.

1.2 Related Work

There is an enormous number of results concerning vehicle routing problems; see the survey [14].

The SBP is closely related (but not equal) to the Distance Constrained Vehicle Routing Problem (DVRP). DVRP enforces a bound D on the length of a vehicle tour, and the goal is the minimization of the number of routes used to visit all locations. It was raised and studied for applications in [9] and [10]. Routing problems like the DVRP can directly be encoded as instances of Minimum Set Cover, and thus often admit logarithmic approximations. The authors of [11] give a careful analysis of the set cover integer programming formulation of the DVRP and bound its integrality gap by $\mathcal{O}(\log D)$ on general graphs and by $\mathcal{O}(1)$ on a tree. They also obtain a constant approximation for the DVRP on a tree and a $\mathcal{O}(\log D)$ approximation in general. We remark here that, despite the similarity of the two problems, a straightforward adaptation of their methods to the SBP does not work. Therefore, in order to develop our approximation results for the SBP on trees, we need to introduce some new ideas.

The Capacitated Vehicle Routing Problem (CVRP) enforces a limit C on the number of visited locations in each route, and the goal is the minimization of the total length of all the routes. The paper [7] established a strong link to the underlying Travelling Salesman Problem (TSP) by giving an approximation algorithm that relies on the approximation algorithms for TSP. Depending on the capacity bound C, it is possible to obtain a PTAS in the Euclidean plane for some special cases (if the capacity is either small [1], or very large [2]). If we restrict the input to trees, there is a 2-approximation [8].

Many practical problems involving school buses have been studied, but primarily within the context of *heuristic methods* for real-life instances. We refer to [12] for a thorough survey of possible formulations and heuristic solution methods. Our notion of regret was first introduced as a vehicle routing objective in [13]. They considered a more general problem involving timing windows for customers and applied metaheuristics to produce solutions to real-life instances.

The School bus problem can be seen as a special case of the Vehicle Routing with time windows (VRP-TW), where every node is associated with a time interval in which it must be visited [6]. There is a $\mathcal{O}(\log^3 |V|)$ approximation algorithm for VRP-TW based on a $\mathcal{O}(\log^2 |V|)$ approximation for Orienteering with time windows [3].

2 Preliminaries

We first observe that the capacity bound can be neglected for a slight loss in the approximation factor for the SBP. The proof is essentially identical to that of a similar result proven in [11] for the DVRP, but we recall the proof for the sake of completeness.

Lemma 1 *Given an α-approximation to the SBP with unlimited capacity for each bus, there is an $\alpha + 1$-approximation to the SBP with capacity bound C.*

Proof Given an instance of the SBP with capacity bound C, we first ignore the capacity bound and run the α-approximation algorithm for the resulting SBP instance with unlimited capacity. The output will be a set of walks $P_1, \dots, P_k$ covering all children W. The idea is to cut each walk into parts of capacity C and connect them directly to the school. Let APX_C denote the number of buses output in case of capacity C and OPT_C, OPT_∞ the optimum solutions of the capacitated and uncapacitated case, respectively. We obtain

$$\begin{aligned} APX_C &\le \sum_{i=1}^{k} \left(\frac{|\{w \in W : w \text{ is covered by } P_i\}|}{C} + 1 \right) \\ &\le \frac{|W|}{C} + \alpha OPT_\infty \le (1+\alpha) OPT_C. \end{aligned}$$ □

If P is a walk ending at s and covering a subset S of nodes, then we say that P has regret R if a regret bound of R is respected for all children in S. The following useful fact holds for both the SBP and the SBP-R:

Proposition 1 *Let P be a walk starting at some node v and ending at s. For all nodes covered by P the regret bound R is respected if and only if it is respected for v.*

Proof Assume for contradiction that there is a node $w \in S$ with $d^P(w,s) > d(w,s) + R$ while $d^P(v,s) \le d(v,s) + R$. The triangle inequality then implies

$$d^P(v,s) \ge d^P(w,s) + d(v,w) > R + d(w,s) + d(v,w) \ge R + d(v,s),$$

a contradiction. □

We next give a covering integer programming formulation of the SBP. Let $\mathcal{S}$ be the family of all feasible sets of C or fewer children that can be covered by a single walk ending at s having regret at most R. We introduce a variable x_S for each $S \in \mathcal{S}$ and give the following formulation:

$$\begin{aligned} \min \quad & \sum_{S \in \mathcal{S}} x_S \\ \text{s.t.} \quad & \sum_{S : w \in S} x_S \ge 1 \quad \forall w \in W \\ & x_S \in \{0,1\} \quad \forall S \in \mathcal{S}. \end{aligned} \tag{IP}$$

Theorem 1 *There is a $\mathcal{O}(\log C)$-approximation to SBP in general graphs. Furthermore, the integrality gap of the LP relaxation of SBP's covering integer programming formulation can also be bounded by $\mathcal{O}(\log C)$.*

