Layered graph approaches for combinatorial optimization problems

Luis Gouveia[1], Markus Leitner[2] and Mario Ruthmair[2]

[1]Universidade de Lisboa, Faculdade de Ciências, Departamento de Estatística e Investigação Operacional, Lisbon, Portugal. `legouveia@fc.ul.pt`
[2]University of Vienna, Department of Statistics and Operations Research, Vienna, Austria. {`markus.leitner|mario.ruthmair`}`@univie.ac.at`

April 8, 2018

Abstract

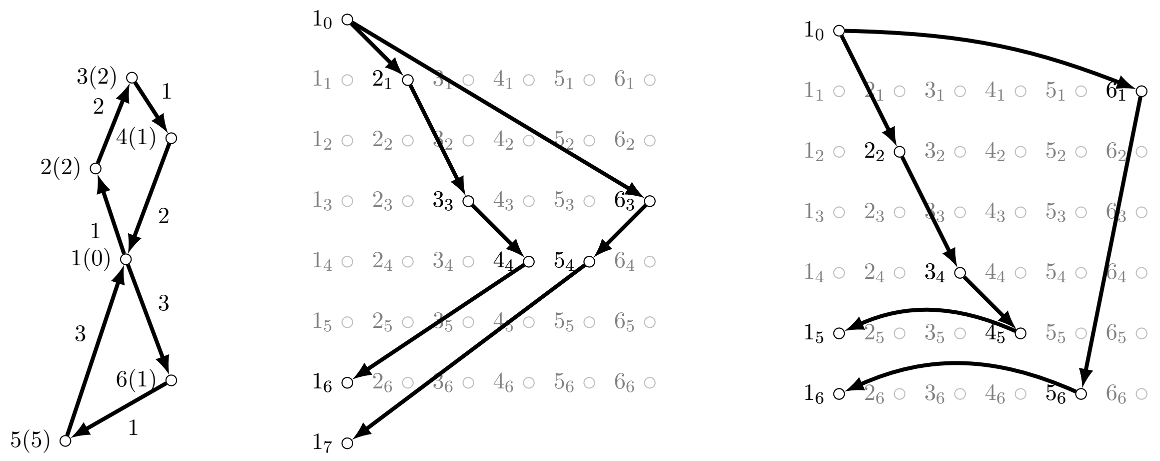
Extending the concept of time-space networks, layered graphs associate information about one or multiple resource state values with nodes and arcs. While integer programming formulations based on them allow to model complex problems comparably easy, their large size makes them hard to solve for non-trivial instances. We detail and classify layered graph modeling techniques that have been used in the (recent) scientific literature and review methods to successfully solve the resulting large-scale, extended formulations. Modeling guidelines and important observations concerning the solution of layered graph formulations by decomposition methods are given together with several future research directions.

Keywords: Layered graphs, integer programming, extended formulations, reformulation techniques, decomposition methods

1 Introduction

Several articles describing so-called layered graph formulations for modeling and solving combinatorial optimization problems have been published recently. This article classifies these formulations according to generic, mostly problem-independent characteristics, surveys existing approaches, and details cases where the use of layered graph models might be particularly beneficial. On several occasions, we also point out that some of the most promising, recently proposed ideas can be extended (easily) for other relevant problem classes. While our main concern is related to modeling aspects, we also point out that the (typically) tight dual bounds obtained from the associated linear programming (LP) relaxations are often hard to compute due to the large sizes of the underlying models. Thus, we also detail some of the most successful approaches that have been proposed to reduce the size of the layered graphs in order to make exact algorithms relying on layered graph formulations more successful in practice.

Though the designation *layered graph* has become popular rather recently, several older papers (some published even decades ago) exist that fit into this framework. Hence, it is difficult to point out what are the first papers referring directly or indirectly to layered graphs and that have initiated this stream of research. While important (and in particular, early) developments related to layered graphs have been made using time-space networks, we point out that our understanding of layered



(a) Solution (b) Time-indexed layered graph (c) Load-indexed layered graph

Figure 1: (a) A solution with two routes of an example instance of a VRP with travel time limit 7 and vehicle capacity 6. Travel times are provided next to the arcs and node demands are given in parentheses next to the node numbers. (b) The same solution shown in a time-indexed layered graph (aka time-space network). (c) The same solution shown in a load-indexed layered graph.

graphs is more generic and in no sense restricted to this conventional point of view which indeed can be seen as one particular special case.

We start by giving an intuitive idea of layered graphs and show that they extend the concept of time-space networks. Consider, e.g., a vehicle routing problem that is defined on a graph whose node set V contains a depot node 1, and customer nodes $\{2, 3, \dots, |V|\}$. A nonnegative travel time and a nonnegative demand are associated to each arc $a \in A$ and node $u \in V \setminus \{1\}$, respectively, see Figure 1a for an example. Then, the node set of the layered graph that corresponds to the conventional view as a time-space network is obtained by replicating each node u several times such that every copy u_t is associated with a particular visiting time t of customer u. Arcs (u_p, v_t) are included between pairs of replicated nodes u_p and v_t if the travel time from u to v is $t - p$ and the arc (u, v) exists in the original graph. Implicit or explicit bounds on the maximum travel time of each route restrict the size of such a layered graph. Observe that the load-dimension of this problem allows us to adopt an alternative viewpoint that results in a different layered graph where each node u is replicated several times to obtain copies u_q but, this time, indicating that the load of a vehicle is equal to q after serving node u. Arcs (u_q, v_l) of this load-indexed layered graph are included if the demand of node v is $l - q$. Figures 1b and 1c show the solution given in Figure 1a in the time-indexed and load-indexed layered graphs, respectively.

In this survey, we want to emphasize that there are many ways (different from time) to define the layers of a layered graph. This perspective allows to define layered graphs that implicitly restrict values of arbitrary (cumulative) resources within certain bounds (i.e., by the graph structure) rather than by using explicit constraints (which usually leads to formulations with weaker LP relaxation bounds). We will also point out that layered graphs often allow the modeling of complex dependencies relatively easy (e.g., costs depending on the current consumption of some resource) that would be difficult to model (by linear formulations) using alternative approaches.

Goal and structure. The main goal of this survey is to increase the awareness of the modeling options of layered graphs and of their benefits. A better knowledge of these options can increase the possibility of considering layered graphs when modeling complex problems by mixed integer linear formulations. To describe and classify existing modeling ideas later on, Section 2 will detail some of the consequences of adopting the layered graph perspective instead of only considering the more restricted time-space networks or time-dependent formulations. Section 3 discusses in more detail several layered graph modeling approaches using a capacitated vehicle routing problem with additional (generic) resource constraints as an example. Section 4 recalls and formally describes from an abstract perspective three modeling techniques that, together, can be seen as a unifying framework for most of the frequently used modeling techniques. Section 5 classifies the existing literature. Together, Sections 3 to 5 aim to introduce and explain the main modeling ideas related to layered graphs that appeared in the scientific literature as well as to summarize a significant part of it. For the latter, we do, however, focus on the domains of routing and network design. Since layered graph formulations are known to be attractive from a modeling perspective but may be difficult to solve algorithmically due to their size, Section 6 discusses important aspects and techniques that have been used in order to improve the performance of solution methods relying on layered graph formulations. To conclude, in Section 7 we also detail what we believe to be the main and most important avenues for future research related to layered graphs thereby indicating problem specifications for which layered graph formulations and solution methods might be particularly attractive.

Notation. Given a graph $G = (V, A)$ and a subset of nodes $S \subset V$, we will use notation $\delta^+(S) = \{(u, v) \in A \mid u \in S, v \notin S\}$, and $\delta^-(S) = \{(u, v) \in A \mid u \notin S, v \in S\}$. Analogous notation will be used for different graphs, and for simplicity we will write $\delta^-(u)$ and $\delta^+(u)$ instead of $\delta^-(\{u\})$ and $\delta^+(\{u\})$ for singleton sets $S = \{u\}$. Notation $z(W') = \sum_{u \in W'} z_u$ will be used for a set of variables $\mathbf{z}$ defined on set W and a subset $W' \subseteq W$.

2 Change of perspective: time-dependent vs. layered formulations

In this subsection, we show how researchers changed their view from modeling with time-dependent formulations to modeling with layered graph formulations. This discussion will be based on the Asymmetric Traveling Salesman Problem (ATSP) for which (1) provides a generic formulation (based on arc decision variables) which has been the starting point for several studies [1, 23, 47].

$$\begin{aligned}
\min \quad & \sum_{a \in A} c_a x_a && && \text{(1a)}\\
\text{s.t.} \quad & x(\delta^-(u)) = 1 && \forall u \in V && \text{(1b)}\\
& x(\delta^+(u)) = 1 && \forall u \in V && \text{(1c)}\\
& \{a \in A : x_a = 1\} \text{ induces a connected graph} && && \text{(1d)}\\
& x_a \in \{0, 1\} && \forall a \in A && \text{(1e)}
\end{aligned}$$

It is well known that a solution for (1b), (1c), and (1e), the so-called assignment relaxation, is composed of a set of disjoint circuits covering all nodes of the graph. Thus, the additional constraints for the generic part (1d) of the formulation need to prevent the existence of more than one cycle.

In the natural space of arc design variables this can be achieved, e.g., by the *subtour elimination constraints* first proposed by Dantzig et al. [12] or the equivalent *cut constraints*

$$x(\delta^-(S)) \geq 1 \qquad\qquad \forall S \subseteq V \setminus \{1\},\ S \neq \emptyset. \tag{2}$$

Despite the fact that the number of cut constraints (2) is exponential, they can be efficiently handled in an implicit way since they can be separated in polynomial time by using max-flow computations [48]. The obtained LP bounds are usually comparably good and current state-of-the-art methods for the ATSP as well as closely related problems are typically based on this formulation together with further valid inequalities.

