LINE PLANNING, PATH CONSTRAINED NETWORK FLOW AND INAPPROXIMABILITY

CHRISTINA BÜSING AND SEBASTIAN STILLER

Abstract. We consider a basic subproblem which arises in line planning, and is of particular importance in the context of a high system load or robustness: How much can be routed maximally along all possible lines? The essence of this problem is the *Path Constrained Network Flow* (PCN) problem. We explore the complexity of this problem and its dual. In particular we show for the primal that it is as hard to approximate as MAX CLIQUE and for the dual that it is as hard to approximate as SET COVER. We also prove that the PCN problem is hard for special graph classes, interesting both from a complexity and from a practical perspective. Finally, we present a special graph class for which there is a polynomial-time algorithm.

Keywords: Line Planning, Network Flows, Complexity, Robustness, Planar Graphs, Bounded Treewidth

1. Introduction

Motivation. A classical step in the hierarchy of planning scheduled transportation networks is *line planning*. Its task is to determine the lines and their frequencies along which the transportation service shall be offered. Given a network, i.e., a directed graph with upper capacities on the arcs, a *line plan* is a set of lines, i.e., paths in the network, and an integer assigned to each line, its *frequency*. A line plan is *feasible*, if it satisfies a given transportation demand and respects the upper arc-capacities of the underlying network. Moreover, the lines that can be used, i.e., the trips a physical train and its crew can serve, are subject to various regulations such as length bounds, limits on the number of terminals and several other, partly very specific requirements. Therefore, line planning often relies on a so-called *line*

E-mail: cbuesing@math.tu-berlin.de, stiller@math.tu-berlin.de

This work has partially been supported by the ARRIVAL project, within the 6th Framework Program of the European Commission under contract no. FP6-021235-2, the DFG Research Training Group, Methods for Discrete Structures, and the Berlin Mathematical School.

pool, i.e., an explicitly given set of possible paths to which the line plan is restricted.

The objective in line planning is usually some minimization of operative cost or cost of passenger's discomfort. Special events, like world cups, fairs, concerts, or temporary exhibitions could create exceptional demand for which special line plans have to be designed. In other cases, construction or maintenance works create exceptionally low upper capacities in the network. Again special line plans are required. Finally, even daily, high utilization of a scarce infrastructure may demand for the construction of a line plan that makes the system work to its capacity. It is natural to ask for the maximum demand that can be met given a network, upper arc-capacities and a line pool.

This poses the Path Constrained Network Flow Problem, which has already been mentioned in [5]. We examine the complexity of this question, showing that even its single-source, single-sink version is **NP**-hard and in a strong sense inapproximable. Moreover, we study the problem on special graph classes like planar graphs, graphs with bounded treewidth, and a certain class derived from trees, for which we devise a non-trivial, polynomial-time algorithm.

Related Work. Usually an instance of a line planning problem is given by a directed graph $G = (V, A)$, with arc-capacities $c : A \to \mathbb{R}_+$, and projected demands, e.g., how many passengers intend to travel from node x to node y. Moreover, we are given explicitly or implicitly the set of possible lines, i.e., certain paths in G, the *line pool* denoted by $\mathcal{P}$. A feasible line plan is an assignment of integer frequencies f_p to the lines $p \in \mathcal{P}$, such that the capacity of each arc a is respected, i.e., $\sum_{p \in \mathcal{P}: a \in p} f_p \leq c(a)$, and the demand can be fulfilled.

Whether the demand, i.e., a certain amount of transportation for each origin-destination pair, can be fulfilled by a *fixed* line plan is itself a non-trivial problem. The passengers have to be routed along the chosen lines respecting the capacities on their path offered by the line plan. In practice, the routing constructed by a central optimization may not coincide with the routing actually chosen by the individual passengers. Hereby, violations of capacities may arise in a real, individual routing even though a feasible, central routing exists.

There are different approaches in the literature to address this issue. For example, in the work of Bussieck et al. [2] a system split in optimization is used. Prior to constructing the line plan the passengers are routed through the network G according to their origin and destination, and some assumptions on their routing behavior. This step entails a (fractional) multi-commodity flow problem. The total resulting flow on an arc a is interpreted as its *lower arc capacity* $\ell(a)$. Then an instance of a line planning problem is a quintuple $(G, c, \ell, \mathcal{P}, \mathrm{cst})$ consisting of a network, upper and lower capacities on each arc, a line pool, and some cost function cst. The optimization problem can be expressed as:

$$\begin{aligned} &\min \mathrm{cst}(f) \\ \ell(a) \leq &\sum_{p \in \mathcal{P}: a \in p} f_p \leq c(a) \quad \forall a \in A \\ &f \geq 0 \end{aligned}$$

For f restricted to integer vectors, this is known to be NP-hard [2].

A different way to define the demand satisfaction is pursued by [1, 12]. Here the multi-commodity flow problem of routing the passengers is part of the line planning optimization procedure. The articles [1, 2] contain a comprehensive overview on the literature for their specific approaches. Any approach with implicit or explicit reference to a line pool has the path constrained network flow as a subproblem that becomes crucial when the system works to capacity.

Our question is basic to both approaches, and crucial whenever upper capacities become tight.

The problem we consider is a slight specialization of the Path Constrained Network Flow (PCN) problem defined already in Garey and Johnson ([5], Problem ND34). They stated that the PCN problem is—among some other complexity features—**NP**-complete. Our hardness and inapproximability results carry over to the original PCN as we consider a specialization.