Proof An $\mathcal{O}(\log C)$-approximation algorithm easily follows from adapting the standard greedy strategy for set cover. Such a greedy algorithm, applied to an SBP instance, repeatedly searches for a feasible walk ending at s that picks up the maximum number of uncovered children, doing so until every child is picked up. At each iteration, we guess the starting point v^* (by trying all $|V|-1$ possibilities). Using Proposition 1, the resulting problem we are left with is to find a v^*, s-walk in G of length at most $d(v^*, s) + R$ visiting the maximum number of uncovered nodes in W. Such a problem is well known in the literature as the *Orienteering Problem*, and can be approximated within a constant factor[4, 5]. The results then follow by applying the classical Dual-Fitting set cover analysis (since the analysis is pretty standard, we omit the details and refer to [15]). □

We leave the interesting question open whether a constant factor approximation can be achieved for the SBP in general graphs. This is unknown for related vehicle routing problems like the Distance-constrained vehicle routing problem [11] or the vehicle routing problem with time windows [6] as well. However, we will show now how to obtain a constant approximation on *trees*.

In the remainder of this paper, we will mainly focus on the infinite capacity version of the SBP and SBP-R on a tree T with root s. We denote by $P(u, v)$ and $d(u, v)$ the unique path from u to v in T, and its corresponding length. For a subset of edges F, we let $d(F) := \sum_{e \in F} d(e)$. An Euler tour of a connected set of edges is a walk that visits each edge exactly twice. We note that subtrees of T that contain no vertices in W will never be visited by a bus in any optimal solution, and thus we can assume without loss of generality that all leaves of T contain children. In such an instance, a feasible solution will simply cover all of T with bus routes. Moreover, when assuming infinite capacity, any solution is still feasible if every node of T contains a child and we can therefore assume, without loss of generality, that $W = V$.

3 SBP on Trees

For the SBP on trees, we first present the 4-approximation. After that, we show that the integrality gap of the natural set cover formulation can be bounded by 4 as well. Finally, we prove that the SBP on trees is hard to approximate within a factor 3/2.

3.1 A 4-Approximation

Theorem 2 *There exists a polynomial time* 4*-approximation for the SBP on trees. The approximation factor reduces to* 3 *in the case of unlimited capacity.*

We prove Theorem 2 by first giving a combinatorial 3-approximation for the SBP with unlimited capacity on graphs that are trees, and subsequently applying Lemma 1. Our algorithm is based on the following intuitive observations:

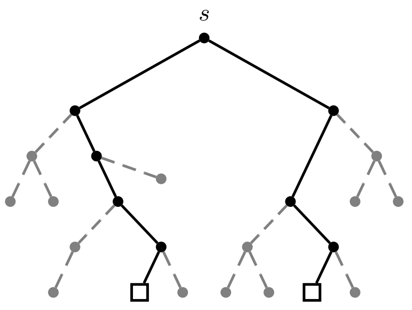

Fig. 1 Example of a tree with unit distance edges and $R = 6$. A maximal set of anchors is drawn as square nodes, the corresponding skeleton is shown in *solid* and the *grey*, *dashed parts* of the tree are the short subtrees

- When the input tree is very short (say, of height at most $\frac{R}{2}$ on an instance with regret R), then it is relatively easy to obtain a 2-approximation for the SBP by simply cutting an Euler tour of the tree into short pieces and assigning each piece to a bus.
- General trees can be partitioned into smaller pieces (subtrees) such that at least one bus is required for each piece, but each piece can be solved almost optimally via a similar Euler tour method.

We begin with some definitions. We call a set of vertices $\{a_1, \dots, a_m\} \subseteq V$ R-*independent* if for all $a_i \neq a_j$, we have $d(a_i, \mathrm{lca}(a_i, a_j)) > \frac{R}{2}$, where $\mathrm{lca}(a_i, a_j)$ is the lowest common ancestor of the vertices a_i and a_j in T. By iteratively marking the leaf in T furthest from the root such that R-independence is maintained among marked leaves, we can obtain, in polynomial time, an inclusion-wise maximal R-independent set of leaves A such that all vertices in T are within a distance of $\frac{R}{2}$ from a path $P(s, a)$ for some $a \in A$. We shall call A a set of *anchors*. By construction, no two distinct anchors a_i and a_j can both be covered by a walk of regret at most R, immediately yielding the following lower bound:

Proposition 2 *The size $|A|$ of the set of anchors is a lower bound on the number of buses that is needed in any feasible solution.*

We now give a second useful lower bound. Let $Q := \bigcup_{a \in A} P(s, a)$. We call Q the *skeleton* of T, noting that Q is a subtree of T whose leaves are the anchors. Observe that all edges in the skeleton Q will automatically be covered if a bus visits each anchor. Since each anchor must be visited at least once, it suffices to only consider covering the anchors and the *non-skeletal edges* of T, i.e. the edges in $T \setminus Q$. The edges in $T \setminus Q$ form a collection of disjoint subtrees, each of which has height at most $\frac{R}{2}$. We call these *short subtrees*. See Fig. 1 for an example tree visualizing these definitions.