In contrast, *extended formulations* use additional variables that frequently allow the derivation of *compact* formulations for which the number of variables and constraints is polynomial with respect to the input size. One of the first compact, extended formulations for the ATSP is a *time-dependent* formulation proposed by Picard and Queyranne [51] in the context of a machine-scheduling problem. This formulation (which uses node 1 as start and end node of the tour) is defined by adding time (or position) dependent variables Z_a^p that are equal to one if and only if arc $a \in A$ is in position $p \in \{0, 1, \ldots, |V| - 1\}$ and by replacing (1d) in the generic formulation by equations (3).

$$Z^0(\delta^+(1)) = 1 \tag{3a}$$

$$Z^p(\delta^-(u)) = Z^{p+1}(\delta^+(u)) \qquad \forall u \in V \setminus \{1\},\ \forall p \in \{0, 1, \ldots, |V| - 2\} \tag{3b}$$

$$\sum_{p \in \{0,1,\ldots,|V|-1\}} Z_a^p = x_a \qquad \forall a \in A \tag{3c}$$

$$Z_a^p \in \{0, 1\} \qquad \forall a \in A, \forall p \in \{0, 1, \ldots, |V| - 1\} \tag{3d}$$

The validity of this formulation (which will be denoted by PQ) will be discussed below when establishing its interpretation in a layered graph. Several articles use similar "time-dependent" formulations (see, e.g., [21, 23, 24]) and compare their LP relaxations. For completeness, we note that one set of assignment constraints, (1b) or (1c), becomes redundant (and can be removed) since the time-dependent flow conservation constraints (3b) already guarantee that if one arc goes into a given node then one arc must leave it (or vice-versa). Additionally, by using the linking constraints (3c), the original variables x_a can be eliminated and we obtain a model written with the time-dependent variables Z_a^p alone (this is how the PQ model was introduced in [51]). A first connection with a layered graph model can be established by considering the layered graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ defined by node set $\mathcal{V} = \{1_0, 1_{|V|}\} \cup \{u_p \mid u \in V \setminus \{1\}, p \in \{1, 2, \ldots, |V| - 1\}\}$ and arc set $\mathcal{A} = \{(u_p, v_{p+1}) \mid \{u_p, v_{p+1}\} \subseteq \mathcal{V}, (u, v) \in A\}$. The time-dependent part of formulation PQ, that is equations (3), can be viewed as modeling a path in $\mathcal{G}$ from node 1_0 to node $1_{|V|}$. While equations (3) do not forbid that multiple copies of the same original node may be included in such a path (in the original graph, such a solution would correspond to a $|V|$-circuit), this is prevented in the complete formulation through in-degree (1b) or out-degree (1c) constraints, respectively. The previously described connection with a layered graph method can go a step further by observing that one can work directly in this graph, that is, one can define cut constraints in the layered graph that strengthen the LP relaxation, and adapt the separation algorithms known for the similar constraints defined in the original graph. To define cut constraints in a layered graph, observe that formulation PQ can be interpreted as a model for the Generalized TSP [18] defined in the layered graph. In the latter problem, the node set is partitioned into subsets and one seeks for a minimum cost circuit including exactly one node from each subset. All copies of each original node in the layered graph form a subset and exactly one copy must be visited. This relation suggests the *layered graph cut constraints* [30]

$$Z(\delta^-(S)) \geq 1 \qquad \forall S \subset \mathcal{V}, 1_0 \notin S, \exists u \in V \setminus \{1\} \mid \{u_p : p \in \{1,2,\ldots,|V|-1\}\} \subseteq S \tag{4}$$

which also appeared in [23] as a theoretical outcome of a compact flow based formulation defined in the layered graph. The cut constraints (4) are valid since at least one copy of each node needs to be included in the tour and the set S is constrained to include all copies $u_1, u_2, \ldots, u_{|V|-1}$ of at least one original node u. Despite producing strong LP bounds (see, e.g., Godinho et al. [23] for bounds obtained with an equivalent flow formulation), existing and well engineered natural space approaches [53] typically render branch-and-cut algorithms based on constraints (4) as not competitive due to the large number of layered graph variables and separated layered graph cut constraints. On the contrary, such layered graph branch-and-cut algorithms may be competitive for slightly different problem variants whose definition excludes the existence of efficient natural space approaches. As one example, we refer to the time-dependent TSP [1] and its particular case of the cumulative TSP for which very tight LP gaps of a layered graph model are reported by Godinho et al. [23] (by using a theoretically equivalent flow based model).

Two special cases of the cut constraints (4) are worth mentioning. When the set S is composed solely of whole subsets of node copies, by using the linking constraints (3c), we obtain the cut constraints (2). The other case is obtained when $S = \{u_p, v_1, v_2, \ldots, v_{|V|-1}\}$ for two nodes u and v (and $2 \leq p \leq |V|-2$) (observe that S contains all copies of a given node v and one copy of another node u which is not on the first or last layer). Using equations (3) it can be shown that in this second special case, the cut inequalities are equivalent to

$$Z_{uv}^p \leq \sum_{(i,u)\in A, i\neq v} Z_{iu}^{p-1} \qquad \forall (u,v) \in A, \forall p \in \{2,3,\ldots,|V|-2\}. \tag{5}$$

These *two-cycle elimination constraints* state that if arc (u,v) is in position p then an arc in position $p-1$ coming from a node different from v has to enter node u. A similar set of constraints symmetric to inequalities (5) can also be obtained which can be shown, however, to be equivalent to the previous one, in terms of the corresponding LP relaxations. To the best of our knowledge, these inequalities have been first proposed in Gouveia [25] as a new class of (time-dependent) inequalities to break subtours in the context of tree problems. Later, such inequalities have been used for related tree problems [11, 43]. As noted above, these inequalities can also be used for routing problems. Results known from the literature [1, 23] suggest that adding inequalities (5) to model PQ improves the LP bound, more noticeable and significant for instances with time-dependent costs. In the next sections we will introduce layered graph modeling aspects, a classification of existing approaches, and a discussion of known methods to reduce the size of layered graphs. Before going into these details we would like to point out that layered graph models and solution approaches might be well suited to solve rather complex problems for which no tight natural space models and correspondingly well-performing natural space methods exist. On the contrary, the large number of layered graph variables may require significant implementation effort when aiming to achieve a performance comparable to, e.g., such natural space methods.

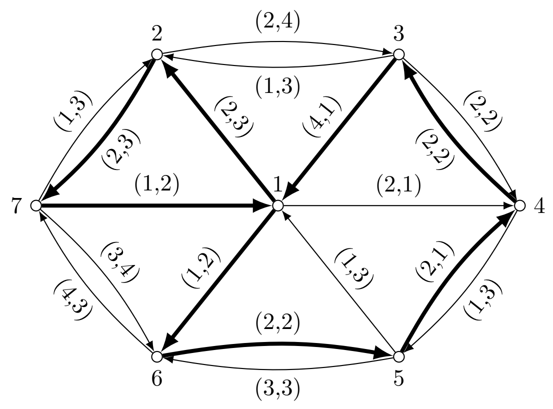


Figure 2: An example instance of the RCVRP with unit node demands, $Q = 4$, and $R = 8$. Arcs $a \in A$ are labeled by (c_a, r_a). Bold arcs indicate a feasible solution with an objective value of 16 consisting of the two routes $(1, 2, 7, 1)$, with resource consumption 8 and demand 2, and $(1, 6, 5, 4, 3, 1)$, with resource consumptions 8 and demand 4, to this instance.

3 Resource Constrained Capacitated Vehicle Routing

In this section, we will discuss in detail some of the modeling techniques with layered graphs that have been used recently, see, e.g., Gouveia et al. [35]. For this analysis we use the *Resource Constrained Capacitated Vehicle Routing Problem (RCVRP)* that is defined on a directed graph $G = (V, A)$ whose node set is the disjoint union of a depot node 1 and customer nodes $N = \{2, 3, \ldots, |V|\}$. Each customer $u \in N$ has an associated demand $q_u > 0$ while a cost $c_a \geq 0$ and resource consumption $r_a > 0$ are associated to each arc $a \in A$. The goal is to identify a set of at most K tours (each starting and ending at the depot) with minimum total cost such that the total demand of customers included in each tour does not exceed Q and the total resource usage of each tour does not exceed R. An example instance together with a feasible solution is given in Figure 2.

The RCVRP generalizes well known variants of the CVRP such as the distance constrained VRP [36], which is the special case with limited travel costs for each tour, i.e., $r_a = c_a$, $\forall a \in A$. The extra limiting resource might also be interpreted as battery usage in the context of electric vehicle routing problems. In the following, we will focus on a particular variant of the RCVRP with unit demands, i.e., in which $q_u = 1$, $\forall u \in N$. This assumption covers cases in which the number of customers each vehicle can visit is the restrictive factor instead of a more general, individual customer demand. It will also allow us to discuss implications and provide modeling and implementation suggestions for problems with multiple resources and where the number of feasible values (and the limit) of one constraint is comparably small while other resources have more general and larger domains.

Generic formulation (6) for the RCVRP will be used for the following discussion. This formulation will be augmented with various sets of variables and constraints to introduce different classes of layered graph modeling approaches. Natural space decision variables $x_a \in \{0, 1\}$, $\forall a \in A$, indicate whether an arc is included in a tour or not.

$$
\begin{aligned}
\min \quad & \sum_{a \in A} c_a x_a && && \text{(6a)}\\
\text{s.t.} \quad & x(\delta^+(1)) \leq K && && \text{(6b)}\\
& x(\delta^-(u)) = 1 && \forall u \in N && \text{(6c)}\\
& x(\delta^+(u)) = 1 && \forall u \in N && \text{(6d)}\\
& x(\delta^-(S)) \geq 1 && \forall S \subseteq N && \text{(6e)}\\
& \{a \in A : x_a = 1\} && \text{satisfies the capacity constraint for each route} && \text{(6f)}\\
& \{a \in A : x_a = 1\} && \text{satisfies the resource constraint for each route} && \text{(6g)}\\
& x_a \in \{0, 1\} && \forall a \in A && \text{(6h)}
\end{aligned}
$$

Constraints (6b)–(6d) ensure that at most K vehicles leave the depot and that the in- and out-degree of every customer node is exactly one. Connectivity constraints (6e) eliminate subtours and the constraints (6b)–(6e) therefore ensure that each solution forms at most K routes each starting and ending at the depot. Thus, a complete formulation is obtained by additionally ensuring the capacity and resource constraints for each route, cf. generic constraints (6f) and (6g).