For the proof of hardness Garey and Johnson refer to a private communication with H.J. Prömel. To our knowledge no proof has ever been published and Prömel himself in a further private communication with the authors kindly stated that he was no longer in possession of one. The proofs we give recover all results stated in [5] and add several complexity results, in particular the *inapproximability* of the optimization problem. Moreover, we add results for special graph classes and study the dual problem.

The (s,t)-PCN problem on (s,t)-extended trees is closely related to the maximum multi-commodity flow problem in trees. Garg et al. [6] show that the problem is **NP**-hard for capacities of 1 and 2.

Definition of the PCN Problem. In the classical flow theory, flow is sent in a network from source to sink respecting arc capacities. In the 60s Ford and Fulkerson showed that any flow can be decomposed into path flows and cycle flows [4]. In the general case the set of paths used for the decomposition is not restricted. In our case we are given a set of paths $\mathcal{P}$. We are now interested in those flows that can be decomposed using only the paths in $\mathcal{P}$. We give the formal definition of the PCN problem according to Garey and Johnson [5]:

Definition 1.1. [Path Constrained Network Flow Problem] Let $G = (V, A)$ be a directed graph with capacities $c(a) \in \mathbb{N}$ for each $a \in A$, a source $s \in V$ and a sink $t \in V$. Furthermore, a collection $\mathcal{P}$ of directed paths in G and a requirement $K \in \mathbb{N}$ is given. The set of paths $\mathcal{P}_a \subset \mathcal{P}$ is the set of all paths in $\mathcal{P}$ containing the arc a. The question is whether there is a function $y : \mathcal{P} \to \mathbb{N}$ such that for the flow function $f : A \to \mathbb{N}$ defined by $f(a) = \sum_{p \in \mathcal{P}_a} y(p)$, the following three conditions hold:

(1) $f(a) \leq c(a)$ for all $a \in A$,
(2) for each $v \in V \backslash \{s,t\}$, flow is conserved at v, and
(3) the net flow into t is at least K.

We specialize the PCN problem twice: first to the (s,t)-PCN by restricting the set of paths $\mathcal{P}$ to a set of (s,t)-paths, and second to the (s,t)-PCN$_1$ problem by additionally demanding unit capacities on the arcs. Note that flow conservation (2) is automatically given in this case. Due to the unit capacities any solution for the (s,t)-PCN$_1$ consists of arc disjoint (s,t)-paths. Hardness results for (s,t)-PCN$_1$ hold for (s,t)-PCN and those for (s,t)-PCN carry over to PCN.

As we also consider inapproximability we have to define a PCN optimization problem. The objective is to maximize the value of the net flow K. We will refer to both problems as the PCN problem, as it will be clear from the context whether the optimization or the decision problem is meant.

The dual problem of (s,t)-PCN is defined as it arises from the dualization of the obvious IP formulation:

Definition 1.2. Let $G = (V, A)$ be a directed graph with capacities $c(a) \in \mathbb{N}$ for each $a \in A$, a source $s \in V$ and a sink $t \in V$. Furthermore, a collection $\mathcal{P}$

of directed (s,t)-paths in G and a requirement $K \in \mathbb{N}$ is given. The objective is to find a set of arcs A' with weight $\sum_{a \in A'} c(a)$ smaller than K, such that every path in $\mathcal{P}$ traverses at least one arc in A'.

The optimization version of this problem is to find a proper arc set A' with minimum weight.

Results and Techniques. We give positive and negative results for the (s,t)-PCN problem and the (s,t)-PCN$_1$ problem and their duals. Note that the hardness results carry over to the more general cases.

In detail we prove **NP**-hardness for the (s,t)-PCN$_1$ problem, and we establish **NP**-completeness for the questions whether its integrality gaps, respectively its duality gaps, are zero. We prove that the maximum value of an (s,t)-PCN$_1$ cannot be approximated within a constant factor, unless $\mathbf{P} = \mathbf{NP}$. We extend the inapproximability of the (s,t)-PCN$_1$ problem for graphs with bounded treewidth and planar graphs, and for the (s,t)-PCN problem on grid graphs.

We will present two paradigmatic reductions in detail. All hardness and inapproximability results are shown by reductions similar to those.

Our factor preserving reduction from MAX CLIQUE for the inapproximability is reversible. Therefore, any practical algorithm for MAX CLIQUE can easily be turned into an algorithm for the (s,t)-PCN$_1$ problem.

For the dual of the (s,t)-PCN$_1$ problem (and thus the less specialized dual PCN problems) we give an L-reduction from SET COVER. Under certain conditions the dual problem is easy while the primal remains inapproximable.

Finally, we devise a polynomial-time algorithm for the (s,t)-PCN$_1$ problem on (s,t)-extended outtrees, which we define in this paper. The problem on this graph class is equivalent to the transshipment problem on outtrees. Moreover, (s,t)-extended outtrees are motivated by transportation systems, which are trees except for the connection to a common depot of the vehicles.

Note that the fractional PCN problem in general is easy as it can be formulated as an LP of polynomial size.

Structure of this Paper. In Section 2 we compile the general complexity results and the paradigmatic proofs. In Section 3 and 4 we give the positive and negative results for special graph classes. In particular, we present the algorithm for (s,t)-extended outtrees.

2. Complexity and Inapproximability for Arbitrary Graphs

We start this section with the **NP**-completeness of the PCN problem. All PCN problems considered are in **NP**: Given a path multiplicity vector $y \in \mathbb{N}^{|\mathcal{P}|}$ we can calculate its value in polynomial time, construct the corresponding flow and check its feasibility. By our reductions for the PCN problem it will become clear that a certificate that would only comprise the flow without the decomposition could not in polynomial time be checked to be decomposable into paths in $\mathcal{P}$, unless $\mathbf{P} = \mathbf{NP}$.