Suppose that a feasible walk starts at a vertex v in a short subtree $\mathcal{T}$. It will cover all the edges in $P(s, v)$, and may possibly cover some additional detour edges having total length at most $\frac{R}{2}$. Since $\mathcal{T}$ is a short subtree, the non-skeletal edges in $P(s, v)$ have total length at most $\frac{R}{2}$. It follows that:

Observation 1 *The set of non-skeletal edges covered by any feasible walk P must have total length at most R: at most $\frac{R}{2}$ in length along the path from its starting vertex to the root, and at most $\frac{R}{2}$ length in edges covered by detours.*

From this, we can observe the following lower bound on the number of buses:

Proposition 3 *The number $\frac{1}{R}\sum_{e \in T \setminus Q} d(e)$ is a lower bound on the number of buses that are needed in any feasible solution.*

We build our 3-approximation from these two lower bounds by partitioning the edges of T into a family of subtrees each containing a single anchor, and approximating the optimal solution well on each of these subtrees. For anchors $A = \{a_1, \dots, a_m\}$, we define associated paths of edges $\{P_1, \dots, P_m\}$ as follows: (i) $P_1 = P(s, a_1)$, and (ii) $P_i = P(s, a_i) \setminus (\bigcup_{j=1}^{i-1}\{P_j\})$ for $2 \le i \le m$. The edges in $\{P_1, \dots, P_m\}$ form a partition of the skeleton Q into paths (not necessarily ending at s), each of which starts at a different anchor.

We then let T_i be the set of all edges in both the path P_i and the set of all short subtrees attached to P_i. If a short subtree is attached to a junction point where two paths P_i and P_j meet, we arbitrarily assign it to either P_i or P_j so that the sets $\{T_1, \dots, T_m\}$ form a partition of all of the edges of T into a collection of subtrees, each containing a single anchor.

For each $1 \le i \le m$ we define a directed walk W_i that starts at the anchor a_i, proceeds along P_i in the direction toward the root s, and collects every edge in T_i by tracing out an Euler tour around each of the short subtrees in T_i that are attached to P_i. One may easily verify that it is always possible to quickly find such a walk such that the following properties are satisfied:

- W_i contains each edge in P_i exactly once and always proceeds in the direction toward s when collecting each edge in P_i.
- W_i contains each edge in $T_i \setminus P_i$ exactly twice: once proceeding in the direction away from s, and once in the direction toward s.

We now greedily assign the edges in the short subtrees in T_i to buses by simply adding edges to buses in the order in which they are visited by W_i. We first initialize a bus β_1 at the anchor a_i and have it travel along W_i until the total length of all of the edges it has traversed in the *downward* direction (away from the root s) is exactly $\frac{R}{2}$. At this point, we assume it lies on some vertex v_1 (if not, we may imagine adding v_1 to the middle of an existing edge in T_i, although this will not be relevant to our solution as there are then no children at v_1). We send bus β_1 from v_1 immediately back to the root s and create a new bus β_2 that starts at v_1 and continues to follow W_i until it too has traversed exactly $\frac{R}{2}$ length in edges of T_i in the downward direction. We assume it then lies at a vertex v_2, create a new bus β_3 that starts at v_2 and continues to follow W_i, and so on. Eventually, some bus β_k will pick up the last remaining children and proceed to the root s, possibly with leftover detour to spare. We observe that the number of buses used is exactly $\lceil \frac{2\sum_{e \in T_i \setminus P_i} d(e)}{R} \rceil$ since each bus other than the last one consumes exactly $\frac{R}{2}$ of the downward directed edges in W_i, and W_i proceeds downward along each edge in $T_i \setminus P_i$ exactly once. We also note that this is a feasible solution since a bus travelling a total downward direction of $\frac{R}{2}$ must make a detour no greater than R.

Doing this for each edge set T_i yields a feasible solution to the original instance using exactly $\sum_{i=1}^{m} \lceil \frac{2\sum_{e\in T_i \setminus P_i} d(e)}{R} \rceil$ buses. This is at most

$$m + \frac{2}{R}\sum_{i=1}^{m} \sum_{e\in T_i \setminus P_i} d(e) = m + 2\frac{\sum_{e\in T\setminus Q} d(e)}{R} \leq 3OPT$$

by Proposition 2 and Proposition 3, where OPT is the optimal number of buses required in any feasible solution. Together with Lemma 1, this proves Theorem 2.

3.2 Integrality Gap of Set-Cover Formulation

We will next show that the bounds given in Propositions 2 and 3 are necessarily also respected by fractional solutions to the LP relaxation (LP) of (IP). Together with the argument above, this immediately implies that (LP) has an integrality gap of at most 4 (and 3 in the case of infinite capacities).

Theorem 3 *The integrality gap of the natural set-cover formulation of the SBP on trees is at most* 4. *In case of unlimited capacity, the gap is at most* 3.

