Improving Customer Proximity to Railway Stations*

Evangelos Kranakis[†‡] Paolo Penna[§] Konrad Schlude[§] David Scot Taylor[§]
Peter Widmayer[§]

Abstract

We consider problems of (new) station placement along (existing) railway tracks, so as to increase the number of users. We prove that, in spite of the NP-hardness for the general version, some interesting cases can be solved exactly by a suitable dynamic programming approach. For variants in which we also take into account existing connections between cities and railway tracks (streets, buses, etc.) we instead show some hardness results.

Keywords: Railway optimization, facility location, dynamic programming.

1 Models and Problems

There are many instances when public or private sector bodies are faced with making decisions on how to allocate facilities optimally. Such problems with mathematically quantifiable optimization constraints have been studied extensively in the scientific literature (e.g., see the book [6]). Recently the European Union has been encouraging the privatization of railway assets in various EU countries in order to improve system efficiency as well as customer satisfaction. In this paper we approach one such problem by studying how customer proximity can affect the railway station location. More specifically, given a set of settlements and an existing track, one wishes to build a set of new stations such that (some of) the settlements can easily access those stations and, thus, use the railway. This gives a gain in terms of (potentially) new users, but it also turns into a cost for the old ones (for instance, a new station results into a delay for those trains traveling on the track). Let us consider the following problem:

Input: A set of $P = \{p_1, \ldots, p_n\}$ of settlements (i.e. points) on the Euclidean plane, each of them with an associated *demand* d_i, and an existing railway, that is, a set of straight-line segments forming a connected polygonal and whose endpoints represent existing stations (see Fig. 1).

Solution: A set of new stations along the track.

Given a solution to this problem, we have a *gain* and a *cost* function due to the new stations. The cost of building a new station, in general, depends on the position we are placing it. In the sequel, we describe some possible definitions for the gain function. All such definitions are distance-based, that is, the gain due to the new stations depends on how far a settlement is from its closest new station. We will first assume that the distance is the Euclidean one (although, some of the results can be extended to other metrics).

Single radius. We first consider the following (simplified) scenario. A certain settlement p_i is far away from every existing station. So, for the people living there it is not worth to use the railway. If we build a new station which is close enough to p_i (let us say at distance less than R) then the railway transportation becomes "competitive" with respect to other transportations and *all* the people in p_i (let their number be d_i) will use this new station. We then have the following model: a settlement p_i uses a (newly built) station if and only if (a) this station is at distance less than or equal to some *radius* R and (b) no existing station was at distance less than R. Fig. 1 shows an example.

*Work partially supported by the Swiss Federal Office for Education and Science under the Human Potential Programme of the European Union under contract no. HPRN-CT-1999-00104 (AMORE).

†School of Computer Science, Carleton University, Canada, E-mail: kranakis@cs.carleton.ca.

‡Research of E. Kranakis was supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) and MITACS (Mathematics of Information Technology and Complex Systems) grants.

§Institute for Theoretical Computer Science, ETH Zentrum, CH-8092 Zürich, Switzerland, E-mail: {lastname}@inf.ethz.ch.

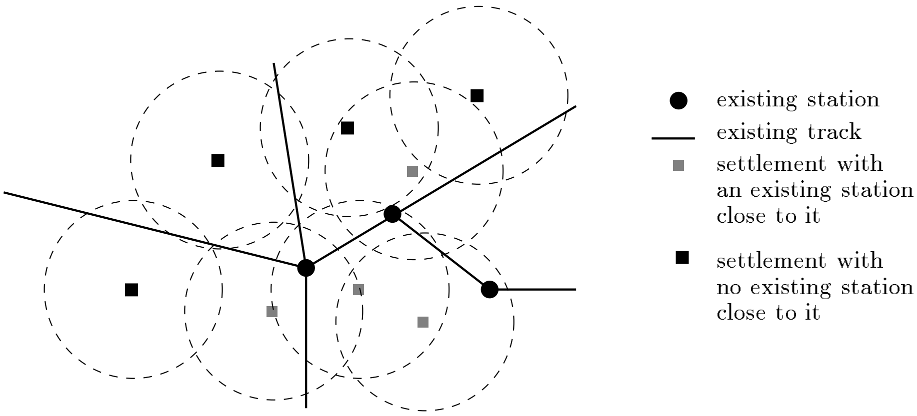

Figure 1: The single radius model: people living in a settlement can use the railway if the radius of the settlement contains at least one existing station.

Notice that we can assume w.l.o.g. that no settlement in P is currently "covered" by the existing stations. Hence, the gain of a set S of new stations is the sum of the demands d_i of those p_i that are covered by the radius of some $s \in S$. Formally,

$$\sum_{i=1}^{n} d_i \cdot cover(S, p_i),$$

where $cover(S, p_i)$ equals 1 if there exists an $s \in S$ at distance less than or equal to R from p_i, and it equals 0 otherwise.

Distance based costs. Notice that the single radius model is, in some cases, too unrealistic since it assumes that a station at distance $R = 500m$, for instance, is accessible, while a station at distance $R' = 550m$ is not. A more realistic model should take into account the fact that the closer a station is the more (potential) customers from a settlement are expected. For instance, we could say that the expected number of users from p_i is $d_i/(\delta + 1)$, where δ is the distance of p_i to the closest station. More generally, given a *monotone* (decreasing) function $\alpha(\cdot)$, the gain of a set of new stations can be expressed as

$$\sum_{i=1}^{n} d_i \cdot \alpha(\delta(p_i, S))$$

where $\delta(p_i, S)$ is the distance between p_i and the closest station in S.