We start by showing that the (s,t)-PCN$_1$ problem is strongly **NP**-complete by using a reduction from 3SAT.

Recall that a decision problem $\mathcal{X}$ is called strongly **NP**-hard, iff there is a polynomial p such that the restriction of $\mathcal{X}$ to the set of instances x where the largest integer in the input of x is bounded from above by $p(\text{inputsize}(x))$ is still **NP**-hard (cf. [9]). (We can assume that the input contains no numbers other than integers.) In other words, even a unary encoding of the numbers in the input would not allow for algorithms polynomial in the input size, unless $\mathbf{P} = \mathbf{NP}$.

Theorem 2.1. *The (s,t)-PCN$_1$ is strongly* **NP***-complete.*

Proof. We reduce from 3SAT. Let I be an instance of 3SAT with n variables $x_1, ..., x_n$ and m clauses $C_1, ..., C_m \subseteq \{x_1, \overline{x}_1, \ldots, x_n, \overline{x}_n\}$ with $|C_j| = 3$.

We start with the construction of G and $\mathcal{P}$. The set $\mathcal{P}$ contains for every variable x_i a path p_{x_i} and a path $p_{\overline{x}_i}$ both crossing the arc e_i. For every clause C_j there are three paths $\tilde{p}_{j,1}, \tilde{p}_{j,2}$ and $\tilde{p}_{j,3}$ in $\mathcal{P}$ traversing the arc e_{n+j} and three arcs $c_{j,1}, c_{j,2}$ and $c_{j,3}$. Each arc $c_{j,i}$ represents exactly one literal $z_{j,i}$ of C_j. The path $\tilde{p}_{j,i}$ traverses arc $c_{j,i}$. Any other path p_z with $z \in \{x_1, \overline{x}_1, ..., x_n, \overline{x}_n\}$ contains arc $c_{j,i}$ if and only if $z_{j,i} = \overline{z}$ (Fig. 2.1). To complete the construction of an (s,t)-PCN$_1$ instance we set K to $n + m$.

Note the following properties of the construction:

(1) Any feasible function y must satisfy $y(p_{x_i}) + y(p_{\overline{x}_i}) \leq 1$ and $y(\tilde{p}_{j,1}) + y(\tilde{p}_{j,2}) + y(\tilde{p}_{j,3}) \leq 1$ for capacity reasons on the arcs e_ℓ.
(2) Any feasible function y with net flow value $n+m$ must satisfy $y(p_{x_i}) + y(p_{\overline{x}_i}) = 1$ and $y(\tilde{p}_{j,1}) + y(\tilde{p}_{j,2}) + y(\tilde{p}_{j,3}) = 1$, because $\{e_1, \ldots, e_{n+m}\}$ is a cut.

Now we show that there exists a function $y : \mathcal{P} \to \mathbb{N}$ with value $n + m$ and $\sum_{p \in \mathcal{P}_a} y(p) \leq 1$ for all arcs $a \in A$ if and only if the instance I can be satisfied.

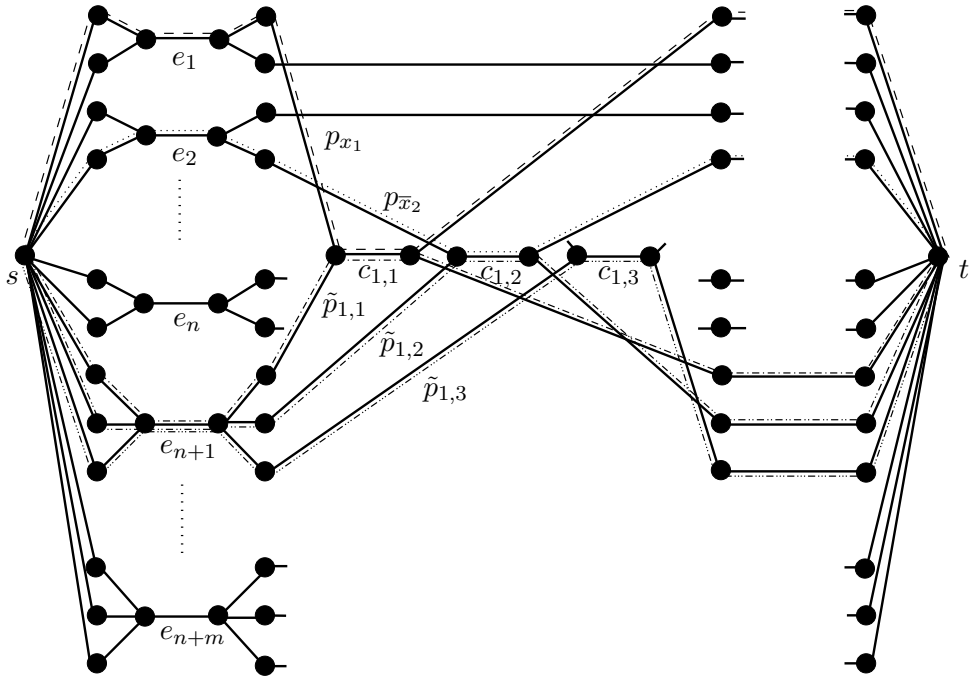

FIGURE 2.1. The set of paths $\mathcal{P}$ contains $2n + 3m$ (s, t)-paths.

Let $y : \mathcal{P} \to \mathbb{N}$ be a function with $\sum_{p \in \mathcal{P}_a} y(p) \leq 1$ and $\sum_{p \in \mathcal{P}} y(p) = n+m$. Define the vector x by

$$x_i := \begin{cases} 1 & \text{if } y(p_{x_i}) = 1 \\ 0 & \text{if } y(p_{\overline{x}_i}) = 1. \end{cases}$$

By observation (2) x is well defined.