Proof The dual of (LP) is a packing problem with a variable y_v for each $v \in V$; we think of y_v as the *profit* of child v. The dual then has an exponential number of constraints bounding the total profit of each feasible set of children that can be picked up in a single walk.

$$\begin{aligned} \max \quad & \sum_{v\in V\setminus\{s\}} y_v \\ \text{s.t.} \quad & \sum_{v\in S} y_v \leq 1 \quad \forall \text{ feasible sets of children } S \\ & y_v \geq 0 \quad \forall v \in V \setminus \{s\}. \end{aligned} \tag{D}$$

To prove our bound, we need to state the following lemma proven by [11]:

Lemma 2 (Distribution Lemma) *For any tree H with root r and distance function d on the edges, it is possible to distribute a total profit of π among the leaves of H such that the profit contained in any subtree F rooted at r is at most $\frac{d(F)}{d(H)} \cdot \pi$.*

There are three things to prove for a feasible fractional solution to the LP relaxation of (IP):

(i) The number $\frac{|V|}{C}$ is a lower bound on the value of any fractional solution.
This lower bound is trivial. We assign a profit of $\frac{1}{C}$ to every node $v \in V \setminus \{s\}$. Any walk that collects profit > 1 has to visit more than C nodes.

(ii) The size $|A|$ of the set of anchors is a lower bound on the value of any fractional solution.
The profit function that assigns a profit 1 to every anchor and 0 to all other vertices is a feasible solution to (D), since a feasible walk can never visit more than one anchor and thus collects profit at most 1.

(iii) The lower bound of Proposition 3 holds for all fractional solutions.
In order to show this, we apply the distribution lemma to every short subtree H from the collection $\mathcal{H}$ of short subtrees that form $T \setminus Q$. On each subtree H, an amount of $d(H)/R$ is distributed among its leaves. Every other vertex gets profit 0. Therefore we distribute in total a profit of $\sum_{e\in T\setminus Q} d(e)/R$ over the vertices of T.
It remains to prove the feasibility of this dual solution. Consider a feasible walk P in T and assume that it collects a profit > 1. P visits the following length among edges of short subtrees:

$$\sum_{e\in(T\setminus Q)\cap P} d(e) = \sum_{H\in\mathcal{H}} \frac{\sum_{e\in H\cap P} d(e)}{d(H)} d(H) = \sum_{H\in\mathcal{H}} \frac{\sum_{e\in H\cap P} d(e)}{d(H)} \operatorname{profit}(H)\cdot R$$

By the distribution lemma, we know that

$$\sum_{H\in\mathcal{H}} \frac{\sum_{e\in H\cap P} d(e)}{d(H)} \operatorname{profit}(H)\cdot R \geq \operatorname{profit}(P)\cdot R > R$$

This is a contradiction to the Observation 1.

Now we bring the three lower bounds together to obtain the claimed result. Let APX denote the feasible integer solution that we obtain from combining Theorem 1 with Lemma 1. Combining their proofs, we have:

$$APX \leq \frac{|V|}{C} + |A| + 2\frac{\sum_{e\in T\setminus Q} d(e)}{R} \leq 4\cdot OPT_f.$$

Note that in case of unlimited capacity, the term $\frac{|V|}{C}$ is omitted and we obtain an integrality gap of at most 3. □

The worst example that we are aware of has integrality gap 2. It consists of a star with $n+1$ nodes and edges of unit distance from the center. Set $R := 2(n-2)$ and the capacity C unlimited. The best integer solution uses exactly two routes while the fractional solution considers n routes (each possible route that skips exactly one leaf) with $\frac{1}{n-1}$ fraction. This yields a fractional solution close to 1 for n big enough and thus the claimed result.

3.3 Hardness of Approximation

Theorem 4 *The SBP is NP-hard to approximate within a factor $\frac{3}{2}$.*

Proof The reduction is from the subset sum problem. In an instance $\mathcal{I}$ of the subset sum problem, we are given a collection $\{a_1,\dots,a_m\}$ of m non-negative integers with

$\sum_{i=1}^{m} a_i$ even and $B := \frac{1}{2}\sum_{i=1}^{m} a_i$. The goal is to determine whether there exists a subset $S \subseteq [m]$ such that $\sum_{i\in S} a_i = B$. This problem is known to be NP-complete.

To complete our inapproximability proof, we construct an instance of SBP on trees as follows. The root s has 2 children c_1, c_2 and m additional children. The edges (s, c_1) and (s, c_2) have length $2B$, while for each $j = 1, \dots, m$, edge (s, j) has length a_j. The regret bound is $R := 2B$. Note that the size of the SBP instance is polynomial in the size of $\mathcal{I}$, and the construction runs in polynomial time.

Suppose $\mathcal{I}$ is a yes-instance; that is, there is some subset $S \subseteq [m]$ with $\sum_{j\in S} a_j = B$. Then we can construct one walk starting at c_1 and ending at s visiting vertices $\{j : j \in S\}$, and one walk starting at c_2 and ending at s visiting vertices $\{j : j \in [m] \setminus S\}$). Note that the regret of each walk is exactly $2B$.