Before turning our attention to layered graph formulations, we observe that several, well known possibilities allow to ensure the resource constraints. Some of them use only the arc variables and use capacity cut inequalities [42] (possibly together with additional sets of inequalities of exponential size). Natural space branch-and-cut algorithms based on them belong to the best approaches for solving different variants of VRPs but require the development of sophisticated separation procedures. Other approaches use a polynomial number (in terms of the size of the input graph) of additional variables and constraints, i.e., compact formulations. In the following, we will describe one such option that is based on single-commodity flows. Let variable $f_a \geq 0$ define the total load of a vehicle when traversing arc $a \in A$. Then, formulation (7) ensures the capacity constraints, i.e., models constraints (6f).

$$
\begin{aligned}
& f_a = x_a && \forall a \in \delta^+(1) && \text{(7a)}\\
& f(\delta^+(u)) - f(\delta^-(u)) = 1 && \forall u \in N && \text{(7b)}\\
& 0 \leq f_a \leq (Q+1)x_a && \forall a \in A \setminus \delta^+(1) && \text{(7c)}
\end{aligned}
$$

Using variables $g_a \geq 0$, $\forall a \in A$, that define the total resource spent by a vehicle after traversing arc $a \in A$, a similar formulation (8) ensures the resource constraints, i.e., models constraints (6g).

$$
\begin{aligned}
& g_a = r_a x_a && \forall a \in \delta^+(1) && \text{(8a)}\\
& g(\delta^+(u)) - g(\delta^-(u)) = \sum_{a \in \delta^+(u)} r_a x_a && \forall u \in N && \text{(8b)}\\
& 0 \leq g_a \leq R x_a && \forall a \in A \setminus \delta^+(1) && \text{(8c)}
\end{aligned}
$$

The following subsections will introduce separate and combined layered graph formulations for the two resource constraints of the RCVRP. To simplify notation, we will thereby assume integral resource values $r_a \in \mathbb{N}_+$. For the same reason, we also introduce a copy of each original node at each possible (and feasible) resource state. With respect to the former assumption, we observe that the only necessary assumption is that the number of achievable resource values is finite and that this assumption always holds due to the combinatorial nature of the RCVRP since there exists only a finite number of paths from the depot to any other node. With respect to the latter assumption, we will provide details about efficient construction and preprocessing methods of layered graphs in Section 6. Further note that throughout this article, we will denote by *achievable resource values* the subset of feasible resource values that can be realized in a solution.

We refrain from introducing models that contain one layered graph per vehicle. Such formulations have the advantage of allowing to model one resource constraint by a single knapsack constraint per vehicle, and implicitly ensuring the other resource constraint due to the structure of the layered graph. However, besides the large number of variables they also include a significant amount of symmetries with well-known disadvantages in (branch-and-bound based) solution methods.

3.1 Resource layered graph

In this subsection, we detail how to complete the abstract formulation (6) using a layered graph $G_\mathrm{R} = (V_\mathrm{R}, A_\mathrm{R})$ that encodes the resource constraint, see Figure 3. Its node set is defined by $V_\mathrm{R} = \{1_0, 1_R\} \cup \{u_r \mid, u \in N, r \in \{1, 2, \ldots, R-1\}\}$, and consists of two copies of the depot node (a starting depot at resource value 0 and an end depot at resource value R) and copies of all customer nodes $u \in N$ at all layers (i.e., resource consumption values) r that may possibly be obtained in a feasible tour, i.e., $r \in \{1, 2, \ldots, R-1\}$. Visiting a node u_r, $u \in N$, in a solution indicates that the total resource consumption of the path from the depot to customer u is equal to r. The arc set is defined by $A_\mathrm{R} = \{(u_p, v_l) \mid \{u_p, v_l\} \subseteq V_\mathrm{R}, (u, v) \in A, p \in \{0, 1, \ldots, R - r_{uv}\}, l = p + r_{uv}, v \neq 1\} \cup \{(u_p, 1_R) \mid (u, 1) \in A, p \in \{1, 2, \ldots, R - r_{u1}\}$, that contains arcs (u_p, v_l) connecting two copies of original nodes u and v, $v \neq 1$, whenever $(u, v) \in A$ and $r_{uv} = l - p$, as well as arcs from copies u_p of customer nodes to the end depot 1_R if $(u, 1) \in A$ and $p + r_{u1} \leq R$.

One main difference to the layered graphs sketched in the introduction is that copies of the depot node are avoided at layers different from 0 and R. Thus, arc $(i_p, 1_R)$ is used instead of $(i_p, 1_l)$, $l = p + r_{i1} \leq R$, for all arc copies that possibly terminate a route, and as a consequence each feasible route can be modeled as a path from 1_0 to 1_R that contains at most one copy of a node $i \in N$. Formulation (9) is similar to the equation system (3) of the PQ formulation (see Section 2) adapted to the case of multiple tours and can be used in place of (6g). It uses layered graph arc variables $X^r_{uv} \in \{0, 1\}$, $\forall (u_r, v_s) \in A_\mathrm{R}$, that are equal to one if and only if arc $(u, v) \in A$ is used in one of the tours such that the total resource consumption after visiting u is equal to r.

$$X^0_{1v} = x_{1v} \qquad \forall (1, v) \in A \tag{9a}$$

$$X(\delta^-(u_r)) = X(\delta^+(u_r)) \qquad \forall u_r \in V_\mathrm{R} \setminus \{1_0, 1_R\} \tag{9b}$$

$$\sum_{(u_r, v_s) \in A_\mathrm{R}} X^r_{uv} = x_{uv} \qquad \forall (u, v) \in A \tag{9c}$$

$$X^r_{uv} \in \{0, 1\} \qquad \forall (u_r, v_s) \in A_\mathrm{R} \tag{9d}$$

As in the case of the (time dependent) ATSP discussed in Section 2, we can strengthen the LP bounds of the formulation obtained from (9) using two-cycle elimination constraints (10).

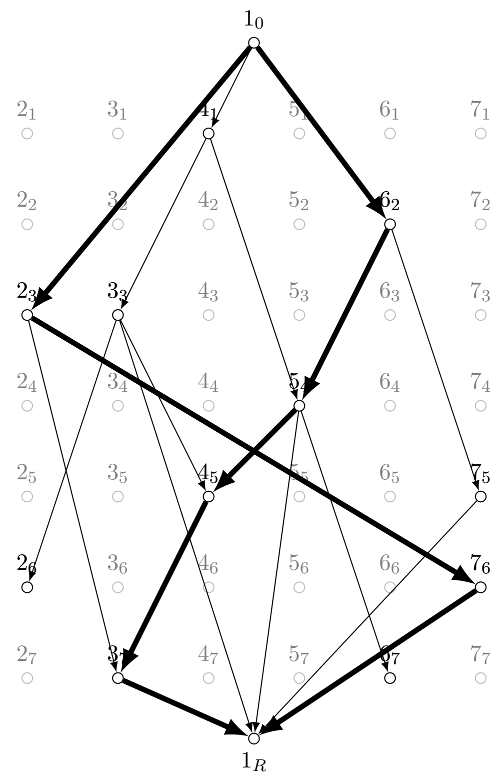


Figure 3: Resource-indexed layered graph G_{R} for the RCVRP instance given in Figure 2. The solution from Figure 2 is indicated by bold arcs. Gray nodes are not reachable from the depot and arcs adjacent to them are not shown (they can be removed in preprocessing).

$$X^r_{uv} \leq \sum_{(i_p,u_r)\in\delta^-(u_r), i\neq v} X^p_{iu} \qquad \forall (u_r, v_s) \in A_{\mathrm{R}},\ u, v \neq 1 \tag{10}$$

The family of layered graph cut constraints (11) is obtained from observing that there must exist a copy of every node that is reachable from the source depot 1_0. Thus, at least one arc must cross each cut separating the source depot from all copies of at least one customer node.

$$X(\delta^-(S)) \geq 1 \qquad \forall S \subseteq V_{\mathrm{R}} \setminus \{1_0\}, \exists u \in N : \{u_r, r \in \{1, 2, \dots, R-1\}\} \subseteq S \tag{11}$$

We note that original graph connectivity constraints (6e) are not required in a formulation obtained from replacing (6g) by (9) and (6f) by the single-commodity flow system (7) but typically strengthen, however, the associated LP relaxation. The latter does not hold if layered graph cut constraints (11) are included. It is easy to see that constraints (11) imply the original graph cuts (6e), see Section 2 for an explanation in the context of the ATSP. However, as detailed in Section 6 there may still be good reasons, from a practical point of view, to explicitly consider the original graph cuts (6e). Finally, we observe that despite the fact that a formulation using (9), possibly

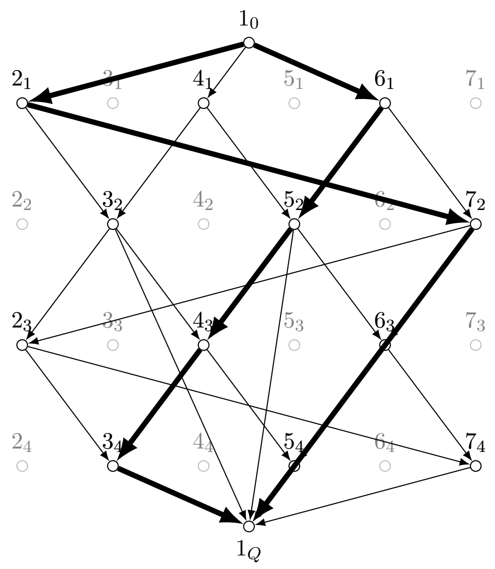


Figure 4: Load-indexed layered graph G_{Q} for the RCVRP instance given in Figure 2. The solution from Figure 2 is indicated by bold arcs. Gray nodes are not reachable from the depot and arcs adjacent to them are not shown (they can be removed in preprocessing).

augmented by (10), contains “only” a pseudo-polynomial number of variables and constraints, using it in practice often imposes non-trivial challenges for medium to large resource limits R and sets of achievable resource states. As indicated in the next section, in the context of the RCVRP one may alternatively focus on a layered graph that implicitly ensures the capacity constraint since the associated capacity limit will usually be much smaller in the case of unit demands. Other approaches and techniques to solve huge layered graph formulations are summarized in Section 6.