We now show that x satisfies all clauses. Suppose there exists a clause C_j which is not satisfied. Then $y(p_{z_1}) = y(p_{z_2}) = y(p_{z_3}) = 1$ for $z_i = \overline{z}_{j,i}$ and due to capacity no other path using arc $c_{j,1}, c_{j,2}$ or $c_{j,3}$ contains any flow. Therefore, $y(\tilde{p}_{j,1}) + y(\tilde{p}_{j,2}) + y(\tilde{p}_{j,3}) = 0$, a contradiction to observation (2).

Let now x be a vector that satisfies all clauses. We compose a function $y : \mathcal{P} \to \mathbb{N}$ with $\sum_{a \in \mathcal{P}_a} y(p) \leq 1$ by

$$y(p_{x_i}) = \begin{cases} 1 & \text{if } x_i = 1 \\ 0 & \text{else} \end{cases} \quad \text{and } y(p_{\overline{x}_i}) = \begin{cases} 1 & \text{if } x_i = 0 \\ 0 & \text{else.} \end{cases}$$

For any clause there exists a literal $z_{j,i^*} = 1$. Therefore, no flow is passing through arc c_{j,i^*}. Define $y(\tilde{p}_{j,i^*}) = 1$ and $y(\tilde{p}_{j,i}) = 0$ for $i \neq i^*$. The function y therefore fulfills all capacity restrictions and has the value $n + m$. □

This also follows from Theorem 2.6. But the construction in this proof leads nicely to two further reductions concerning the dual gap and the LP-IP-relation of the PCN optimization problems.

The PCN problem (and its considered specializations) with non-integral flow is equivalent to linear programming and therefore polynomially solvable. But the question whether the best rational flow fails to exceed the best integral flow is **NP**-complete. By the construction of G and $\mathcal{P}$ in Theorem 2.1 this is easy to see, since a feasible, rational flow $y^* : \mathcal{P} \to \mathbb{Q}^{|\mathcal{P}|}$ with value $n + m$ is always given by $y^*(p_z) = \frac{1}{2}$ for all $z \in \{x_1, \overline{x}_1, \ldots, x_n, \overline{x}_n\}$ and $y^*(\tilde{p}_{j_i}) = \frac{1}{3}$ for all $j = 1, \ldots, m$ and $i = 1, 2, 3$.

Theorem 2.2. *Given an (s,t)-PCN$_1$ instance, the decision whether the best rational flow fails to exceed the best integral flow is* **NP***-complete.*

Furthermore, the decision problem whether for the (s,t)-PCN$_1$ optimization problem the dual gap equals zero is **NP**-complete. Recall that the dual (s,t)-PCN$_1$ optimization problem asks for a subset of arcs A' with minimum cardinality $|A'|$ such that every path $p \in \mathcal{P}$ traverses at least one arc of A'. For the (s,t)-PCN$_1$ instance in the proof of Theorem 2.1 the arcs $\{e_1, \ldots, e_{n+m}\}$ form an optimal dual solution A' with the value $n+m$. This observation suffices to prove the following theorem:

Theorem 2.3. *Given an (s,t)-PCN$_1$ instance, the decision whether the dual gap is equal to zero is* **NP***-complete.*

We are now interested in the complexity and approximability of the dual PCN problems. A certain type of reduction, called L-reduction, preserves approximability (whereas in general polynomial transformations do not). Recall that to establish an *L-reduction* (cf. [9]) from an optimization problem $\mathcal{X}$ to an optimization problem $\mathcal{X}'$ we have to devise a pair of functions f and g, both computable in polynomial time, and two constants $\alpha, \beta > 0$ such that for any instance x of $\mathcal{X}$:

- $f(x)$ is an instance of $\mathcal{X}'$ with $\mathrm{OPT}(f(x)) \leq \alpha \mathrm{OPT}(x)$;
- For any feasible solution y' of $f(x)$, $g(x, y')$ is a feasible solution of x such that $|c_x(g(x, y')) - \mathrm{OPT}(x)| \leq \beta |c_{f(x)}(y') - \mathrm{OPT}(f(x))|$,

where c_x is the cost function of the instance x. We now introduce an L-reduction from the SET COVER problem to the dual (s,t)-PCN problem.

Theorem 2.4. *The SET COVER problem is L-reducible to the dual (s,t)-PCN problem with constants $\alpha = \beta = 1$ and vice versa.*

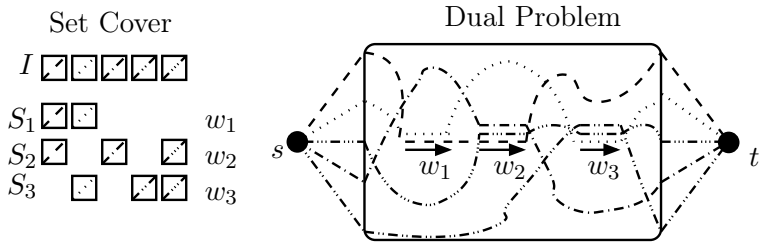

FIGURE 2.2. Every optimal solution of the dual (s,t)-PCN problem yields a solution for the SET COVER problem.