Conversely, suppose $\mathcal{I}$ is a no-instance. Then we claim that the optimal value of the SBP instance is at least 3. Due to the length of the edges (s, c_1) and (s, c_2), the nodes c_1 and c_2 must be the starting nodes of two different walks, since otherwise we will exceed the regret bound. On the other hand, each of these 2 walks can only cover a subset $S \subseteq [m]$ with $\sum_{j\in S} a_j \le B$, otherwise the regret bound will be exceeded. Since $\sum_{i=1}^{m} a_i = 2B$, but there is no valid walk covering a subset $S \subseteq [m]$ such that $\sum_{j\in S} a_j = B$, we require at least 3 walks to cover all nodes. The result follows. $\square$

4 A 13.5-Approximation to the Uncapacitated SBP-R on Trees

In this section, the School Bus Problem with Regret Minimization (SBP-R) is considered. SBP-R differs from SBP because of the exchanged roles of maximum regret and number of bus routes. In case of SBP-R, the number of routes is bounded by a given parameter $N \in \mathbb{N}$ while the maximum regret is to be minimized. We present here a polynomial time 13.5-approximation algorithm for SBP-R with unlimited capacity.

Without loss of generality, we may assume the tree T to be binary. Suppose we can fix a value R for the regret. We will develop an algorithm that, given an instance and the value R, either outputs a set of at most N bus routes, with a maximum regret of $13.5R$, or asserts that every solution with at most N buses must have a regret value $> R$. Then, we can do binary search on the regret values and output the best solution found.

Suppose we have guessed a value for R. We begin by finding a set of anchors A with respect to R as described in Sect. 3.1. Our algorithm shall only generate solutions where each bus begins its journey at an anchor. As it turns out, this restriction will be very helpful and causes only a small loss in the approximation factor that we obtain. By assigning a bus to start at each of the anchors, we can ensure that the anchors and skeleton will be covered. However, this time, covering the short subtrees (as defined for SBP in Sect. 3.1) is much trickier because the number of buses we may use is limited. In fact, it may be the case that all feasible solutions using N buses require some bus to travel from an anchor, up part of the skeleton, and then down another part of the skeleton partially toward a different anchor, before returning upward to travel to the root. Consequently, greedy assignments of short subtrees to buses using methods like those used in Sect. 3.1 do not work for SBP-R.

We introduce a few additional definitions to help us describe how we get around this difficulty. Every vertex $v \in Q$ of the skeleton is called a *junction point* if it is

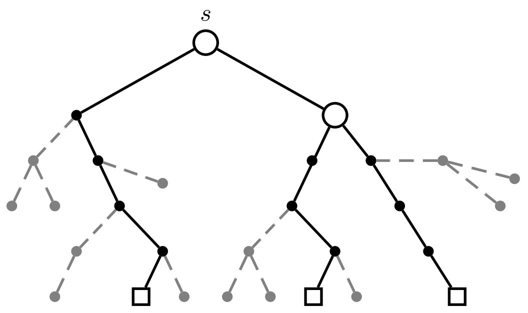

Fig. 2 Example of a tree with unit distance edges and $R = 6$. Anchors are drawn as *square nodes*, junction points as *empty circles*

either the root s or if v has degree more than 2 in Q. Let J be the set of junction points. The skeleton Q can be split at its junction points into a set of edge-disjoint paths, which we will call *core segments*. Formally, a path in Q is a core segment if and only if its endpoints are anchors or junction points and it contains no junction points in its interior. See Fig. 2 for an example tree visualizing these definitions.

Now, we employ the fact that, in a solution of regret R using N buses, no bus starting at some anchor $a \in A$ may travel too far off from the path from a to the root s; in particular, it may not travel a distance of more than $\frac{R}{2}$ along core segments not contained in the path from a to s. As intuition for how our algorithm exploits this, imagine taking each short subtree in T, detaching it from the skeleton Q, and then reattaching it to Q at some point $\frac{R}{2}$ closer to the root s from its original point of attachment in T. In the resultant tree T', a short subtree $\mathcal{T}'$ will be attached to the skeleton Q along the path from an anchor a to the root s if and only if the root of the corresponding short subtree $\mathcal{T}$ in T could be reached by a bus starting at a having regret at most R. Accordingly, by relocating all of the short subtrees, we effectively get rid of the need to consider buses that travel downward along portions of the skeleton. We will show that this relocation results only in a small loss in the approximation guarantee.

However, we must also ensure that the short subtrees themselves can be collected efficiently by a small number of buses. To do this, we cluster and cut the short subtrees into suitable pieces (called *tickets*), each of which can be collected efficiently by a bus starting at some particular anchor. By sizing the tickets appropriately, we also obtain a lower bound on the number of buses with regret R that are required to cover all of the points.

The next lemma shows how we obtain suitable tickets from a collection of short subtrees.