3.2 Load layered graph

An alternative formulation can be defined by considering a load-indexed layered graph $G_{\mathrm{Q}} = (V_{\mathrm{Q}}, A_{\mathrm{Q}})$ that encodes each vehicle's load and thus can be used to implicitly guarantee the capacity constraints. While it can be used in general, such an approach is particularly appealing when the number of achievable load values and the capacity limit Q are small, as, e.g., in the considered special case of unit demands. Then, node set V_{Q} consists of two copies of the depot node (a start depot with load 0 and an end depot with load Q) and copies of customer nodes at layers between 1 and Q, i.e., $V_{\mathrm{Q}} = \{1_0, 1_Q\} \cup \{u_q \mid q \in \{1, 2, \ldots, Q\}\}$. Arc set A_{Q} connects node copies that are connected in the original graph whenever a partial route can be continued using the respective arc, i.e., $A_{\mathrm{Q}} = \{(u_p, v_{p+1}) \mid \{u_p, v_{p+1}\} \subseteq V_{\mathrm{Q}}, (u, v) \in A, p \in \{0, 1, \ldots, Q-1\}, v \neq 1\} \cup \{(u_q, 1_Q) \mid (u, 1) \in A, q \in \{1, 2, \ldots, Q\}\}$.

The PQ system (12) builds upon variables $Y^q_{uv} \in \{0, 1\}$, $\forall (u_q, v_l) \in A_{\mathrm{Q}}$, which are equal to one if and only if a vehicle travels directly from u to v with a load equal to q after serving node u.

$$Y^0_{1v} = x_{1v} \qquad \forall (1, v) \in A \tag{12a}$$

$$Y(\delta^-(u_q)) = Y(\delta^+(u_q)) \qquad \forall u_q \in V_\mathrm{Q} \setminus \{1_0, 1_Q\} \tag{12b}$$

$$\sum_{(u_q,v_l)\in A_\mathrm{Q}} Y^q_{uv} = x_{uv} \qquad \forall (u,v) \in A \tag{12c}$$

$$Y^q_{uv} \in \{0,1\} \qquad \forall (u_q, v_l) \in A_\mathrm{Q} \tag{12d}$$

A complete formulation is obtained after replacing (6f) by (12) and using flow system (8) in place of (6g). As for the formulation introduced in Section 3.1 based on the resource layered graph we can reinforce the associated dual bounds by considering the compact set of two-cycle elimination constraints (13) and / or layered graph cut constraints (14).

$$Y^q_{uv} \leq \sum_{(i_{q-1},u_q)\in\delta^-(u_q), i\neq v} Y^{q-1}_{iu} \qquad \forall (u_q, v_l) \in A_\mathrm{R},\ u,v \neq 1 \tag{13}$$

$$Y(\delta^-(S)) \geq 1 \qquad \forall S \subseteq V_\mathrm{Q} \setminus \{1_0\}, \exists u \in N : \{u_q \mid q \in \{1,2,\ldots,Q\}\} \subseteq S \tag{14}$$

3.3 Independent resource and load layered graphs

If the LP relaxation gaps of a layered graph formulation is small, the additional burden of handling the associated large number of variables and constraints may pay off from a computational perspective. The use of single-commodity flow systems (or other well-known approaches involving a compact number of variables and constraints such as, e.g., Miller-Tucker-Zemlin constraints) for modeling a second class of resource constraints may, however, result in too large LP relaxation gaps. Thus, when using such an approach for problems with multiple resource constraints, one may end up with a huge formulation (due to one layered graph) producing a large LP optimality gap (due to the weak model used for the second resource). To overcome this disadvantage, we can use both layered graph models simultaneously, i.e., replace (6f) and (6g) by (12) and (9), respectively. The resulting formulation ensures both types of resource constraints implicitly and can be expected to produce (in many cases) much tighter LP relaxation gaps than the two previously described approaches, each employing a single layered graph. Two-cycle elimination constraints and layered graph cut constraints can be considered in one or both graphs to further strengthen the dual bounds. To further improve the LP bounds, it might be worth to investigate stronger (problem specific) linking constraints between the two sets of layered graph variables.

3.4 Three-dimensional load and resource layered graph

This section goes one step further by considering the two resource constraints in a single, three-dimensional layered graph $G_\mathrm{QR} = (V_\mathrm{QR}, A_\mathrm{QR})$, see Figure 5. Recent works [26, 32, 35] provide evidence that such graphs lead to significantly stronger LP bounds than the options discussed above. Thereby, node u_{qr} encodes that the load after visiting node u is equal to q while the total resource consumption from the depot up to u is equal to r. Consequently, $V_\mathrm{QR} = \{1_{00}, 1_{QR}\} \cup \{u_{qr} \mid u \in N, q \in \{1,2,\ldots,Q\}, r \in \{1,2,\ldots,R-1\}\}$ and $A_\mathrm{QR} = \{(u_{ql}, v_{q+1,r}) \mid \{u_{ql}, v_{q+1,r}\} \subseteq V_\mathrm{QR}, (u,v) \in A, r_{uv} = r - l, q \in \{0,1,\ldots,Q-1\}, v \neq 1\} \cup \{(u_{qr}, 1_{QR}) \mid (u,1) \in A, r + r_{u1} \leq R, q \in \{1,2,\ldots,Q\}\}$.

Using the PQ model (15) on G_QR in place of (6f) and (6g) yields a valid formulation. Thereby, layered graph arc variables $Z^{qr}_a \in \{0,1\}$, $\forall a = (u_{qr}, v_{ls}) \in A_\mathrm{QR}$, indicate whether arc (u,v) is used within a tour with a load of q and a total resource consumption of r after visiting node u.

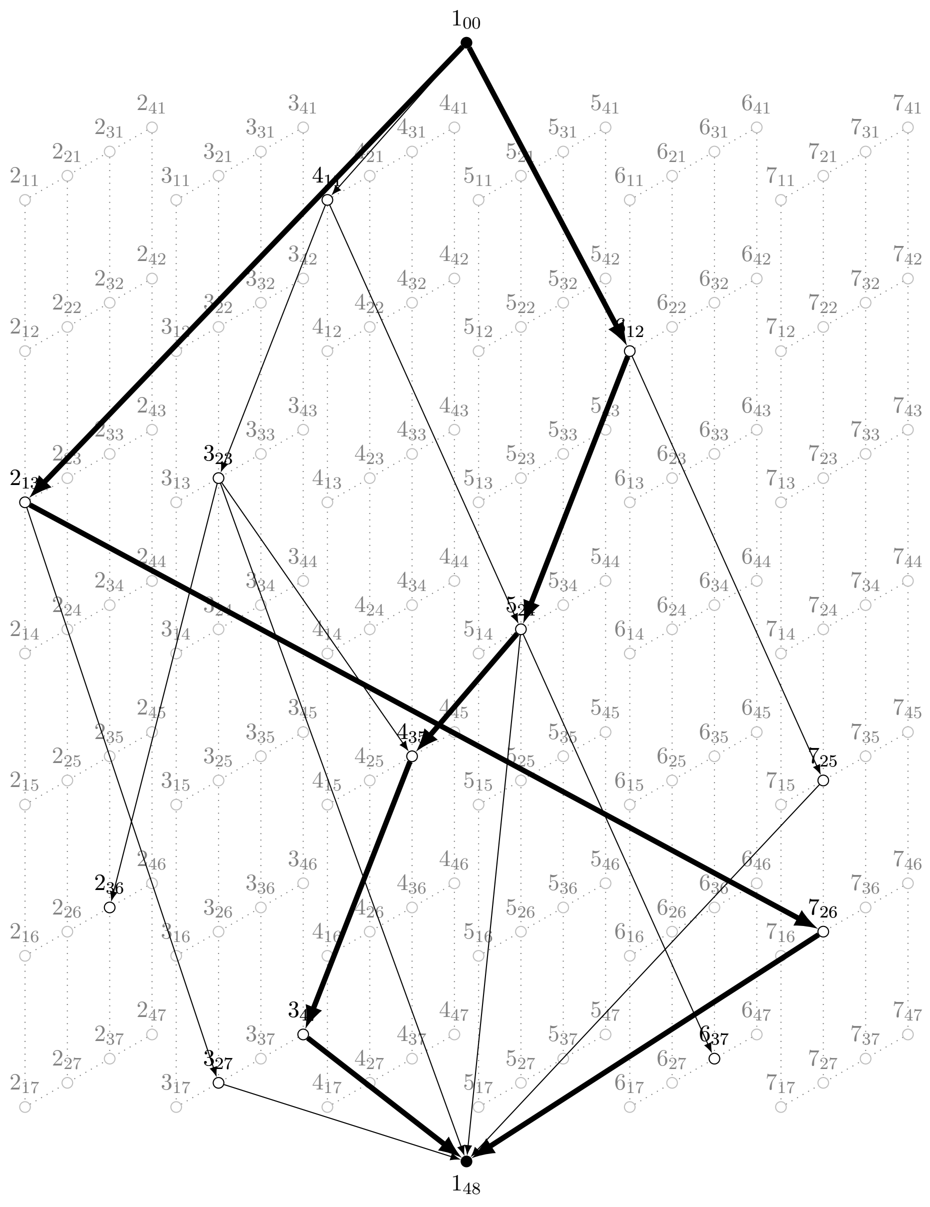


Figure 5: Load and resource layered graph G_{QR} for the RCVRP instance given in Figure 2. The solution from Figure 2 is indicated by bold arcs. Gray nodes are not reachable from the depot and arcs adjacent to them are not shown (they can be removed in preprocessing).