Multiple radii. This setting is somewhat in between the two previous ones. Indeed, it can be used to approximate any distance based cost function with a fixed set of radii. Roughly speaking, these radii result from a "discretization" of an arbitrary function $\alpha(\cdot)$. For instance, the function $\alpha(\delta) = 1/(\delta + 1)$ can be approximated by two (or more) radii $R_1 = 1$ and $R_2 = 2$: the first radius captures the cases in which the gain $\alpha(\delta)$ is between 100% and 50% of the settlement demand, while R_2 corresponds to a gain between 33% and 50%. Clearly, the more radii we consider, the better we approximate $\alpha(\cdot)$.

Two optimization problems for single radius. We now focus on the single radius model and we assume the cost of building a new station to be constant. For this version, one can envision the following two optimization problems:

- Min Number of Stations (Min Station): minimize the number of stations needed to cover all the settlements.
- Budget Constrained Max Gain (Max Gain): given an integer k, with $1 \leq k \leq n$, find the placement of k stations that maximizes the gain.

We first observe that the second problem can be easily used to solve the first one: one just has to try all the k from 1 up to the smallest one for which the gain is the biggest possible, that is, we cover all the settlements. On the other hand, the other way round does not necessarily work. The limitation on the number of new stations seems to complicate things: with only k new stations at hand, we may not be able to cover all the settlements. In this case, our task is to find the best subset of settlements that can be covered with k stations only.

1.1 Previous Work

Our model is inspired by [10]. In that paper, the authors consider two different variants. The first one corresponds to what we here (re-) named single radius model (*accessibility model* in [10]). For this model the authors proved that, when a line set $\mathcal{L}$ (i.e., tracks) and a set S of *integer* points (i.e., settlements) are given, finding the best placements for k stations is NP-hard. The second model, named *travel time model*, takes into account the saved travel time over all the travelers. In contrast with the accessibility model, here it is assumed that all people in a settlement use the closest station and there is no a priori limit on the number of new stations we are allowed to build. As also mentioned above, there is a clear trade-off between the saved travel time due to a new station closer to some settlements, and the increased travel time due to the fact that trains must stop several times. Minimizing the saved travel time is also NP-hard [10].

The *geometric disk cover* problem [12] (related to the single radius model) asks to cover a set of points in the plane with the minimum number of disks of unit radius. This and other variants, in which the possible locations for the center of the disks is given in the input, admit a polynomial-time approximation scheme [12, 7], which easily applies also to the Min Station problem. As for more general distance based functions, we observe that placing the minimum number of stations to maximize the gain is a restriction of *uncapacitated facility location with metric spaces* [9]. The latter problem (and hence our problem(s)) admits constant-ratio approximation algorithms [9, 21]. Moreover, the case in which the *service cost* is the Euclidean distance (in our model this corresponds to choose $\alpha = 1/\delta$) has a polynomial-time approximation scheme [1] (in short PTAS). The same paper also gives (with the same technique) a PTAS for the k-median problem, while constant-factor approximation algorithms for the metric version are presented in [4, 14]. Notice that the Max Gain problem is a special case of the k-median one (while in the facility location problem there is no such restriction for the number of new facilities to open). Noticeably, the Euclidean version of the k-median problem is already NP-hard [17].

The main difference between the above (more general) problems and our problem(s) concerns the restriction on the possible locations for the new stations. Indeed, in practice it is quite unlikely that the radius associated with a settlement crosses several tracks far apart from one another: if somebody walks to a station, then the distance he/she can cover is relatively small; if he/she goes by car, then probably driving for more than, let us say, 10 minutes to reach such a station would already make the railway transportation not that convenient (with respect to simply driving to the destination). This is confirmed by the data from the *Deutsche Bahn AG* used for the experiments in [10] and from the *Swiss Federal Railways SBB* [15].

Therefore, in many cases the whole network can be broken down into simpler smaller components. These components are nothing but single segments (i.e. parts of a track) and the solution of one segment does not affect the others. Actually, as already observed in [10], if every radius intersects the railway network in at most one interval, then the Min Station problem is polynomially solvable. Also, the Max Gain problem can be formulated as (uncapacitated) k-facility location problem with unimodular matrices [20], which is solvable in polynomial time [16, Chapt. 3.1] (see also [19] for more efficient methods). In [13] the case of facilities and customers located both on a line at n given points and cost service function satisfying the unimodal property (a generalization of Euclidean case [11]) an efficient dynamic programming approach is given. This result can be also used to obtain efficient algorithms for the single track versions in which we do no have a single radius per settlement (namely monotone cost functions, which include the multiple radii case). In the same paper, the authors also proved the NP-hardness of generalization of the unimodal case (namely, bimodal functions).

Finally, in [20] several variant problems have been studied, including non-continuous versions in which the possible locations of the stations is not given a priori.

1.2 Our Results

In this work we focus on the Max Gain problem in the single radius model. In particular, we aim in finding efficient exact algorithms for interesting cases that *do not* satisfy the unimodular property assumption [10].

To this aim, in Sect. 2 we present a novel dynamic programming approach for the single straight-line track. This restriction is solvable (within a better time complexity) using the results in [11, 22]: Indeed, the result in [11] implies an $O(n^2)$-time extact algorithm, which can be further improved to $O(kn \log n)$ by using orthogonal range queries [22]. However, we use the ideas contained in our dynamic programming approach to solve more complex situations where the results of [11, 13] do not apply and do not yield exact polynomial-time algorithms. The natural extension of the single straight-line track is the case in which we have two parallel tracks and radii may intersect *both* of them. As we discuss at the beginning of Sect. 3, this apparently simple version of the problem already contains some complicating factors that make a natural extension of the dynamic approach fail. However, we are still able to modify our dynamic programming to exactly solve the following two versions:

- There is a minimum distance between consecutive stations we want to place; (Sect. 3.1)
- All settlements lie in between the two tracks. (Sect. 3.2)

The first variant is motivated by the practical consideration that putting two stations very close to each other has the only negative effect of delaying trains. The second case makes a non-trivial use of geometric properties of the radii generated by the settlements and might be of theoretical interest towards a characterization of those instances that admit polynomial-time exact algorithms. In both cases, the techniques used in [11, 13] do not apply.