Proof. We start by defining an L-reduction from SET COVER to the dual (s,t)-PCN problem. Let I be an instance of the SET COVER problem given by a set $S = \{x_1, \ldots, x_m\}$ of m elements to be covered and a collection of sets $S_j \subseteq S$, $j \in J = \{1, \ldots, n\}$ with integral weights w_j for each $j \in J$. W.l.o.g. we assume that every element x_i is element of at least one set S_j. Therefore, we can extend the given sets S_j by the sets $\overline{S}_i = \{x_i\}$ with weights $\overline{w}_i := \max\{w_j \mid x_i \in S_j\}$ without changing the problem.

The equivalent dual (s,t)-PCN instance I' contains a graph $G = (V', A')$ with an arc a_j corresponding to the set S_j with upper capacities $c_{a_j} = w_j$ and a set of (s,t)-paths $\mathcal{P}$ which is comprised of exactly one (s,t)-path p_i for every element $x_i \in S$ (Fig. 2.2). The paths are constructed such that p_i traverses arc a_j iff $x_i \in S_j$. Apart from those arcs a_j the paths shall be disjoint. The required arcs to obtain this property get upper capacities $c_a = \overline{w}_i$ iff p_i crosses arc a. The graph G' can be constructed with at most $(n+1)\cdot m+n$ arcs and $2\cdot(n+1)$ nodes. The set of paths contains m different paths.

A feasible solution A'' of I' can be converted into a feasible solution of I by choosing S_j for every $a_j \in A''$ and $\overline{S}_i$ for every $a \in A''$ with $c_a = \overline{w}_i$. The values of the objective function for both solutions are the same.

The reduction from the dual (s,t)-PCN problem to the SET COVER problem is similar. □

Due to this reduction the **NP**-completeness and all inapproximability results for SET COVER can be transferred to the dual (s,t)-PCN problem. Recall that the SET COVER problem remains **NP**-complete for unit weights. This problem is L-equivalent to the dual (s,t)-PCN$_1$ problem. Furthermore, we get the following approximability classification:

Corollary 2.5. *If there is some $\varepsilon > 0$ such that a polynomial-time algorithm can approximate the dual (s,t)-PCN problem within $(1-\varepsilon)\ln(|\mathcal{P}|)$, then* **NP** *is contained in DTIME($n^{\mathcal{O}(\log\log(n))}$).*

The class DTIME($n^{\mathcal{O}(\log\log(n))}$) is the class of problems solvable by a deterministic algorithm in time $n^{\mathcal{O}(\log\log(n))}$. This result follows immediately from Feige's lower bound on the complexity of the SET COVER problem [3].

In the following theorem we introduce an L-reduction from MAX CLIQUE to the (s,t)-PCN$_1$ problem. In Section 3 this reduction from MAX CLIQUE will be transferred to other PCN instances with graphs of special graph classes.

Theorem 2.6. *The* MAX CLIQUE *problem is L-reducible to the (s,t)-PCN$_1$ problem with constants $\alpha = \beta = 1$ and vice versa.*

Proof. We define an L-reduction from MAX CLIQUE to (s,t)-PCN$_1$. Given a graph $G = (V,E)$ as a MAX CLIQUE instance I we want to construct an (s,t)-PCN$_1$ instance I' such that for any clique $C \subseteq V$ in G one easily obtains a set of arc disjoint (s,t)-paths with the same cardinality $|C|$ and vice versa.

To this end, the instance I' shall consist of a graph $G' = (V',E')$ and a set of (s,t)-paths $\mathcal{P}'$ such that for any $v \in V$ there exists an (s,t)-path $p_v \in \mathcal{P}'$, and we have: p_v and p_u are arc disjoint if and only if the arc (u,v) is in E. We will call a graph G' together with a set of paths $\mathcal{P}'$ that fulfills those requirements *clique-reducing*.

The construction of a clique-reducing pair $(G',\mathcal{P}')$ for a MAX CLIQUE instance G with at most n^2+2 nodes and at most $3n^2+2n$ arcs in G' is best understood as a process: Assume V to be indexed by $\{1,\dots,n\}$. We start with n parallel (disjoint) paths from s to t. We change the graph and the paths successively from 1 to n. For each $n \geq i > j \geq 1$ and $\{v_i,v_j\} \notin E$ we deviate the path p_{v_i} corresponding to node v_i such that it intersects on an exclusive arc of path p_{v_j}. Carefully inserting necessary arcs for this construction we can end up with paths of at most $3n+2$ arcs each.

Let $\overline{\mathcal{P}}$ be a subset of $\mathcal{P}'$ and $\overline{C} := \{v \in V | p_v \in \overline{\mathcal{P}}\}$. By construction $\overline{C}$ is a clique if and only if all paths in $\overline{\mathcal{P}}$ are arc disjoint. Therefore, any feasible solution of the (s,t)-PCN$_1$ instance yields a feasible solution for the MAX CLIQUE instance with the same value.

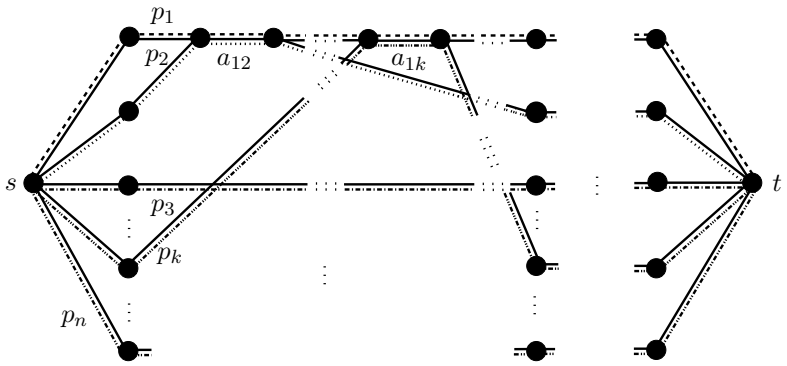

FIGURE 2.3. The (s,t)-paths p_1 and p_2, and p_1 and p_k of $\mathcal{P}'$ have a common arc, whereas p_1 and p_3 are disjoint. Correspondingly in the graph of the clique instance the arc $(1,3)$ exists, whereas $(1,2)$ and $(1,k)$ do not exist.