$$Z_{1v}^{00} = x_{1v} \qquad \forall (1, v) \in A \tag{15a}$$

$$Z(\delta^-(u_{qr})) = Z(\delta^+(u_{qr})) \qquad \forall u_{qr} \in V_{\mathrm{QR}} \setminus \{1_{00}, 1_{QR}\} \tag{15b}$$

$$\sum_{(u_{qr}, v_{ls}) \in A_{\mathrm{QR}}} Z_{uv}^{qr} = x_{uv} \qquad \forall (u, v) \in A \tag{15c}$$

$$Z_{uv}^{qr} \in \{0,1\} \qquad \forall (u_{qr}, v_{ls}) \in A_{\mathrm{QR}} \tag{15d}$$

As in the previous subsections, two-cycle elimination constraints (16) and / or layered graph cut constraints (17) can be used to strengthen the LP relaxation.

$$Z_{uv}^{qr} \leq \sum_{(i_{q-1,p}, u_{qr}) \in \delta^-(u_{qr}), i \neq v} Z_{iu}^{q-1,p} \qquad \forall (u_{q,r}, v_{ls}) \in A_{\mathrm{QR}}, u, v \neq 1 \tag{16}$$

$$Z^{qr}(\delta^-(S)) \geq 1 \qquad \forall S \subseteq V_{\mathrm{QR}} \setminus \{1_{00}\}, \exists u \in N : \{u_{rq} \mid r \in \{1, 2, \ldots, R-1\}, q \in \{1, 2, \ldots, Q\}\} \subseteq S\} \tag{17}$$

3.5 Comparison of LP relaxation bounds

Finally, we discuss and compare the LP relaxation bounds of the different modeling approaches described above based on a small example. We consider an instance of the RCVRP with 20 nodes located randomly in a 1000×1000 grid, unit customer demands, Euclidean arc costs, random arc resource values $r_a \in \{1, ..., 10\}, \forall a \in A$, and a minimal number of vehicles $K = \lceil |N|/Q \rceil$ for different vehicle capacities $Q \in \{4, 7, 10\}$ and different resource limits $R \in \{20, 30, 40\}$. The different layered graph formulations discussed in this section are compared with respect to their LP gaps which are defined as $(OPT - LB)/OPT$, where OPT and LB denotes the optimal integer value and the lower bound obtained from the LP relaxation, respectively. Base model (6) without connectivity constraints (6e) is part of all formulations. For the abstract parts (6f) and (6g) modeling capacity and resource constraints, respectively, we either use single-commodity flow (SCF) systems (7), (8), or PQ systems (9), (12), (15), depending on the layered graph(s) utilized. Additionally, different sets of valid inequalities are added in the variants. Detailed configurations and the resulting LP gaps are given in Table 1.

It can be clearly seen that all formulations based on layered graphs obtain tighter LP gaps than the variant with two SCF systems, except for the case when no further valid inequalities are added to the PQ systems. This immediately indicates that PQ systems alone do not necessarily lead to strong formulations but as soon as valid inequalities are added—especially in the space of the discretized variables, i.e., the two-cycle elimination (2C) and layered graph cut constraints (LCC)—gaps are heavily reduced in many cases.

When comparing variants based on single layered graphs, we observe that it is beneficial to use layered graphs in cases when the corresponding capacity or resource constraint is very tight. Then, the size of the corresponding formulations is moderate, the obtained LP gaps are tight, and there is not too much overhead related to handling the corresponding layered graphs. On the other hand, when capacity or resource bounds are loose, the overhead due to the large layered graphs might be too large to compensate the comparably small (in some cases negligible) gain in the value of the bounds when compared to the alternative modeling approaches. Examples for this observation can be seen when comparing the gaps for the extreme cases of $Q = 4, R = 40$ and $Q = 10, R = 20$. The results also indicate that using both layered graphs simultaneously can in most cases further reduce the gaps but usually only by a slight amount. The reason for this might be that the constraints linking the variables of each PQ system with the original arc variables are, usually, not too strong. As noted before, such a model might benefit from additional constraints directly linking the two sets of layered graph variables. This weakness is also avoided in the three-dimensional layered graphs which lead to the smallest gaps (indicated in Table 1 by bold values).

4 Modeling with layered graphs

The case study in the previous section detailed how to use single, multiple, and multi-dimensional layered graphs in the presence of two knapsack-type resource constraints for each "topological unit" which in the RCVRP is a route starting and ending at the depot. In this section, we review these three main modeling concepts from a general perspective, thereby, using an abstract optimization problem as a guideline. The discussion will also allow us to discuss more general resource constraints in addition to the knapsack-type constraints mentioned above. Furthermore, we will also point out further important modeling aspects potentially arising in problems extending (or being different from) the one used throughout this section.

Consider a generic combinatorial optimization problem defined on a directed graph $G = (V, A)$ with a dedicated depot (or root) node 1 and K resource values $r_a^k > 0$, $k = 1, 2, \ldots, K$, associated to each arc $a \in A$. Further assume that each feasible solution is the union of a set of at most L (not necessarily disjoint) building blocks $\mathcal{C}_l \subset A$, $l \in \{1, 2, \ldots, L\}$, each containing the special node 1, that need to respect certain topological constraints, e.g., form routes or paths. For simplicity, we assume throughout this section that each feasible solution connects node 1 with all nodes and contains precisely one path $P_u \subseteq \bigcup_{l=1}^{L} \mathcal{C}_l$ that connects the depot 1 with node $u \in V$, i.e., a unique total resource consumption from 1 to any other node can be computed. We also assume that for each resource $k \in \{1, 2, \ldots, K\}$ and node $u \in V$, set $\mathcal{R}_u^k$ specifies the feasible values for the total resource

Table 1: LP gaps of different formulations for a single RCVRP instance. (SCF: single-commodity flow system, PQ: Picard-Queyranne system in layered graph, CC: cut constraints in graph G, 2C: two-cycle elimination constraints in layered graph, LCC: layered graph cut constraints)

Model			LP gaps in % for Q/R								
Capacity	Resource	Valid inequalities	4/20	4/30	4/40	7/20	7/30	7/40	10/20	10/30	10/40
SCF (7)	SCF (8)	CC (6e)	13.3	11.8	11.4	12.0	11.1	9.9	6.7	4.6	4.5
PQ (12)	SCF (8)	-	7.2	5.1	5.0	12.3	13.3	11.3	6.8	10.7	10.3
		CC (6e)	5.6	3.5	3.1	8.2	8.4	6.3	4.8	2.1	2.0
		2C (13)	5.2	2.6	2.2	7.6	8.3	5.5	4.6	1.8	4.0
		CC (6e) + 2C (13)	5.2	2.6	2.2	7.6	7.0	5.1	4.6	1.4	1.6
		LCC (14)	5.2	2.6	2.2	7.6	6.9	5.1	4.6	1.3	1.2
SCF (7)	PQ (9)	-	9.6	13.0	14.3	10.4	12.4	13.6	7.3	11.2	9.7
		CC (6e)	7.0	10.2	11.0	6.6	6.4	7.4	4.8	2.0	1.9
		2C (10)	4.4	7.6	9.2	3.8	5.9	7.8	2.2	0.8	3.4
		CC (6e) + 2C (10)	4.4	7.0	8.9	3.8	4.8	5.9	2.2	0.6	1.4
		LCC (11)	4.3	6.9	8.9	3.8	3.6	5.9	1.6	0.4	0.9
PQ (12)	PQ (9)	-	4.9	4.6	4.9	10.2	11.3	10.6	6.6	10.2	9.3
		CC (6e)	4.5	3.3	3.1	6.6	6.3	5.9	4.8	1.9	1.9
		2C (10),(13)	3.3	2.2	2.2	3.8	5.8	5.0	2.2	0.4	3.1
		CC (6e) + 2C (10),(13)	3.3	2.2	2.2	3.8	4.5	4.3	2.2	0.4	1.2
		LCC (11),(14)	3.2	2.2	2.2	3.8	3.5	4.0	1.6	**0.0**	0.6
PQ (15)		-	3.5	4.1	4.5	8.1	9.7	9.3	5.2	6.8	7.4
		CC (6e)	3.4	2.7	3.0	6.0	6.0	5.3	3.2	0.9	1.7
		2C (16)	**2.5**	2.1	2.2	3.7	4.3	4.5	1.9	0.3	2.8
		CC (6e) + 2C (16)	**2.5**	2.1	2.2	3.7	3.2	3.9	1.9	**0.0**	0.9
		LCC (17)	**2.5**	**1.8**	**1.9**	**3.0**	**2.1**	**3.4**	**0.7**	**0.0**	**0.0**

usage $\sum_{a\in P_u} r_a^k$ on the path from 1 to node u. Concerning the latter set, observe that the discrete nature of the underlying optimization problem ensures that the number of achievable resource values (and hence a resulting layered graph in case it is not constructed by simply replicating nodes for all possible resource values but using the more clever method described in Section 6.1) is finite even in cases where $\mathcal{R}_u^k$, $k \in \{1, 2, \ldots, K\}$, is a continuous set, e.g., an interval in the context of time windows for vehicle routing problems. Knapsack-type resource constraints (as discussed in the previous section) are obtained as a special case of an interval when all lower bounds are equal to zero and when the upper bounds of all nodes coincide.

Assuming that the topological constraints are already enforced, the following three main patterns for using layered graphs to (implicitly) ensure (some of) the resource constraints can be seen as a unifying framework for most existing layered graph models from the literature.