In Sect. 4 we show how an exact algorithm for the single track problem can be also used to exactly solve a problem of simultaneously building a new straight-line track and new stations on it: in this case we also have to decide where the new track should lie.

Finally, in Sect. 5, we go back to non Euclidean cases (motivated by the existence of streets/buses connecting settlements to the tracks) and show that, even with a single track, this version of the Max Gain problem is NP-hard. Moreover, the corresponding Min Station problem is hard to approximate within a factor $c \log n$, for some $c > 0$. This highlights the role played by the "geometry" in our solutions and indicates the need of some assumptions on the geometry of the streets to obtain exact (in some cases even approximate) solutions in polynomial time.

2 Dynamic Programming

In this section we describe our exact algorithm for the problem restricted to single radius and constant cost per new station for the case of only a single track. The main ideas of this algorithm will be used in the sequel to solve the two parallel track versions.

2.1 One Single (Straight Line) Track

We first observe that a circle around a settlement p_i with radius R intersects the track in an interval I_i. By construction, this interval is the only region that can contain a new station serving p_i. Moreover, if we place a station in the intersection of two (or more) intervals I_i, I_j, then this station will cover *all* the corresponding settlements.

So, the Max Gain problem translates into the following one:

- Max Gain 1 Track: Given a collection of (weighted) intervals[1] on a line L and an integer k, find k points on L that maximize the sum of the weights of the intervals containing at least one of such points.

We first observe that there is only a finite set of points on the line that must be taken into account to optimally place the k stations: the endpoints of the intervals. Hence, a simple brute-force approach yields an algorithm whose complexity is $O(n^k)$, where n is the number of intervals. However, a more efficient approach can be used to have a running time polynomial in both n and k. Consider a set of k stations s_1, s_2, up to s_k from left to right. Then, we hope that the set of intervals that contain *only* the station s_k can be computed

[1] The weight of interval I_i equals the demand d_i.

by looking (only) at the position of s_{k-1} (i.e. it is independent from what the solution on the left of s_{k-1} is). Intuitively, this will allow us to break the instance into two *independent* subproblems: on the "left" of s_{k-1} (there we have to use at most $k-1$ stations) and on the right of it (where only one station must be located). This will be accomplished by a suitable dynamic programming approach. Before describing this, we first prove formally the above statement.

Lemma 1 *Given a collection of intervals on a line L and given three points i_1, i_2 and i_3 on L, with $i_1 < i_2 < i_3$,*

$$\mathsf{contain}(i_2, i_3) \subseteq \mathsf{contain}(i_1, i_3),$$

where $\mathsf{contain}(i, j)$ *denotes the set of intervals containing j and not containing i.*

Proof. Consider an interval I in $\mathsf{contain}(i_2, i_3)$. By definition, $i_3 \in I$ and $i_2 \notin I$. Since $i_1 < i_2 < i_3$, it follows that $i_1 \notin I$. Hence, $I \in \mathsf{contain}(i_1, i_3)$. This proves the lemma. □

Intuitively, if we are given a set of stations in which the rightmost one is in position i and we add a new station in position j on the right of i, the gain due to the new station is given by

$$\mathsf{gain}(i, j) = \sum_{I \in \mathsf{contain}(i,j)} w(I), \tag{1}$$

where $w(I)$ is the weight associated to the interval I. We are now in a position to describe the dynamic programming algorithm to solve the problem. Given a set of intervals, let $\mathsf{opt}(i; k)$ denote the minimum cost among the solutions that use k stations with the rightmost one in position i. Then, the following lemma states how $\mathsf{opt}(i; k)$ can be computed if we have already computed this value for all the possible $i' < i$ and $k' < k$.

Lemma 2 *For any i and for any integer $k \geq 2$, the following condition holds:*

$$\mathsf{opt}(i; k) = \max_{i' < i} \{\mathsf{opt}(i'; k-1) + \mathsf{gain}(i', i)\}, \tag{2}$$

where $\mathsf{gain}(\cdot, \cdot)$ *is defined as in Eq. 1.*

Proof. Let $i_1^*, i_2^*, \ldots, i_{k-1}^*, i$ be an optimal placement of k stations (from the leftmost to the rightmost one) subject to the rightmost one being in position i. The contribution of i is given by the set of intervals containing only the station i. By definition and from Lemma 1 this set is equal to

$$\bigcap_{j=1}^{k-1} \mathsf{contain}(i_j^*, i) = \mathsf{contain}(i_{k-1}^*, i).$$

Hence, $\mathsf{gain}(i_{k-1}^*, i)$ is the contribution of the rightmost station. To complete the proof, we show that the contribution of $\{i_1^*, \ldots, i_{k-1}^*\}$ is equal to $\mathsf{opt}(i_{k-1}^*; k-1)$. Indeed, this set of stations is a feasible solution for the problem of placing $k-1$ stations, subject to the rightmost one being in position i_{k-1}^*. So, if the contribution of $\{i_1^*, \ldots, i_{k-1}^*\}$ is less than $\mathsf{opt}(i_{k-1}; k-1)$, then there exists $S' = \{i_1', \ldots, i_{k-2}', i_{k-1}^*\}$ yielding a bigger contribution. Moreover, $S' \cup \{i_k^*\}$ is a feasible solution for the k stations placement problem (with the restriction that i is the position of the rightmost one) and its measure is bigger than $\{i_1^*, \ldots, i_{k-1}^*, i\}$ (notice that the contribution of i does not change). This contradicts the hypothesis that $\{i_1^*, \ldots, i_{k-1}^*, i\}$ was optimal. □

The above lemma allows us to efficiently compute an optimal solution. Indeed, we first observe that we need to compute tables $\mathsf{gain}(\cdot, \cdot)$ and $\mathsf{opt}(\cdot; \cdot)$ only for at most $2n$ values: the endpoints of the intervals. Hence in the following the point (or position) i will denote the ith endpoint of this set from left to right.