In a similar way an L-reduction from (s,t)-PCN$_1$ to MAX CLIQUE can be constructed. □

The complexity of MAX CLIQUE is well studied and belongs to the highest class, Class IV, of approximation problems classified in [8]. In particular, an $n^{0.5-\epsilon}$ approximation is **NP**-hard. Further MAX CLIQUE is hard to approximate within a factor $n^{1-\varepsilon}$ for any $\varepsilon > 0$, unless **NP** = **ZPP** [7]. Due to the given L-reduction these results transfer to the PCN problems.

Corollary 2.7. *For any $\epsilon > 0$ approximating the (s,t)-PCN$_1$ within a factor of $|\mathcal{P}|^{0.5-\epsilon}$ is* **NP***-hard.*

By the L-reduction from (s,t)-PCN$_1$ to MAX CLIQUE any algorithm, heuristic or algorithmic result for the MAX CLIQUE problem directly transfers to the (s,t)-PCN$_1$ problem. Note that we have no L-reduction from the (s,t)-PCN without unit capacity to MAX CLIQUE.

3. Planar Graphs and Bounded Treewidth

In many cases **NP**-complete problems can be solved efficiently on special graph classes. We studied the PCN problem on graphs with bounded treewidth and planar graphs. In both cases the complexity of the problems remained the same.

In the 80s Robertson and Seymour introduced the notion of treewidth for undirected graphs [11].

Definition 3.1. A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(\{X_i | i \in I\}, T = (I, F))$ with $\{X_i | i \in I\}$ a family of subsets of V, one for each node of T, and T a tree such that

- $\bigcup_{i \in I} X_i = V$,
- for all arcs $(v, w) \in E$, there exists an $i \in I$ with $v \in X_i$ and $w \in X_i$,
- for all $i, j, k \in I$: if j is on the path from i to k in T, then $X_i \cap X_k \subset X_j$.

The *treewidth of a tree-decomposition* $(\{X_i | i \in I\}, T = (I, F))$ is $\max_{i \in I} |X_i| - 1$. The *treewidth of a graph* G is the minimum treewidth over all possible tree-decompositions of G. The *treewidth of a directed graph* equals that of its underlying undirected graph.

For many **NP**-complete problems on graphs there exists a polynomial, often even a linear algorithm on graphs with bounded treewidth. For the PCN problem this is not the case. The clique-reducible pair $(G', \mathcal{P}')$ in the reduction of the proof of Theorem 2.6 can be constructed such that G' is a chain graph: A *chain graph* consists of n nodes $v_1, \ldots, v_n$ and parallel arcs between two nodes v_i and v_{i+1}. A simple tree-decomposition of a chain graph is defined by $X_i := \{v_i, v_{i+1}\}$ for $i \in I = \{1, ..., n-1\}$ and the path $T = (I, E)$ with arcs $(i, i+1) \in E$. The treewidth of this decomposition is 1. Due to the parallel arcs we can construct an (s, t)-PCN$_1$ instance I' on a chain graph for any MAX CLIQUE instance I such that two paths p_u and p_v are arc disjoint if and only if the nodes u and v are connected by an arc in I (Fig. 3.1). Replacing each parallel arc by a path of length 2 gives a simple graph, that serves the same purpose except that the treewidth increases by 1.

Theorem 3.2. *The MAX CLIQUE problem is L-reducible to the (s, t)-PCN$_1$ problem on graphs with bounded treewidth.*

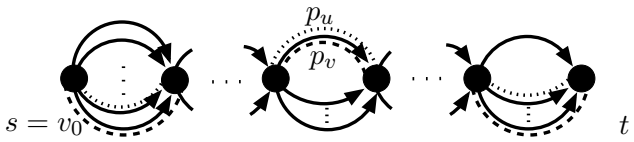

FIGURE 3.1. The L-reduction in Theorem 2.6 can be transferred to chain graphs.

The construction of the proof yields inapproximability already for the class of instances, where the graph is a chain graph and the path set has the following, strong property: Each arc is crossed by at most two paths.

Observe that in general the dual of the (s,t)-PCN$_1$ problem is polynomial solvable by a maximum matching, if the latter property is fulfilled.

Since chain graphs are planar the result holds for this graph class, too.

Corollary 3.3. *The (s,t)-PCN$_1$ problem on planar graphs is* **NP***-complete.*

An interesting class of graphs for line planning are grid graphs. The street network of cities like Manhattan or Mannheim are based on this structure. Moreover, the class of grid graphs has unbounded treewidth. But even on this class of well structured graphs the (s,t)-PCN problem cannot be approximated within a constant factor, unless $\mathbf{P} = \mathbf{NP}$.