Lemma 3 *There exists a polynomial-time algorithm that, given a vertical path P in the skeleton Q, and a collection $\mathcal{C}$ of short subtrees whose roots lie on P, produces a partition of the edges of $\mathcal{C}$ into tickets $E_1^{\mathcal{C}}, \dots, E_k^{\mathcal{C}}$ with $k \le \lfloor \frac{\sum_{\mathcal{T} \in \mathcal{C}} d(\mathcal{T})}{R} \rfloor$ and an overhead ticket $E_0^{\mathcal{C}}$ such that:*

(P1) *All of the edges in $E_0^{\mathcal{C}}$ can be collected with an additional regret $\le 2.5R$ by a single bus whose route contains P.*
(P2) *For all $1 \le i \le k$, all the edges in $E_i^{\mathcal{C}}$ can be collected with an additional regret at most $3R$ by a single bus whose route contains P.*

Proof First, we identify the roots of all short subtrees with the lowest node v of path P. The idea is to cut the Euler tour of all short subtrees in $\mathcal{C}$ into suitable pieces. Starting from v, cut it at the first node such that the current piece has length $> 2R$. Continue like this to obtain $k+1$ walks. Denote by $E_1^{\mathcal{C}}, \ldots, E_k^{\mathcal{C}}$ all but the last piece. The last part of the Euler tour defines the overhead ticket $E_0^{\mathcal{C}}$. Note that $k \leq \lfloor \frac{\sum_{\mathcal{T} \in \mathcal{C}} d(\mathcal{T})}{R} \rfloor$, and that every cutting point is endpoint of a ticket and starting point of another ticket, thus it is covered by exactly two tickets. Since every node needs to be covered only once in a solution, we can ignore the last edge in each set $E_i^{\mathcal{C}}$ $(1 \leq i \leq k)$. By construction, the resulting length is at most $2R$. Both the starting and the end point of each ticket $E_i^{\mathcal{C}}$ $(1 \leq i \leq k)$ are at distance at most $\frac{R}{2}$ from v (cf. the definition of a short subtree). Since the Euler tour goes back to v after one subtree is finished, we obtain from a ticket a set of tours of total length at most $3R$. Note that we can arrange those tours in bottom-up order of the roots of visited short subtrees on the skeleton. A bus whose root walk contains P can then cover all edges corresponding to a ticket with regret at most $3R$. The last piece $E_0^{\mathcal{C}}$ of the Euler tour remaining from the cutting procedure certainly has length $\leq 2R$. Since the Euler tour goes back to v, only the distance to the starting point of the last piece has to be connected to v in order to obtain a set of tours. As in the previous case, one bus can cover these tours with regret $2.5R$, since the height of each subtree is at most $\frac{R}{2}$.

This strategy fulfills the properties (P1) and (P2) and runs in polynomial time. □

The next simple lemma will be useful later when we assign the overhead tickets to buses.

Lemma 4 *One can find a mapping $\phi : J \longrightarrow A$ from junction points to anchors in polynomial time with the following properties*:

(i) *Every junction point $j \in J$ is mapped to an anchor $\phi(j)$ down in the subtree rooted at j*;
(ii) *For all $a \in A$, there is at most one junction point $j \in J$ with $\phi(j) = a$.*

Proof We construct ϕ by iteratively considering anchors $A = \{a_1, \ldots, a_m\}$ and building the skeleton Q using paths from anchors to junction points. We begin with the path $P(s, a_1)$ and set $\phi(s) = a_1$. When each new path $P(s, a_i)$ is added, a new junction point j_i is formed, namely the lowest intersection point of $P(s, a_i)$ with the previous paths. We set $\phi(j_i) = a_i$ for the remaining i, and we easily see that the resulting mapping ϕ satisfies properties (i) and (ii). □

Now, for our final algorithm, we don't actually detach, move, and reattach all of the short subtrees in T. Instead, we employ a bit more care and cleverness to combine this step with the ticketing process. For every core segment S, let $t(S)$ and $b(S)$ be the top and the bottom junction points in S. Furthermore, let $r(S)$ be the highest junction point at distance at most $R/2$ from $t(S)$ (see Fig. 3a). The junction point $r(S)$ represents the common ancestor of all anchors from which a bus could reach S without exceeding the regret R. Our algorithm essentially deals with all short subtrees on S by converting them to tickets, and then 'placing' all of these tickets on $r(S)$, where they will be collected by buses travelling through the junction $r(S)$.

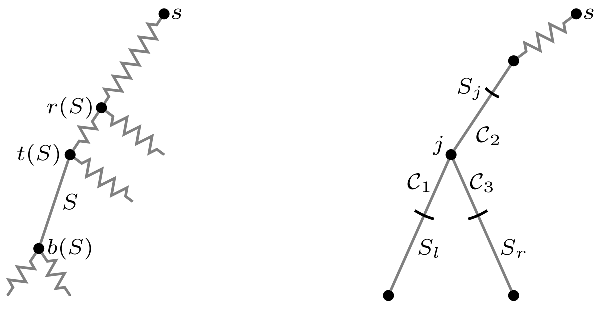

Fig. 3 Illustrations for Algorithm 1

Formally, our algorithm proceeds as follows.