Lemma 3 *The table* $\mathsf{gain}(i, j)$ *can be computed for all $1 \leq i < j \leq 2n$ within $O(n^2)$ time.*

Proof Sketch. For every i and j, with $1 \leq i < j \leq 2n$, $\mathsf{gain}(i, j)$ is the sum of the weights of those intervals that start to the right of i and on the left of j. For every possible i, we construct a balanced binary tree T_i serving us as an interval tree [5, pp 294-295, Problem 15-1] as follows. In T_i we store the weights of all the intervals whose left endpoint is greater than i. In particular, each interval can be represented in T_i with a single (possibly internal) node. We thus store the weight of such an interval into these nodes. After

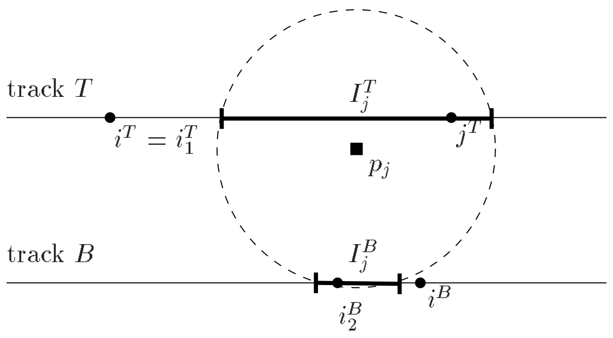

Figure 2: A counterexample to the extension of dynamic programming algorithm to the case of two (parallel) tracks.

doing this for all the intervals, T_i has the following property: For every leaf $j \in [i+1, 2n]$, the sum of the weights of the nodes in the path from j to the root is equal to $\mathsf{gain}(i,j)$. So, with a simple BFS visit we can store in every leaf j the value $\mathsf{gain}(i,j)$. Since this visit can be done in $O(n)$ time for a single T_i, then the overall time is $O(n^2)$. $\square$

We are now in a position to prove the main result.

Theorem 4 *The* MAX GAIN 1 TRACK *problem can be solved within $O(k \cdot n^2)$ time, where n is the number of settlements.*

PROOF. The proof is by induction on the number of stations we are allowed to place. For $k=1$ we simply try all the positions i and, for each of them, we compute the sum of the weights of the intervals containing i. For any $k \geq 2$, we apply Lemmas 2 and 3. In particular, by applying Lemma 3, we first compute the table $\mathsf{gain}(\cdot,\cdot)$ for all possible values. Thence, from Lemma 2, we can easily obtain $\mathsf{opt}(i;k)$ as follows: for $k' = 2,3,\ldots,k$, we compute (in this order) the table $\mathsf{opt}(i,k')$, for the possible i. Eq. 2 implies that, for each k' and i, $\mathsf{opt}(i;k')$ can be computed in $O(n)$ time. So, the overall time is $O(k \cdot n^2)$.

Finally, we observe that it is possible, during these computations, to keep track of the optimal recursive solutions that have been applied when using Eq. 2. Those parameters are nothing but the optimal placement of the k stations. Hence the theorem follows. $\square$

3 Two Parallel Tracks

We now consider the following extension of MAX GAIN 1 TRACK: instead of one track, we are given *two parallel tracks* (segments) and we have to place k stations on them. Then, the natural extension of the dynamic programming algorithm for the single track problem is as follows. We consider two parameters i^T and i^B which are the positions of the rightmost station on the top and on the bottom track, respectively. Then, $\mathsf{opt}(i^T, i^B; k)$ is defined accordingly. Let us observe that every settlement p_i turns into a *pair* of weighted intervals I_i^T and I_i^B (with one of the two or both possibly empty). Moreover, such two intervals *cannot be considered separately*: if a station j^T intersects I_i^T and a station j^B intersects I_i^B, then the gain due to such two stations is not the sum of the weights of I_i^T and I_i^B. Indeed, those two stations are satisfying the same settlement p_i. Because of this, we have to measure the contribution of a station as (the sum of the weights of) the pairs (I_i^T, I_i^B) such that I_i^T *and* I_i^B do not contain any other station. So, we define $\mathsf{contain}(i^T, i^B, j^T)$ and $\mathsf{contain}(i^T, i^B, j^B)$ accordingly. Thence, we would like to extend Lemma 1 and prove that, for any $i_1^T \leq i^T < j^T$ and $i_2^B \leq i^B$,

$$\mathsf{contain}(i_1^T, i_2^B, j^T) \subseteq \mathsf{contain}(i^T, i^B, j^T).$$

Unfortunately, *the above statement is false.* (Fig. 2 shows a counterexample: $p_j \in \mathsf{contain}(i^T, i^B, j^T)$ but $p_j \notin \mathsf{contain}(i^T, i_2^B, j^T)$.)

3.1 Minimum Distance Between the Stations

The above example suggests a (reasonable) restriction of the problem for which, instead, the dynamic programming approach works. Assume we require the distance between any two consecutive stations in the solution to be at least $2R$. Then, in the example of Fig. 2, the station i_2^B would be on the left of I_j^B. We can prove that this is always the case.

We start with some useful observations. For any two points i and j, let $d_x(i, j)$ denote the distance between the respective projections on the x-axis. Then, the following fact holds:

Fact 5 *For any $I = (I^T, I^B)$ and for any two points i and j on the tracks, if $d_x(i, j) > 2R$ then not both i and j intersect I.*

PROOF. By contradiction, if $d_x(i, j) > 2R$, then the longest among I^T and I^B has length greater than $2R$. □

Lemma 6 *For any $i_1^T \leq i^T < j^T$, and $i_2^B \leq i^B$, such that $d(i_1^T, i_1^T) > R$ and $d(i_2^B, i^B) > R$, the following holds:*

$$\mathsf{contain}(i_1^T, i^B, j^T) \cup \mathsf{contain}(i^T, i_2^B, j^T) \subseteq \mathsf{contain}(i^T, i^B, j^T). \tag{3}$$

Moreover, the same holds by considering some $j^B > i^B$ in place of j^T.