Theorem 3.4. *The MAX CLIQUE problem is L-reducible to the (s,t)-PCN problem on grid graphs.*

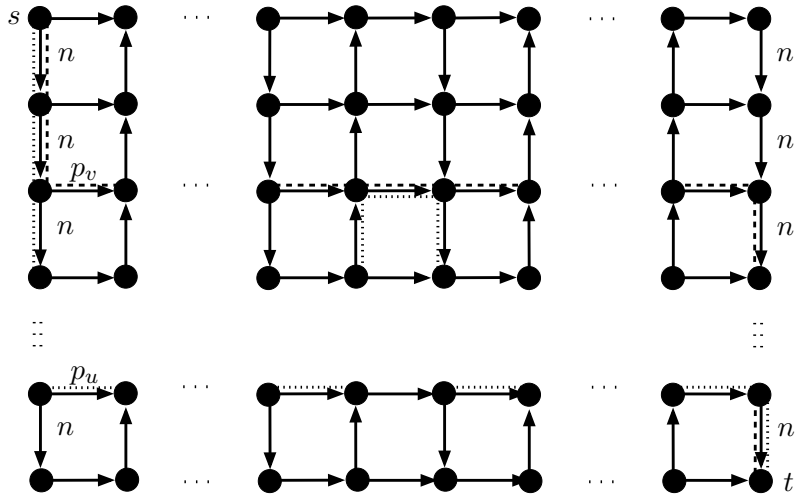

FIGURE 3.2. The paths p_u and p_v have a common arc if and only if (u,v) is not in the graph of the clique instance.

The L-reduction is again based on the construction of Theorem 2.6 (Fig. 3.2). It is possible to construct the clique-reducible pair, such that the graph is a grid. Yet, in this case it is essential to use arbitrary capacities. In fact we show in the next section that there is a polynomial-time algorithm for the (s,t)-PCN$_1$ problem in grids.

4. Tractable Graph Classes

So far we considered only classes of graphs on which the PCN problems remain difficult. In this section we consider two classes, grid graphs and tree like graphs, on which at least the (s,t)-PCN$_1$ problem is solvable in polynomial time.

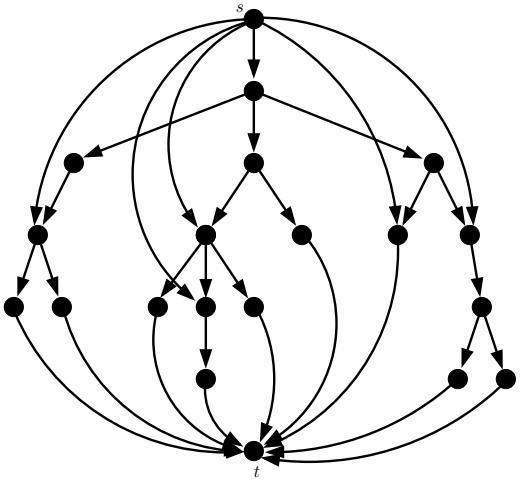

FIGURE 4.1. An (s,t)-extended outtree.

The (s,t)-PCN$_1$ on Grids. We have seen before that the (s,t)-PCN problem on grid graphs with arbitrary capacities c is **NP**-hard. In the case of unit capacities we can exploit the property of an optimal solution to consist of arc disjoint paths. Since the degree of s is at most 4 any optimal solution consists of at most 4 different paths. Checking the feasibility of any subset of the set of paths $\mathcal{P}$ with at most 4 elements is possible in polynomial time. In that way we can compute the optimal solution by enumeration. Since the only feature of grids we use is the constant bound on the degree of s (respectively t), the idea extends to any class of (s,t)-PCN$_1$ problems for which the minimum of the degrees of s and t is a constant.

The (s,t)-PCN on Tree-Like Graphs. On trees the (s,t)-PCN problem is trivial since there exists only one path from s to t. If we set $y(p) := \min_{a\in p} c(a)$ with $p \in \mathcal{P}$ we obtain the optimal solution. A natural extension is to consider transshipment on directed trees, which is equivalent to the (s,t)-PCN on (s,t)-extended outtrees: A graph $G = (V,A)$ is an (s,t)-*extended outtree* if $G - \{s,t\}$ is an outtree (Fig. 4.1). An *outtree* is a directed tree $T = (V',A')$ with a root node $r \in V'$ such that there is a directed path (not necessarily in $\mathcal{P}$) from r to any other node of V'.

In an (s,t)-extended outtree $G = (V,A)$ we denote the set of nodes incident to s by V_s, those incident to t by V_t and the child nodes of a node v by C_v. The nodes of the outtree $T = G - \{s,t\}$ can be partitioned into disjoint sets, called level L_i, according to their distance i to r. The level with largest distance to r shall be L_k. An (s,t)-path p in an (s,t)-extended outtree G is

uniquely defined by a pair of nodes (a, b) with $a \in V_s$ and $b \in V_t$, thus we will denote this path by $[a, b]$.

W.l.o.g. we assume for the remainder of this section that every arc is traversed by at least one path of $\mathcal{P}$. Therefore, t has to be connected to all leaves. Further, we can assume that only leaves are connected to t. In case some other node is connected to t, we add a dummy leaf as a child of this node in which we switch the connection to t. The path set is adjusted in the obvious way. Similarly we avoid $v \in V_s \cap V_t$ by adding a dummy node after v and changing the path set. Solving the problem on this new graph is apparently equivalent to solving the original problem. Therefore, we tacitly assume those two conditions in the remainder.

Theorem 4.1. *The (s,t)-PCN$_1$ problem on (s,t)-extended outtrees is solvable in polynomial time.*

Proof. The polynomial running time of Algorithm 1 is obvious. To prove Theorem 4.1 it remains to show feasibility and optimality of the path set $\mathcal{P}^*$ returned by Algorithm 1. The feasibility is immediate from the construction once its correctness is established. For the latter it suffices to note that when the algorithm is in one of the two cases POSTPONED and FIXED the set $\mathcal{P}'$ contains the path required in the algorithm. For the FIXED case this is again immediate from construction. For POSTPONED, because G is an outtree, in any feasible set of paths a node $v \in T$ can be used by at most one path that entered the tree on a higher level than v. When the POSTPONED case is invoked for some node v, only paths entering on higher or equal levels were chosen so far. By induction we can assume this prior chosen path set to be feasible. As the paths in $\mathcal{P}'$ containing (s, v) use at least two different children of v, not all of them can be in conflict with the prior chosen set of paths.