Algorithm 1

```
1. Find a maximal $R$-independent set of anchors $A$, i.e. leaves such that a feasible
   bus can visit at most one of them. This defines a skeleton $Q$ of the instance $T$.
2. Initialize a default bus at each anchor $a \in A$.
3. For each junction point $j \in J$ in bottom-up order do:
   Assign an arbitrary left-to-right ordering of the two segments $S_l, S_r$ with $t(S_l) =$
   $j = t(S_r)$.[1] See also Fig. 3b.
   (a) Let $\mathcal{C}_1$ be the collection of short subtrees whose root node lies in the left core
       segment $S_l$ at distance $\leq R/2$ from $j$. Let $\mathcal{C}_2$ be the collection of short subtrees
       whose root node lies in the core segment $S_j$ with $b(S_j) = j$ at distance $> R/2$
       from $t(S_j)$. Note that $C_2 = \emptyset$ is possible.
   (b) Apply Lemma 3 to $\mathcal{C}_1 \cup \mathcal{C}_2$ on $P = S_l \cup S_j$, and obtain tickets $E_1, \ldots, E_y$ and
       overhead ticket $E_0$.
   (c) Let $\mathcal{C}_3$ be the collection of short subtrees whose root node lies in the right core
       segment $S_r$ at distance $\leq R/2$ from $j$.
   (d) Apply Lemma 3 to $\mathcal{C}_3$ on path $P = S_r$, and obtain tickets $F_1, \ldots, F_z$ and
       overhead ticket $F_0$.
   (e) Assign $E_0$ and $F_0$ to the default bus at $\phi(j)$.
   (f) Place the $y$ tickets $E_1, \ldots, E_y$ and $z$ tickets $F_1, \ldots, F_z$ at $r(S_l) = r(S_r)$.
4. For each anchor $a \in A$ do:
   Let $\mathcal{C}_a$ be the collection of short subtrees whose root node lies in the core segment
   $S_a$ with $b(S_a) = a$ at distance $> R/2$ from $t(S_a)$.
   (a) Apply Lemma 3 to $\mathcal{C}_a$ on $P = S_a$, and obtain tickets $K_1, \ldots, K_w$ and overhead
       ticket $K_0$.
   (b) Assign $K_0$ to the default bus at $a$.
   (c) Place the $w$ tickets $K_1, \ldots, K_w$ at $a$.
5. Assign every bus greedily the lowest ticket available on its path to the root.
6. Add a new bus for each unmatched ticket.
```

[1]If $j = s$, then $S_l = S_r$ is possible.

Let B be the number of bus routes output by the algorithm. We show:

(i) the regret of each route output by the algorithm is at most $13.5 \cdot R$ and
(ii) B is a lower bound on the number of buses with regret at most R that are needed to cover all nodes.

Lemma 5 *Algorithm* 1 *outputs a set of buses with maximum regret* $13.5R$.

Proof Any bus starting at an anchor a collects at most 4 different regret amounts:

– at most one non-overhead ticket in step 5. Suppose that the ticket corresponds to short subtrees with roots on core segment S. We claim that these roots are at distance at most R from $P(s,a)$. To see this note that the ticket might have been placed at $r(S)$ in step 3(f) of the algorithm. In this case, $P(s,a)$ is by definition at distance at most $R/2$ from the top end point $t(S)$ of segment S, and from there the short subtrees hanging off S can be reached within an additional distance of at most $R/2$. The claim follows, and together with property $(P2)$ from Lemma 3, we can cover a ticket with a walk of regret $\leq 5R$.
– at most two remaining pieces of the Euler tour in step 3(e). There is at most one junction point j with $\phi(j) = a$ by (ii) of Lemma 4. For a junction point j, each part E_0 and F_0 can be covered with regret $\leq 2.5R$ by (P2). Note that an additional regret R can be necessary to get to the roots of the short subtrees in either E_0 or F_0 that do not lie on the path from a to the root.
– at most one remaining piece of the Euler tour at a assigned in step 4(b). This ticket K_0 can be covered with regret $\leq 2.5R$ by (P2).

In total, the bus from anchor a collects regret of at most $5R + 5R + R + 2.5R = 13.5R$. □

In the following, we refer to the non-overhead tickets that are only assigned in step 6 of the algorithm as *unmatched tickets*. Let B be the number of bus routes output by the algorithm. We obtain:

Lemma 6 *B is a lower bound on the number of buses with regret at most R needed to cover all points.*

Proof We can interpret B as the number of anchors plus the number of unmatched tickets. We traverse the tree now bottom up from the anchors and inductively compute at each anchor or junction point a lower bound on the number of buses needed to cover some subtree below. Let j be a node that is either an anchor or a junction point. Denote by T_j the subtree of the skeleton rooted at j, together with all the (non-overhead) tickets placed within this subtree. Our claim is that the number of buses with regret R needed to cover T_j is at least the number of anchors in T_j plus the number of unmatched tickets in T_j. We distinguish two cases at j:

Case 1: There is at least one unmatched ticket placed at j. In T_j, the number of non-overhead tickets clearly exceeds the number of anchors by exactly the number of unmatched tickets within T_j. However, the number of non-overhead tickets is a lower bound by Observation 1 and our claim holds.