4.1 Single (2-dimensional) layered graph with extra resource constraints

In this case, the main idea is to pick one resource, say l, and implicitly ensure the associated resource constraints in a layered graph $G^l = (V^l, A^l)$ while the constraints for resources $k = 1, 2, \ldots K$, $k \neq l$, are enforced through explicit (sets of) constraints and possibly via the inclusion of additional variables. The node set V^l typically consists of a root node 1_0 and set $\{u_r \mid u \in V \setminus \{1\}, r \in \mathcal{R}_u^l\}$ containing copies of all other nodes at feasible layers. In case each building block needs to end at a particular node (e.g., the depot 1) an additional copy (e.g., 1_R) of the latter node is typically included at the highest feasible resource value $R = \max \mathcal{R}_1^l$. The arc set A^l contains arcs $\{(u_p, v_q) \mid \{u_p, v_q\} \subseteq V^l, (u, v) \in A, q = p + r_{uv}^l\}$ connecting any two node copies u_p and v_q whose associated original nodes u and v are connected and when the resource usage r_{uv}^l along arc (u, v) satisfies $r_{uv}^l = q - p$. In case the above mentioned "target node" (say 1_R) is included, the arc set A^l typically also includes arcs $\{(v_q, 1_R) \mid v_q \in V^l, (v, 1) \in A, R \geq q + r_{v1}^l\}$. Assuming that the basic model ensuring the topological constraints includes arc design variables $x_{uv} \in \{0, 1\}$, $\forall (u, v) \in A$, a complete model is typically obtained through adding the following components:

(i) Layered graph arc variables $X_{uv}^p \in \{0, 1\}$ for each $(u_p, v_q) \in A^l$. In many situations, these variables can be replaced by their continuous relaxation since they become binary automatically due to the other constraints.

(ii) Linking constraints between layered and original graph arc variables. Under the assumptions made at the beginning of this section, these constraints can have the form

$$\sum_{(u_p,v_q)\in A^l} X_{uv}^p = x_a, \quad \forall a = (u, v) \in A. \tag{18}$$

Note that there exist particular problems (see, e.g., Leitner et al. [41]) where not all arcs of a solution must be used in each layered graph in which case above equations are replaced by less than or equal to constraints. If a problem allows for using an arc multiple times (at different resource values), the linking constraints typically need to be replaced by their weaker form

$$X_{uv}^p \leq x_a, \quad \forall (u_p, v_q) \in A^l, a = (u, v) \in A \tag{19}$$

or by

$$\sum_{(u_p,v_q)\in A^l} X_{uv}^p \leq M \cdot x_a, \quad \forall a = (u, v) \in A^l \tag{20}$$

in case an upper bound M on the number of traversals of arc a is known.

(iii) Necessary or strengthening constraints ensuring connectivity and / or the required topological structure on layered graph G^l. If G^l is acyclic (as for the considered problem since all resource values are assumed to be strictly greater than zero), constraints

$$X_{uv}^p \leq X(\delta^-(u_p)), \quad \forall (u_p, v_q) \in A^l, u \neq 1$$

can be used to ensure connectivity on G^l, i.e., to ensure that there exists a path from 1_0 to (the source of) each selected arc. For (routing) problems in which the in- and out-degree of each node must be equal to one, we can replace the latter constraints by constraints similar to the ones proposed by Picard and Queyranne [51], see, e.g., (9b). Note that two-cycle elimination constraints and layered graph connectivity constraints also fall into this class of constraints. Finally, we note that in contrast to all previous examples there exist cases in which layered graph cut constraints need to be included (even if original graph cuts and above compact connectivity constraints are considered). Situations like this arise, e.g., when a layered graph contains cycles and nodes may be visited more than once in solutions to the considered problem.

(iv) Additional variables and / or constraints (defined on the original graph or on layered graph G^l) ensuring all constraints related to resources different from l.

Observe, that K different options (one for each resource) for such formulations exist and it may not be obvious which of them is preferable both in theory and practice. Besides the theoretical strength of a resulting formulation, main factors influencing the choice are the size of G^l and options to model the constraints associated to resources different from l.

4.2 Multiple (independent, 2-dimensional) layered graphs

In contrast to the pattern discussed above, we can consider a set of K layered graphs where for each index k, $k = 1, 2, \ldots, K$, the layered graph $G^k = (V^k, A^k)$ implicitly ensures the resource constraints with index k, see, e.g., Gouveia et al. [35]. Each such graph is defined analogously to graph G^l for resource l introduced in the previous subsection. Hence, there is no need to include further explicit resource constraints, but instead one needs to ensure the existence of a solution (on the original graph) that can be embedded in a feasible way into each of these layered graphs. In that respect, a comparably simple (but nevertheless frequent) case arises whenever the strong linking constraints (18) are valid (for each resource) in which case they allow to directly link all layered graph variables to the original space variables. As opposed to that, replacing the latter constraints by inequalities or even considering only weak linking constraints (19) might lead to situations where the solutions on different layered graphs may differ. Thus, in such cases one may need to include further, problem specific constraints to ensure that all layered graph solutions map to a single, feasible solution in the original space (see, e.g., Gouveia et al. [31] for a similar situation). Necessary or strengthening constraints related to connectivity and / or the required topological structure (i.e., type (iii) from the list above) can be included for all or just a subset of all considered layered graphs. In addition, the explicit information about the resource usage on nodes or arcs for every resource may allow the removal of options in preprocessing or the inclusion of further, strengthening inequalities that improve the LP relaxation when compared to simply linking each set of layered graph variables to those defined in the original space. We conclude this pattern by pointing out that there exist several modeling approaches that may be considered in between the one discussed in this section and the previous one. These alternatives (for which we will skip a formal description) stem from cases where layered graphs are considered for a subset (of cardinality greater than one) of the resources and the remaining resources are modeled with extra constraints.

4.3 Multi-dimensional layered graphs

If $K \geq 2$, we can also employ a single, $(K+1)$-dimensional layered graph $G_{\mathrm{L}} = (V_{\mathrm{L}}, A_{\mathrm{L}})$ in which each node $u_{\mathbf{r}} \in V_{\mathrm{L}}$, $u \in V$, $\mathbf{r} = (r_1, r_2, \ldots, r_K) \in \mathbb{Q}^K$, $r_k \in \mathcal{R}^k_u$, $k = 1, 2, \ldots, K$, encodes the resource state vector of all K resources when visiting it. Arcs $(u_{\mathbf{r}}, v_{\mathbf{s}}) \in A_{\mathrm{L}}$ exist if $\mathbf{s} = \mathbf{r} + \mathbf{r}_{uv}$, whereby $\mathbf{r}_{uv} = (r^1_{uv}, r^2_{uv}, \ldots, r^K_{uv})$ is the resource vector associated to arc (u, v). Depending on the considered problem (see, Section 4.1) a copy of a target node, say $1_{\mathbf{R}}$, $\mathbf{R} = (\max \mathcal{R}^1_1, \max \mathcal{R}^2_1, \ldots, \max \mathcal{R}^K_1)$, may also be included together with arc set $\{(u_{\mathbf{r}}, 1_{\mathbf{R}}) \mid (u, 1) \in A, \mathbf{r} + \mathbf{r}_{u1} \leq \mathbf{R}\}$. While the size of such a graph can be enormous (even for $K = 2$), it has the benefit that each node (and hence every route or path in the graph) intrinsically satisfies all resource constraints. The application of techniques that aim to reduce its size (see Section 6 for a detailed overview on existing work in that domain) is, in this case, even more essential than for the modeling paradigms of the two previous subsections. Besides the fact that there is no need to explicitly ensure resource constraints, the modeling aspects and observations provided in Section 4.1 directly generalize to this multi-dimensional case (partly with slightly more complicated notation due to the vector-valued indices) and we therefore refrain from their repetition. Further details can also be found in Gouveia and Ruthmair [26] and Gouveia et al. [32, 35] that apply the concept of three-dimensional graphs in the context of routing and network design problems. Finally, note that many intermediate options exist that slightly differ from this concept or even combine the paradigms discussed in this and the latter two subsections. These options (which to the best of our knowledge have not been considered in the literature) include, e.g., considering one K'-dimensional layered graph, $3 \leq K' < K + 1$, together with explicit further constraints and / or one or more further layered graphs (which again might by multi-dimensional).

5 Further classifications of layered graph models

The previous two sections have detailed three main modeling concepts that can be used to classify most layered graph formulations. In this section, we augment this classification by pointing out additional, important aspects (together with implications concerning formulations and solution methods). Most of them can occur in any of the previously described three modeling concepts. Along the description of these aspects, we also provide pointers to corresponding articles (a more detailed overview is given in Table 2). Before introducing the additional aspects, we note that the formulations proposed in several articles, see, e.g., [27, 28, 29, 33], can be interpreted as introducing one layered graph per commodity (though they do not explicitly make use of layered graphs). Thus, in some sense they could be classified as introducing multiple, independent layered graphs. As the structure of the layered graph is not explicitly used or exploited and the different graphs are related to the same resource, we do, however, refrain from including such approaches in our classification.

Table 2: Overview on articles considering layered graph (LG) formulations and solution methods according to the criteria described in Sections 4 and 5.