For the optimality we have to show that the set of arcs in A^* blocks all paths in $\mathcal{P}$. To this end observe: Every path $p \in \mathcal{P}$ starts with an arc of the form $a = (s, v)$. In case $a \in A^*$, we are done. In case $a \notin A^*$, consider the step when the algorithm scanned node v. Either $p \in \mathcal{P}'$ at this time or not. In the latter case, p was removed when a node v' on a higher level was scanned and the FIX case occurred. In this situation the algorithm added an arc $(v', u) \in p$ to A', i.e., A^*. Thus A^* blocks p. In the other case, namely, $p \in \mathcal{P}'$ when v was scanned, we have $\Phi(v) = 1$, because for $\Phi(v) > 1$ the arc

Algorithm 1: (s,t)-PCN$_1$ on (s,t)-extended outtrees

Data: (s,t)-extended outtree $G=(V,A)$ with root r,
$\mathcal{P} \subseteq \{[a,b] |\ a \in V_s, b \in V_t\}$ set of paths in G

Result: optimal solution $\mathcal{P}^*$ of the (s,t)-PCN$_1$ instance, optimal solution A^* of the dual (s,t)-PCN$_1$ instance

$k =$ maximum distance of node $v \in V \setminus s,t$ to root r
$A' = \emptyset$ a stack
$\mathcal{P}' = \mathcal{P}$ a set
$i = k-1$ an integer
while $i \geq 0$ **do**
  forall $a=(s,v)$, $v \in L_i$ **do**
    $\Phi(v) = \{u \in C_v \mid \exists p \in \mathcal{P}', (s,v) \in p$ and $(v,u) \in p\}$
    case $|\Phi(v)| = 0$ (VOID)
      do nothing
    case $|\Phi(v)| = 1$ (FIX)
      do add (v,u) with $u \in \Phi(v)$ to A'
      do for all $p \in \mathcal{P}'$ with $(v,u) \in p$, $(s,v) \notin p$ remove p from $\mathcal{P}'$
    case $|\Phi(v)| > 1$ (POSTPONE)
      do add (s,v) to A'
  decrease i by 1
do set $A^* = A'$
while $A' \neq \emptyset$ **do**
  $a = \text{pop}(A')$
  case $a=(s,v)$ for some $v \in V(T)$ (POSTPONED)
    do add one $p \in \mathcal{P}'$ with $a \in p$ to $\mathcal{P}^*$
    do remove all paths from $\mathcal{P}^*$ in conflict with p
  case $a=(v,w) \in A(T)$ (FIXED)
    do add one $p \in \mathcal{P}'$ with (s,v), $a \in p$ to $\mathcal{P}^*$
    do remove all paths from $\mathcal{P}^*$ in conflict with p
return $\mathcal{P}^*$, A^*

$a=(s,v)$ itself is in A^*. Yet, if $\Phi(v)=1$ when v is scanned, then the arc (v,u) that follows (s,v) on p is added to to A^*. □

A detailed proof can be found in [10].

5. Conclusion

We consider a basic subproblem in line planning. Calculating the maximum capacity of a line plan contains the problem of finding a maximum path constrained network flow. We extensively explore the complexity and inapproximability of this problem and its dual, including results for grids, graphs of bounded treewidth and other special classes. For some classes we provide polynomial-time algorithms.

Acknowledgment

We are grateful for discussion with Stefan Hougardy, H. J. Prömel and Ines Spenke. Further, we thank our referees for their substantial input.

References

[1] R. Borndörfer, M. Grötschel, and M. Pfetsch, A path based model for line planning in public transport, Report ZR-05-18, Zuse Institute Berlin, 2005.

[2] M. Bussieck, P. Kreuzer, and U. Zimmermann, Optimal lines for railway systems, Eur. J. Oper. Res. 96 (1997), 54–63.

[3] U. Feige, A threshold of ln(n) for approximating set cover, Journal of the ACM 45 (1998), 634–652.

[4] L. Ford Jr. and D. Fulkerson, Flows in networks, Princeton University Press, Princeton, 1962.

[5] M. Garey and D. Johnson, Computers and intractability: A guide to the theory of NP-completeness, W. H. Freeman and Company, New York, 1979.

[6] N. Garg, V. Vazirani, and M. Yannakakis, Primal-dual approximation algorithms for integral flow and multicut in trees, Algorithmica 18 (1997), 3–20.

[7] J. Hastad, Clique is hard to approximate within $n^{1-\varepsilon}$, Acta Mathematica 182 (1999), 105–142.

[8] D. Hochbaum, Approximation algorithms for NP-hard problems, PWS Publishing Company, Boston, 1997.

[9] B. Korte and J. Vygen, Combinatorial optimization: Theory and algorithms, Springer, Berlin, 2000/2002.

[10] C. Puhl, Robuste Linienplanung und das (s, t)-path constrained network flow problem, Diplomarbeit, TU Berlin, 2007.

[11] N. Robertson and P. Seymour, Graph minors. II. Algorithmic aspects of tree-width, Journal of Algorithms 7 (1986), 309–322.

[12] A. Schöbel and S. Scholl, Line planning with minimal traveling time, Proc 5th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS 05), Dagstuhl, Germany, 2005, webpage. http://drops.dagstuhl.de/opus/volltexte/2006/660