Case 2: There is no unmatched ticket placed at j. If j is an anchor, we clearly need one bus to pick up j. Otherwise, we can focus on the lower bounds at the two successor junction points j_1, j_2 below j. We claim that a bus starting in the subtree T_{j_2} can not go to the subtree T_{j_1} to pick up edges from a ticket (and vice versa). To prove this, we can assume that a ticket $\mathcal{T}$ is placed at j_1. Consider an edge $e \in T \setminus Q$ that is contained in $\mathcal{T}$. We distinguish the cases how the ticket $\mathcal{T}$ was formed to prove that every bus that covers e cannot start in T_{j_2}. Let T_e denote the short subtree containing e.

– T_e is rooted at a core segment S at distance greater than $R/2$ from $t(S) = j$, i.e. $T_e \in \mathcal{C}_2$ for j_1 ($T_e \in \mathcal{C}_a$ if j_1 is an anchor a). Then a bus starting in T_{j_2} can not pick up an edge of T_e, since the regret of the corresponding walk would be $> R$.
– T_e is rooted on a core segment S with $r(S) = j_1$, i.e. $T_e \in \mathcal{C}_1 \cup \mathcal{C}_3$ for some junction point j' (possibly $j' = j_1$). By the definition of $r(S)$, it follows that T_e is rooted at distance greater than $R/2$ from j. Therefore, a bus starting in T_{j_2} can not pick up an edge of T_e, since the regret of the corresponding walk would be $> R$.

It follows that the lower bound at j is the sum of the buses needed at each of the disjoint subtrees rooted at j_1 and j_2. Note that the number of anchors (unmatched tickets, resp.) within T_j is exactly the sum of the anchors (unmatched tickets, resp.) within T_{j_1} and T_{j_2}. This proves our induction step, and therefore our claim. □

Theorem 5 *There exists a polynomial time* 13.5*-approximation algorithm for the SBP-R on trees in the case of unlimited capacity.*

Proof Observe that Lemmas 5 and 6 yield a 13.5-approximation algorithm for SBP-R on trees: Given a value for R, we find a feasible solution to the SBP-R with value at most $13.5 \cdot R$ if $B \leq N$ holds. If not, then Lemma 6 ensures that $OPT > R$. Thus doing binary search for R yields the result. □

References

1. Adamaszek, A., Czumaj, A., Lingas, A.: PTAS for k-tour cover problem on the plane for moderately large values of k. In: Algorithms and Computation. LNCS, vol. 5878, pp. 994–1003. Springer, Berlin (2009)
2. Asano, T., Katoh, N., Tamaki, H., Tokuyama, T.: Covering points in the plane by k-tours: towards a polynomial time approximation scheme for general k. STOC (1997)
3. Bansal, N., Blum, A., Chawla, S., Meyerson, A.: Approximation algorithms for deadline-TSP and vehicle routing with time windows. In: Proceedings of the 36th Annual ACM Symposium on Theory of Computing, pp. 166–174 (2004)
4. Blum, A., Chawla, S., Karger, D.R., Lane, T., Meyerson, A., Minkoff, M.: Approximation algorithms for orienteering and discounted-reward TSP. SIAM J. Comput. **37**(2), 653–670 (2007)
5. Chekuri, C., Korula, N., Pal, M.: Improved algorithms for orienteering and related problems. In: SODA, pp. 661–670 (2008)
6. Desrochers, M., Desrosiers, J., Solomon, M.: A new optimization algorithm for the vehicle routing problem with time windows. Oper. Res. **40**, 342–354 (1992)
7. Haimovich, M., Rinnoy Kan, A.H.G.: Bounds and heuristic for capacitated routing problems. Math. Oper. Res. **10**(4), 527–542 (1985)
8. Labbe, M., Laporte, G., Mercure, H.: Capacitated vehicle routing on trees. Oper. Res. **39**(4), 616–622 (1991)

9. Laporte, G., Desrochers, M., Norbert, Y.: Two exact algorithms for the distance constrained vehicle routing problem. Networks **14**, 47–61 (1984)
10. Li, C.-L., Simchi-Levi, S., Desrochers, M.: On the distance constrained vehicle routing problem. Oper. Res. **40**, 790–799 (1992)
11. Nagarajan, V., Ravi, R.: Approximation Algorithms for Distance Constrained Vehicle Routing Problems. Tepper School of Business, Carnegie–Mellon University Press, Pittsburgh (2008)
12. Park, J., Kim, B.-I.: The school bus routing problem: a review. Eur. J. Oper. Res. **202**, 311–319 (2010)
13. Spada, M., Bierlaire, M., Liebling, Th.M.: Decision-aiding methodology for the school bus routing and scheduling problem. Transp. Sci. **39**(4), 477–490 (2005)
14. Toth, P., Vigo, D.: The Vehicle Routing Problem. SIAM, Philadelphia (2001)
15. Vazirani, V.V.: Approximation Algorithms. Springer, Berlin (2001)