The above lemma easily implies that the dynamic programming algorithm for the single track can be extended for two parallel tracks if we impose this restriction on the minimum distance between two consecutive stations (we denote such a restriction as MAX GAIN 1-ST). The following problem is a generalization of MAX GAIN 1-ST:

> MAX GAIN r-ST: We require any solution $S = \{s_1^T, \ldots, s_l^T, s_1^B, \ldots, s_m^B\}$ to satisfy $d(s_i^T, s_{i+1}^T) \geq 2R/r$ and $d(s_j^B, s_{j+1}^B) \geq 2R/r$, for any $1 \leq i \leq l-1$ and $1 \leq j \leq m-1$.

Also MAX GAIN r-ST admits a polynomial-time exact algorithm:

Theorem 7 *The* MAX GAIN r-ST *problem can be solved within $O(n^{2r+2})$-time.*

PROOF SKETCH. We generalize the dynamic programming as follows. Instead of looking at the rightmost station on each track, we consider the r rightmost stations on each track. Hence, the table $\mathsf{opt}(\cdot, \cdot; k)$ now contains two vectors $\vec{i}^T = (i_1^T, i_2^T, \ldots, i_r^T)$ and $\vec{i}^B = (i_1^B, i_2^B, \ldots, i_r^B)$. We want to show that these sequences are all we need to compute optimal solutions. The crucial property here will be the following: for any i^T on the left of i_1^T, $d(i^T, i_r^T) > 2R$. Roughly speaking, this implies that i^T is too far from i_r^T and hence not relevant. Indeed, for any (new) station j on the right of both i_r^T and i_r^B, with j being on any of the two tracks, it holds that $d_x(i^T, j) > d(i_1^T, i_r^T) \geq 2R$. Hence, Fact 5 implies that any $I = (I^T, I^B)$ intersected by j does not intersect i^T. The same holds if we consider some i^B, instead of i^T. This implies that Lemma 6 generalizes by considering vectors $\vec{i_1}^T$, $\vec{i_2}^B$, $\vec{i}^T$ and $\vec{i}^B$ (we also say that $\vec{i} \leq \vec{j}$ if and only if no element of $\vec{i}$ is on the right of some element in $\vec{j}$). Therefore, we can show that

$$\mathsf{opt}(\vec{i}^T, \vec{i}^B; k) = \max \left\{ \begin{array}{l} \max_{\vec{i'}^T \leq \vec{i}^T} \left\{ \mathsf{opt}(\vec{i'}^T, \vec{i}^B; k-r) + \mathsf{gain}(\vec{i'}^T, \vec{i}^B, \vec{i}^T) \right\}, \\ \max_{\vec{i'}^B \leq \vec{i}^B} \left\{ \mathsf{opt}(\vec{i}^T, \vec{i'}^B; k-r) + \mathsf{gain}(\vec{i}^T, \vec{i'}^B, \vec{i}^T) \right\}. \end{array} \right\} \tag{4}$$

(The proof of this equation essentially mimics that of Lemma 2.) □

3.2 Settlements in Between the Tracks

We now consider the following problem:

MAX GAIN INNER: We consider instances in which all the settlements are located in between the two parallel tracks.

The remaining of this section is devoted to the proof of the following result:

Theorem 8 *The* MAX GAIN INNER *problem can be solved within $O(n^6)$ time, where n is the number of settlements.*

We will provide a polynomial-time algorithm for the MAX GAIN INNER as follows: (i) we first restrict to solutions that have a particular structure and show that the optimum can be found in polynomial time via (a variant of) our dynamic programming used for MAX GAIN r-ST; (ii) then, we show that every instance of MAX GAIN INNER has an optimal solution with the same structure. We begin with some definitions (see Fig. 3):

Definition 9 *For a pair $I = (I^T, I^B)$, we denote by I^{long} (resp., I^{short}) the longer (resp., shorter) between I^T and I^B. Moreover, for every interval I^*, I^*_{right} denotes its right half. We say that an interval is* long *if it is the longer in its pair.*

Definition 10 (Essential solution) *Let $I^{long}_{right} \ominus I^{short}_{right}$ denote the interval obtained by the projection of I^{long}_{right} on the other track minus I^{short}_{right}. A station placement is* essential *if, for every pair $I = (I^{long}, I^{short})$, if I^{long}_{right} contains a station, then no two stations fall in $I^{long}_{right} \ominus I^{short}_{right}$.*

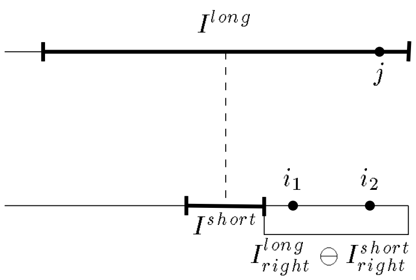

Figure 3: A non essential solution.

We first show that optimal essential solutions are computable in polynomial time.