Paper	problem resource	number of sources	fixed source	single LG (2-dimensional)	multiple LG (2-dimensional)	multi-dimensional LG	acyclic LG	implicit resource LG	general demand LG	natural space LG
Abeledo et al. [1]	position	1	✓	✓			✓			✓
Bendali et al. [3]	position	1	✓	✓			✓			
Boland et al. [6]	time	1	✓	✓			✓		✓	
Botton et al. [7]	position	1	✓	✓			✓			
Botton et al. [8]	position	1	✓	✓			✓			
Brandstätter et al. [9]	time	1	✓	✓			✓		✓	✓
Dash et al. [13]	time	1	✓	✓					✓	
De Boeck and Fortz [14]	position	1	✓	✓			✓			
Fischer and Helmberg [17]	time	1	✓	✓			✓		✓	
Fleischer and Skutella [20]	time	1	✓	✓			✓		✓	
Godinho et al. [23]	position	1	✓	✓			✓	✓		✓
Gouveia and Ruthmair [26]	load	1	✓	✓		✓	✓/✗	✓	✓/✗	
Gouveia et al. [30]	position	1		✓			✓			
Gouveia et al. [31]	position	≥ 1	✓		✓		✓			
Gouveia et al. [32]	position	2			✓	✓	✓	✓		
Gouveia et al. [35]	position/distance	1	✓	✓	✓	✓			✗/✓	
Leitner [37]	position	1		✓			✓			
Leitner et al. [41]	distance	≥ 1	✓		✓				✓	
Ljubić and Gollowitzer [43]	position	1	✓	✓			✓			
Mahjoub et al. [45]	position	≥ 1	✓		✓		✓			
Mahjoub et al. [46]	position	≥ 1	✓	✓						
Ruthmair and Raidl [55]	delay (time)	1	✓	✓					✓	
Ruthmair and Raidl [56]	delay (time)	1	✓	✓			✓		✓	
Sinnl and Ljubić [57]	position	1	✓	✓			✓			
Wang and Regan [62]	time	1	✓	✓					✓	
Wang and Regan [63]	time	1	✓	✓					✓	

5.1 Acyclic and general layered graphs

Until now, this article has focused on cases in which each considered resource may only increase or decrease, such as, e.g., time passed since a tour has been started, position in a solution relative to some source node, or load of a vehicle in vehicle routing problems. It is immediate that layered graphs considered in such situations are acyclic. Significant advantages obtained from exploiting the latter fact in modeling and the design of solution algorithms are one reason why a large part of the literature related to layered graphs focuses on these cases. As described in Section 4, the use of such acyclic layered graphs allows to ensure connectivity by a set of constraints whose size is polynomial with respect to the size of the layered graph (typically one constraint per node or arc is required). Hence, one usually obtains a formulation that is polynomial or pseudo-polynomial in the number of variables and constraints that usually can be strengthened by adding aforementioned cut constraints on the original and layered graph.

Several formulations and approaches for applications in logistics optimization that are based on acyclic layered graphs have been proposed, see, e.g., [1, 9, 23]. Further important examples that are typically motivated from (telecommunication) network design include limiting the transmission delay from one or more central sources (e.g., central servers) via hop-constraints [7, 8, 14, 30, 31, 37, 43, 57], or between all relevant nodes via diameter constraints [30, 32], as well as more general delay or distance constraints from one or multiple central nodes [55, 56]).

However, recent examples have described layered graph approaches where the underlying graph is not acyclic. The reason is that resource usage may both increase and decrease at nodes or along arcs in these problems, see, e.g., Gouveia and Ruthmair [26] for such an approach on pickup and delivery problems. Another example that fits into this category can be found in the context of electric vehicle routing problems where arc resource values correspond to energy consumption which might be negative due to battery recuperation, see Gouveia et al. [34]. Gouveia et al. [35] study the black-and-white TSP where distance and hop-constraints are imposed between consecutive pairs of black nodes in a TSP tour. The last requirement can be modeled as resetting the traveled distance and traversed hops to zero whenever a black node is visited. For this reason, some of the approaches proposed in [35] include layered graphs that contain cycles. Somewhat similar layered graph approaches are studied in [41] for a network design problem with upper bounds on the distance a signal may travel. Expensive signal-recovering equipment (i.e., relays) can, however, be placed at nodes in which case the travel distance associated with a signal is reset to zero. Leitner et al. [41] model the installation of such a device at node u with layered graphs including arcs from u_l to u_0 where l is the total distance of the signal from its source or the last relay node, respectively. Observe that the existence of the latter arcs will typically introduce cycles. Mahjoub et al. [46] consider layered graphs for a rather general class of network design problems where cycles arise from the fact that edges between node copies at identical layers may exist. It is easy to observe that previously mentioned (pseudo-) polynomial constraint sets are typically not sufficient for ensuring connectivity if a graph is not acyclic [26]. Thus, other sets of connectivity constraints on the original graph (or other formulations that include connectivity constraints such as, e.g., flow systems) need to be added to the model. To make matters worse, observe that the latter may not even be sufficient in case nodes (different from the source of a commodity) may be visited multiple times in which case the connectivity requirement needs to be ensured on the layered graph as well.

General layered graphs may also arise in approaches that consider approximations (together with possible refinements later on) of the set of achievable resource states, see, e.g., [13, 54, 55, 62, 63] and Section 6.2 for a detailed overview on such methods.

5.2 Explicit and implicit resource constraints

As detailed above, layered graph formulations have been mainly considered when the definition of an optimization problem contains one or multiple resources for which certain bounds need to be ensured per building block of a solution, see, e.g., [30, 43]. In contrast to these *explicit* resource constraints, a few papers suggested to use layered graphs for modeling *implicit* resource constraints. The latter includes cases that do not (further) limit the use of some resource but in which the underlying layered graph is solely used for modeling reasons. One notable example is the aforementioned work [23] for the ATSP in which a graph with $|V|$ layers is used to eliminate subtours via theoretically strong models. A second class of layered graph approaches making use of implicit resource constraints can be derived from the three-dimensional layered graph model proposed in Gouveia et al. [32]. In the latter work, the authors study a network design problem that aims to identify a minimum cost diameter-constrained spanning tree that contains a diameter-constrained Steiner tree defined on a subset of required nodes and subject to a different (smaller) diameter limit. In addition to the two explicit resource constraints, an implicit upper bound on the distance between the two tree centers is established and used to derive a non-straightforward layered graph formulation. Besides strong linear programming bounds associated to the latter formulation, the reported results also show that a branch-and-cut algorithm based on this formulation significantly outperforms alternative methods. Somehow similar, Gouveia and Ruthmair [26] propose a three-dimensional layered graph formulation for a pickup-and-delivery problem with one explicit (load) and one implicit (position) dimension.

From a more general perspective, we observe that the resource bounds of explicit resources typically do not increase (significantly) when the instance size increases since the latter should not influence the maximally allowed resource usage per solution component. In contrast, bounds for implicit resource constraints typically depend on the input size, e.g., the maximal position of an element within a solution depends on the total number of nodes. Assuming a maximally allowed resource value of R this indicates that each additional node will typically induce at most R additional node copies on a layered graph in case of explicit resources but up to $R + |V|$ additional node copies when modeling implicit resources (in case its upper limit increases by one unit when one node is added). These considerations seem particularly important when considering problems with multiple (implicit or explicit) resource constraints in which a decision whether a layered graph or other constraint sets should be used to ensure the bounds of each resource can be made separately for each resource.

5.3 Unit and general demands

From a pure modeling perspective, it does not make a difference whether layered graph formulations are considered for resources with unit (see, e.g., [1, 30, 43]) or general values (see, e.g., [13, 20, 62]). This is not true, however, with respect to solution methods. To see this, observe that general resource values often allow for a much larger number of achievable resource values and consequently lead to much larger resource-indexed graphs. This, in turn has implications concerning the performance of algorithms operating on them. Further note that the impact of layered graph preprocessing and approximation techniques may be significantly higher in the case of general demands.

5.4 Fixed and selected roots/depots

As indicated in Table 2, most relevant articles consider cases where layered graphs are used to bound resource usage from a fixed, predefined root (e.g., a depot in logistics or routing applications or a central server in telecommunication network design). Notable exceptions from this class include works on diameter constrained spanning trees [30, 32] where the central elements (i.e., nodes or edges) need to be selected and the maximum diameter is subsequently ensured by limiting the distance from them. Layered graph formulations and branch-and-cut algorithms for the generalized hop-constrained minimum spanning tree problem are studied in [37]. In this problem variant, one out of a fixed set of candidate root nodes (those that are contained in a predefined root cluster) needs to be chosen. As in the diameter case, this selection is modeled via an artificial root node that is connected to all these nodes together with a constraint ensuring that the out-degree of this artificial root node is equal to one.

5.5 Single and multiple roots

We can also classify layered graph formulations based on the existence of a single or multiple roots. The vast majority of articles (cf. Table 2) focuses on the former case which is easier to handle and similar to our case study given in Section 3. Therefore, the following discussion will focus on the cases of multiple roots or central nodes which require some additional modeling considerations. Considering applications in telecommunication network design, Gouveia et al. [31] and Leitner et al. [41] create one layered graph (one set of explicit variables) for each root of a fixed set of source nodes each of which is associated with hop constraints to a given set of other nodes. While these subgraphs are unrelated with each other in Leitner et al. [41], further strengthening constraints relating the solutions on the different layered graphs are considered in Gouveia et al. [31] where the union of these partial solutions must form a tree. Mahjoub et al. [46] propose a distance transformation where a flow formulation in a single (undirected) layered graph is used to ensure connectivity between the sources and targets of several commodities. More complex hop-constraints are, however, modeled using one layered graph per commodity. One of the models in [32] extends the approach of selecting a single central node by Gouveia et al. [30] to the case of two nested trees by considering two separate layered graphs together with strong linking constraints between the associated variables.

5.6 Extended and natural space layered graphs

Layered graph formulations are typically proposed as extended formulations with the aim to derive theoretically stronger models or to allow an easier modeling of certain relations. Extending an underlying problem by including resource-dependent costs would, however, make such a layered graph formulation a natural variable space formulation. The fact that an existing model only needs adaptations concerning the cost associated to the layered graph arcs also highlights one advantage of such approaches, i.e., their inherent flexibility. To the best of our knowledge, despite being rather appealing such natural space layered graph approaches have received little attention so far. The few studies in that direction include the work by Abeledo et al. [1] on the time-dependent TSP where costs depend on the position of an arc in the tour. Similarly, Godinho et al. [23] consider the special case of the cumulative TSP, where an arc (u, v) at position p has costs $(|V| - p + 1)c_{uv}$. Among other models, Brandstätter et al. [9] propose a layered graph for modeling the battery state of electric vehicles over a planning horizon, i.e., the graph has nodes b_t that indicate a battery state equal to b at time t. While simpler (and more effective) modeling techniques exist (and are considered in [9]) in case linear charging functions are assumed, such an approach can be used easily for the consideration of more general, non-linear charging functions.