Lemma 11 *The optimal essential solution can be computed in $O(n^6)$ time.*

PROOF. For each track we consider the *two rightmost* stations, provided that their position satisfies Def. 10. Let $\vec{i}^T = (i^T_1, i^T_2)$, and $\vec{i}^B = (i^B_1, i^B_2)$ denote such stations. Also let $\vec{i'}^T = (i'_1, i^T_2)$ where $i'_1 \leq i^T_1$. (Similarly, we define $\vec{i''}^T$ with respect to i^T_2.) Def. 10 easily implies the following fact:

$$\mathsf{contain}(\vec{i'}^T, \vec{i}^B, j^T) \cup \mathsf{contain}(\vec{i}^T, \vec{i''}^B, j^T) \subseteq \mathsf{contain}(\vec{i}^T, \vec{i}^B, j^T), \tag{5}$$

where $\mathsf{contain}(\vec{i}, j)$ denotes those intervals intersecting j and not intersecting any of the stations in $\vec{i}$. Notice that the above inclusion is similar to that of Lemma 6, and it also holds by considering some $j^B > i^B$ in place of j^T. This guarantees that the two rightmost stations are all we need to compute the contribution of a new station j^T (or j^B) when added to a partial essential solution. Therefore we can define a function $\mathsf{gain}(\vec{i}^T, \vec{i}^B, j)$ and solve the problem in a similar way of MAX GAIN r-ST for $r = 2$. □

The next technical lemma will be used to prove the optimality of the essential solutions vs. more general ones.

Lemma 12 *Let $I = (I^T, I^B)$ and $J = (J^T, J^B)$ be two pairs of intervals such that I^{long} and J^{long} lie on the same track. Also, let $J^{short} \subseteq I^{long}_{right} \ominus I^{short}_{right}$. Then, $I^{long}_{right} \subseteq J^{long}$.*

PROOF. Without loss of generality, let us assume that $R = 1$ and let us consider a settlement p corresponding to $I = (I^T, I^B)$. Let its intersections with the two tracks be as shown in Fig. 4. In particular, let $I^T = I^{long}$ and let b and c be the right endpoints of I^T and I^B, respectively. By construction, $d(p, b) = d(p, c) = 1$. Also, let d be the projection of b on the bottom track. We also define p' and p'' as shown in Fig. 4, that is, p' lies below b and has the same y-coordinate as p, and the segment $\overline{p''d}$ is parallel to $\overline{ap'}$. Therefore, $d(p', a) = d(p'', d) = 1$.

Let q be the settlement corresponding to $J = (J^T, J^B)$. Since $J^{short} \subseteq I^{long}_{right} \ominus I^{short}_{right}$, it holds that $d(q, c) \geq 1$, $d(q, d) \geq 1$ and q is inside the rectangle determined by a, b, d and e. We will prove the lemma using the following two facts:

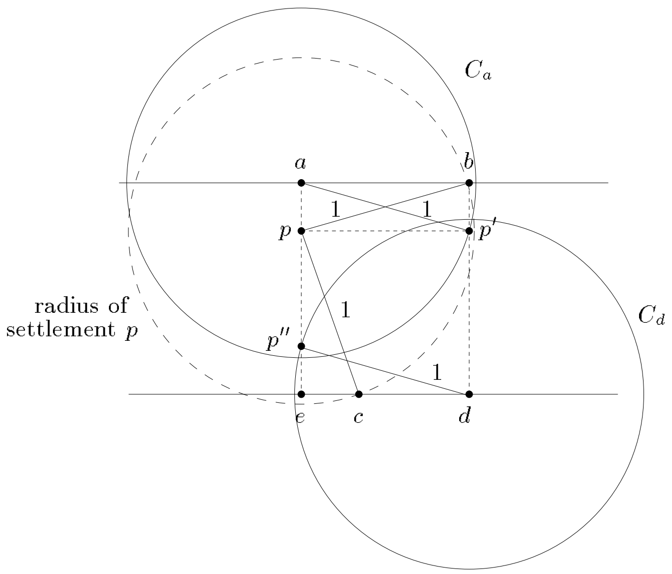

Figure 4: The proof of Lemma 12.

1. $d(d,q) \geq 1 \Rightarrow d(q,a) < 1$. Consider the two unit-radius circles C_d and C_a of center d and a, respectively (see Fig 4) . We will show that such circles intersect outside the rectangle determined by a, b, d, and e. Therefore, if $d(d,q) > 1$, then q must be properly contained in C_a, thus implying $d(q,a) < 1$.
   First, consider the intersection between C_d and C_a with larger x-coordinate. By construction, p' lies on C_a and, since $d(p,c) = 1$, we have that $d(p',d) = d(p,e) < 1$. This implies that such an intersection must be on the right of $\overline{bd}$. By symmetry, the other intersection must be on the left of $\overline{ae}$.
2. $d(c,q) \geq 1 \Rightarrow d(q,b) < 1$. Let us consider the circles C_e and C_b. By symmetry, their intersections lie outside the rectangle determined by a, b, d and e. Since c is on the right of e, the right intersection between C_c and C_b is on the right of $\overline{bd}$ as well. Finally, the left intersection between C_c and C_b is the point p. Since $p \neq q$, also the proof of this case follows.

We thus have that $d(q,a) < 1$ and $d(q,b) < 1$. Since a and b are the endpoints of I^{long}_{right}, the lemma follows. □

Lemma 13 *Every instance of* Max Gain Inner *has an optimal solution which is also essential.*

Proof. We show that any solution not satisfying Def. 10 can be transformed into an essential one that covers the same set of intervals. Let I be an interval pair for which Def. 10 is violated and let i_1 and $i_2 > i_1$ be two stations both in $I^{long} \ominus I^{short}$. We will show that moving i_1 on the right endpoint i' of I^{short} yields a (essential) solution that covers the same set of interval pairs. Let J be an interval pair covered by i_1 but not covered by i_2 nor by i'. If $i_1 \in J^{short}$, then we can apply Lemma 12 and conclude that J^{long} must contain the station in I^{long}_{right}. Otherwise, it must be the case that J^{long} is on the same track of I^{short}. It is then easy to see that Lemma 12 implies that J^{long} covers I^{short}_{right} (simply consider the interval J' obtained by exchanging J^{short} with J^{long}). Therefore, we have $i_2 \in I^{short}_{right} \subseteq J^{long}$. In both cases, the interval J is still covered in the transformed solution. □

By putting together Lemmata 11 and 13 we obtain Theorem 8.

4 Building Tracks and Stations

Here we consider the following problem: we are allowed to build a new track and to place new stations on it. (Again, we are given a maximum budget for the new track and stations.) Rather than solving the whole realistic problem, we aim at showing some interesting cases in which an efficient algorithm for the Max Gain problem immediately translates into an efficient algorithm for this track-station placement problem.

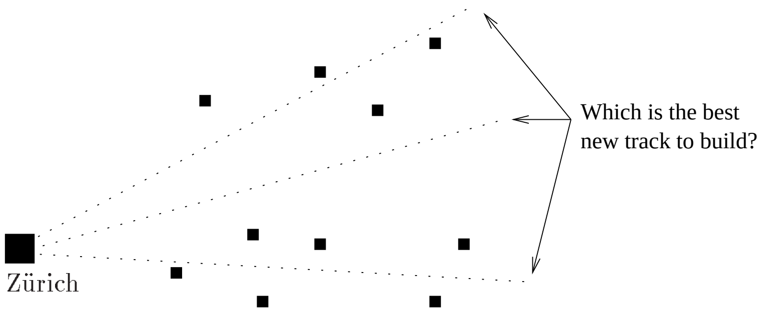

Figure 5: Placing a new track to connect settlements to a common city (dotted lines represent possible new tracks).

As an illustrative example, consider the one in Fig. 5: A big city like Zürich is surrounded by several suburbs and people living there are used to go to Zürich to work. However, no existing railway is given or the existing tracks are too far from such settlements.[2] So, a new track plus new stations should be built. Clearly, the way we locate this track will affect the second phase, that is, the station placement. Moreover, the optimal track may not pass through any of the settlements.

In the sequel we show how to optimally decide the location of a new track: we are allowed to build a straight line track (no matter how long) and to place at most k stations on it so as to maximize the gain (defined in the usual way). We denote this problem as Max Track Gain.

We first show that, given n settlements, there are at most $O(n^4)$ lines to be considered. One among those gives an optimal solution. The main idea is the following: given a straight line track, what really matters is the underlying weighted interval graph. So, changing the inclination of this line while keeping the same interval graph (i.e. only the width of the intervals change but not the way they intersect each other) does not really change the solution. On the other hand, the interval graph changes whenever (i) the line crosses the border of some settlement radius or (ii) it crosses the intersection of two radius borders. Therefore, for every two settlements p_i, p_j, we consider the (at most) four possible lines that are tangent to both such radii, plus the two intersections of their borders (actually, only one of the two cases happens depending on whether the radii intersect or not). In total, there are at most $O(n^4)$ such lines. We claim that one of those is optimal. Indeed, given any line, we can perform the following transformation: fix the left endpoint and change its inclination until, in the interval graph, either *one interval* or the intersection of *two intervals* becomes a point p. Then, fix such a point and perform the same transformation as before. At this point the line passes through two of the $O(n^2)$ points as defined above (is tangent to R_i and R_j) and its interval graph did not change. We thus obtain the following result:

Theorem 14 *Let $t(n)$ be the running time of an algorithm for the* Max Gain *problem. Then the* Max Track Gain *problem can be solved within $O(n^4 t(n))$ time.*

Note that for problems like the one in Fig. 5 the running time is $O(n^2 t(n))$ since one endpoint of the line is fixed and we only have to consider lines passing through some of the $O(n^2)$ points as defined above.

5 Existing Streets: Even a Single Track is Hard

We consider the following generalizations of Min Station/Max Gain: every settlement is connected to the track by means of a certain number of streets. Moreover, whenever two streets leading to different settlements intersect, one street passes over the other by using a bridge. So, people traveling on one street cannot switch to another one[3]. We prove that this variant of Min Station, denoted as Generalized Min Station, is hard to approximate within $c \ln n$, for some $c > 0$. Although this problem looks quite unnatural, it gives us a strong indication: even if we only want to have good (i.e. constant ratio) approximate solutions, we have to take into account some "geometry" of the streets, in particular how they intersect. Since this

[2] Actually, referring to Zürich as a city without a railway system already makes the example quite artificial.

[3] Or at least we can assume the majority of the people not able to jump from a bridge.

result also implies the NP-hardness of Generalized Max Gain, similar geometric properties should be considered (at least) in deriving polynomial-time exact algorithms.

Our hardness proof is an adaptation of the NP-hardness proof given in [11] for the k-facility location problem restricted to bimodal matrices. In particular, we make use of the following geometric problem:

Min Point Cover
Instance: A segment L of length n and a set of m tuples $t_1, \ldots, t_m$ of integer points on L.
Solution: A subset P of integer points on L such that, for every $1 \le i \le m$, $t_i \cap P \neq \emptyset$.
Measure: The cardinality of P.

The first (easy) step is to show that the above problem is equivalent to Min Hitting Set [8]. This can be done by considering any collection C of subsets of a set S as a set of tuples of integer points on a line of length $|S|$. Thus, the following theorem trivially follows:

Theorem 15 *The* Min Hitting Set *problem is* AP*-reducible*[4] *to* Min Point Cover*.*

The second step is then to prove the following:

Theorem 16 *The* Min Point Cover *problem is* AP*-reducible to* Generalized Min Station*.*

Proof Sketch. Let $(L, \{t_1, \ldots, t_m\})$ be an instance of Min Point Cover. Basically, we have to show that it is possible to place m points $\{p_1, \ldots, p_m\}$ on the Euclidean plane so that, for each of those points, (a) we can draw a set c_i of connections (i.e. polylines) from p_i to those integer points of L corresponding to the tuple t_i; (b) all the connections (among all i) have the same length R. So, this construction imposes every settlement p_i to reach the track L at points $t_i = \{a_1, \ldots, a_{k_i}\}$ using one the "streets" in c_i. Moreover, since the distance covered is constant and is equal to R, a railway station on L can serve p_i if and only if it is located on one among the positions $\{a_1, \ldots, a_{k_i}\}$. □

From the non approximability results of Min Hitting Set [18, 3] and from our reductions we get the following result:

Corollary 17 *It is* NP*-hard to approximate* Generalized Min Station *within* $c \ln n$*, for some* $c > 0$*.*

Since any algorithm for Generalized Max Gain can be used to solve Generalized Min Station (the argument is the same used in Sect. 1), we get the following result:

Corollary 18 *The* Generalized Max Gain *problem is* NP*-hard, even when all settlements have the same demand.*

Finally, the reduction of Theorem 16 implies the same hardness results even if we assume the existence of a (common) street $\bar{l}$ parallel to the track: people from p_i first take some street from p_i leading to $\bar{l}$ and then walk (i.e. move along the track) up to some distance R. However, in the instances of the reduction $\bar{l}$ is simply useless since people arrive at $\bar{l}$ already "exhausted", that is, after having covered the maximum distance R.

6 Open Problems

We mention here some problems whose solution does not seem to follow straightforward from our results:

- Consider non-Euclidean distances (e.g., existing streets connecting settlements to the track might be considered). For the simple case of one street leading to the track per settlement, the problem can be solved in the same way: every p_i still correspond to one weighted interval on the track; the length of such an interval depends on the "cost" of traveling through that street. On the other hand, if we make no assumption on the way settlements are connected to the track, then the problem becomes NP-hard, or even worse if we look at the extension of Min Station (see Sect. 5).
- Consider the variant of the Max Track Gain problem in which the new track(s) must connect *two* cities: Which is the best placement of a new track between, let us say, Konstanz and Zürich if we consider polylines, or even 1-bend segments?
- The complexity of some of the above extensions is not polynomial if the budget value B or the number t of parallel tracks is not bounded. Are those problems NP-hard for some values of B and t?

[4] See [2] for the definition of approximation preserving reduction (AP-reduction).

- As a cost function of a set of new stations, consider how much a train is slowed down, with respect to the previous configuration without such stations. Clearly, the speed of the train will depend on the relative position of the new stations and, in general, cannot be expressed as the sum of costs b_i of building a new station in position i.

Acknowledgments. We are grateful to Anita Schöbel and Dorothea Wagner for many useful discussions on the model and on the issue of unimodular matrices, and for providing us with a copy of [10, 20]. We thank Arie Tamir for poiting out a connection between our problem on the single track and [11], and the improvement in [22]. We also wish to thank Jörg Jermann for showing us the system developed for SBB [15].

References

[1] S. Arora, P. Raghavan, and S. Rao. Polynomial Time Approximation Schemes for Euclidean k-medians and related problems. In *Proc. of the 13th ACM STOC*, pages 106–113, 1998.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties.* Springer Verlag, 1999.

[3] G. Ausiello, A. D'Atri, and M. Protasi. Structure preserving reductions among convex optimization problems. *J. Comput. System Sci.*, 21:136–153, 1980.

[4] M. Charikar, S. Guha, É. Tardos, and D.B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proc of the 31st ACM Symposium on Theory of Computing*, pages 1–10, 1999.

[5] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms.* McGraw-Hill, 1989.

[6] M.S. Daskin. *Network and Discrete Location: Models Algorithms, and Applications.* Wiley-Interscience series in Discrete Mathematics and Optimization. John Wiley and Sons, Inc, 1995.

[7] M. Galota, C. Glasser, S. Reith, and H. Vollmer. A Polynomial-Time Approximation Scheme for Base Station Positioning in UMTS Networks. In *Proc. of the 5th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, DIALM*, pages 52–59, 2001.

[8] M.R. Garey and D.S. Johnson. *Computers and Intractability/ A Guide to the theory of NP-completeness.* Freeman and Company, 1979.

[9] S. Gupta and S. Kuller. Greedy strikes back: improved facility location algorithms. In *Proc. of 9th ACM-SIAM Symposium on Descrete Algorithms, SODA*, pages 649–657, 1998.

[10] H.W. Hamacher, A. Liebers, A. Schöbel, D. Wagner, and F. Wagner. Locating new stops in a railway network. In *Proc. of the ATMOS workshop, Algorithmic Methods and Models for Optimization of Railways*, Electronic Notes in Theoretical Computer Science, pages 15–25. Elsevier Science, 2001.

[11] R. Hassin and A. Tamir. Improved complexity bounds for location problems on the real line. *Operations Research Letters*, 10:395–402, 1991.

[12] D.S. Hochbaum and W. Maass. Fast approximation algorithms for a nonconvex covering problem. *Journal of Algorithms*, 8:305–323, 1987.

[13] V.N. Hsu, T.J. Lowe, and A. Tamir. Structured p-facility location problems on the line solvable in polynomial time. *Operations Research Letters*, 21:159–164, 1997.

[14] K. Jain and V. V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *IEEE Symposium on Foundations of Computer Science*, pages 2–13, 1999.

[15] J. Jermann. Private communication. A decision support tool for placing new stations in the SBB railway network.

[16] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, 1988.

[17] C. H. Papadimitriou. Worst-case and probabilistic analysis of a geometric location problem. *SIAM J. Comput.*, 10, 1981.

[18] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and sub-constant error-probability PCP characterization of NP. In *Proc. of the 29th ACM STOC*, pages 475–484, 1997.

[19] A. Schöbel. Set covering problems with consecutive ones property. Technical report, Universität Keiserslautern, 2001.

[20] A. Schöbel, H.W. Hamacher, A. Liebers, and D. Wagner. The continuous stop location problem in public transportation networks. *Submitted to international journal*, 2002.

[21] D.B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems. In *Proc. of the 29th ACM Symposium on Theory of Computing, STOC*, pages 265–274, 1997.

[22] A. Tamir. Personal comunication. July 2002.