6 Solving layered graph formulations

In cases involving large sets of achievable resource values, the size of the layered graphs and thus the size of the corresponding formulations might get huge, especially for multi-dimensional layered graphs [26, 32, 35, 55], see Section 4.3. Even though the LP bound of such formulations is usually quite tight and thus might lead to smaller branch-and-bound trees when compared with methods based on alternative (natural) formulations, solving the LP relaxation can take a substantial amount of time. In this section we discuss issues related to the solution of layered graph formulations from a computational point of view, especially how to deal with the size of large layered graphs and associated resource-indexed formulations.

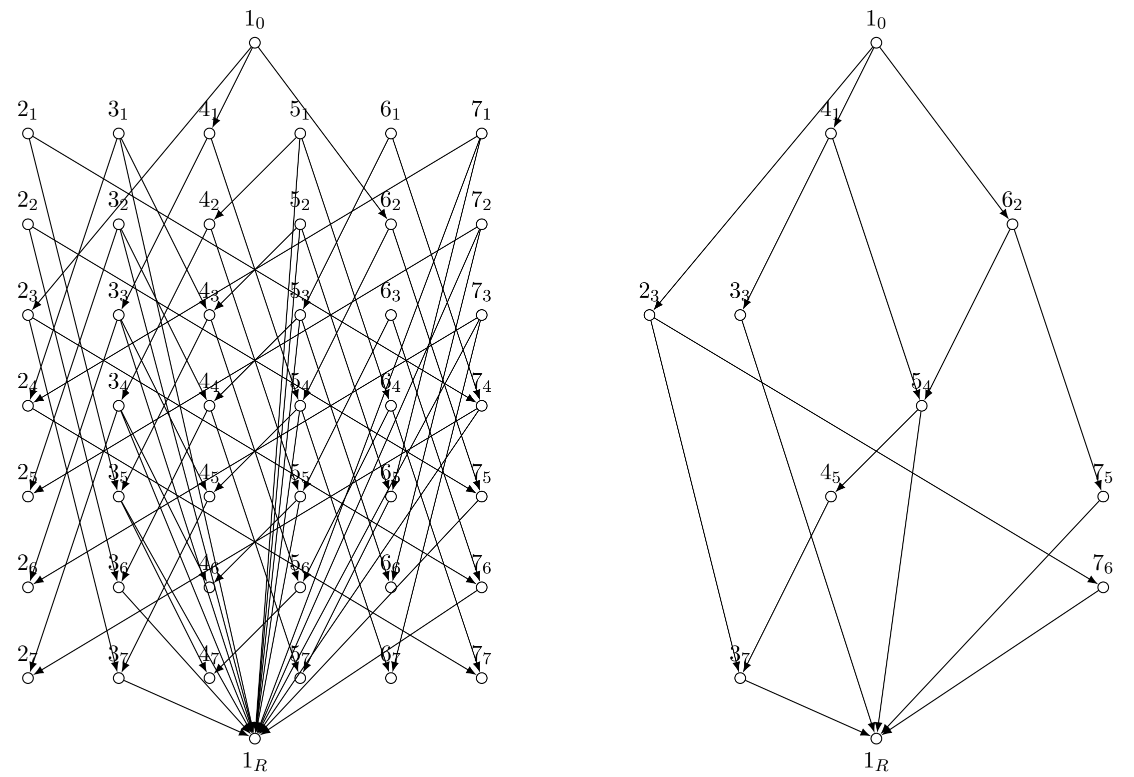
6.1 Preprocessing

Typically, resource-indexed formulations are directly built from a given problem instance, cf., the time-indexed formulations for scheduling problems [61]. More precisely, variables are created for each node and arc in the original graph and for each resource value within a given resource interval (based on a given granularity). However, many of these variables might not correspond to a resource state that is achievable in a feasible solution. Transforming the original graph to a layered graph before building the formulation helps in many cases to avoid the creation of variables corresponding to infeasible resource states. Depending on the problem being considered, nodes and arcs in the layered graph can be removed, e.g., based on the following observations which are applicable to many routing, network design, and scheduling problems: (i) Nodes without incoming arcs cannot be reached from some start node and can thus be removed together with all outgoing arcs. (ii) Nodes without outgoing arcs cannot be used to reach an end node and can thus be removed together with all incoming arcs. (iii) In case some node u can only be visited once and some node v_l has only one incoming arc (u_k, v_l), then arc (v_l, u_m) can be removed (if it exists) since using it would result in a cycle of length two (similar for a single outgoing arc). Depending on the type of layered graph being used, several preprocessing rounds can be necessary to eliminate all the situations mentioned above. See Figure 6 for a comparison of a layered graph for the RCVRP before and after preprocessing.

For some instances it might even be hard to generate the initial, not preprocessed, layered graph by simple replication of nodes and arcs because of too high time and memory consumption. Instead, we suggest the following graph generation directly leading to a layered graph in which each node has at least one incoming arc (except for the start nodes), see Figure 3. Beginning from each start node, for each outgoing arc in the original graph we create a corresponding arc copy in the layered graph together with the target node on the appropriate layer (if it does not exist yet). We repeat this process iteratively for each newly created node copy.

In some cases it might even be possible to detect infeasibility of an instance only based on the associated layered graph, e.g., if (after preprocessing) no copy of some original graph node which needs to be visited can be reached.

Besides eliminating graph components which cannot be in a feasible solution, one could also try



(a) Generated by simple replication of nodes and arcs

(b) After preprocessing

Figure 6: Resource-indexed layered graph for the RCVRP instance given in Figure 2.

to determine nodes and arcs which cannot be in an optimal solution. Such reductions often rely on problem-specific properties or on reduced cost fixing. For the latter case the dual ascent heuristic by Wong [64] has been successfully applied in a layered graph approach for hop- and diameter-constrained tree problems in Gouveia et al. [30] to quickly obtain tight dual bounds. Together with heuristic primal bounds it is possible to further eliminate arcs in the layered graph based on their reduced costs in case the residual optimality gap is small. Though Gouveia et al. [30] do not study the influence of these components on the overall approach, the excellent performance (compared to layered graph approaches for related problems) strongly suggests its enormous importance. This assumption is also supported by several recent articles, see, e.g., [19, 38, 39, 40], that show the tremendous influence when using dual ascent as an initialization or standalone method for solving related problems but which do not consider layered graph formulations.

6.2 Approximations

A more radical way to reduce the size of a layered graph is to remove non-redundant parts which might be relevant to obtain an optimal solution. Such reductions lead to smaller formulations but also to approximations of the original problem. Several such works fitting this approach can be found in the area of scheduling problems modeled by well-known time-indexed formulations, see, e.g., [2, 4, 5, 52]. Further articles mainly focus on dealing efficiently with time-related constraints in the areas of vehicle routing and network design, see, e.g., [6, 10, 13, 20, 44, 62, 63]. Clearly, these ideas are not limited to the time dimension but may also be applicable to other resources. Most of the approaches are based on coarser discretization of the considered resources. Viewed as a layered graph, these ideas correspond to using only a subset of its nodes, redirecting the layered graph arcs if necessary and thus changing their associated resource value, see, e.g., Boland et al. [6], Ruthmair [54], Ruthmair and Raidl [55].

Depending on how the formulation is adapted to the coarser discretization level, i.e., how the original node's and arc's resource values are modified, the solution space can be extended or restricted. In the first case we obtain a relaxation and thus dual bounds [4, 6, 10, 13, 44, 52, 54, 55, 62] while in the latter case we obtain heuristic solutions and thus primal bounds [20]. Both cases can also be combined within a solution framework to close the gap from both directions [10, 54, 55, 62].

The level of discretization can be fixed a priori which gives a single approximate formulation [2, 4, 5, 20] or dynamically refined in an iterative approach leading to a series of (monotonically improving) approximations. In the latter case the iterative refinement is accomplished either by globally increasing the level of discretization in a regular way [10], or by exploiting the LP or IP solution of the approximate formulation [6, 10, 13, 44, 52, 54, 55, 62]. Refinement algorithms are in general finite and end in worst case with the original resource-indexed formulation. In most cases optimality can be shown much earlier, e.g., if the solution to a relaxation is feasible for the original problem. However, due to convergence issues it might also make sense to stop the refinement prematurely which needs further ingredients to obtain an exact approach for the original problem. If the approximate formulation is a relaxation we need to extend the model to ensure feasibility, e.g., by dynamically adding inequalities to cut off infeasible solutions [13, 54], by linking the resource-indexed variables to the variables of an alternative, usually smaller but also weaker, formulation for the original problem [2, 5], or by applying specialized branching rules [4].

Finally, note that for resource states which are allowed to both increase and decrease, e.g., the vehicle load in pickup and delivery problems, it is not obvious how to apply the methods in this section to obtain feasible bounds for the original problem.

6.3 Cutting planes

Due to the typically large size of layered graph formulations it might make sense to not include all constraints a priori in the model but add them dynamically in a cutting plane fashion, which is mandatory for but not limited to exponentially-sized sets of constraints, e.g., layered graph cut constraints (11). Especially the latter ones and other valid inequalities in the extended discretized variable space are known to typically lead to strong LP bounds, see, e.g., [26, 35, 49, 50, 55, 58, 59, 60]. There are, however, some issues which need to be considered when using such large sets of layered graph inequalities:

(i) Solving the separation problem on a large layered graph can be time consuming.

(ii) The cut convergence can be slow, i.e., one may need too many cutting plane iterations which, in turn, add a lot of inequalities and many of them may be violated by a small amount only (tailing off).

(iii) The large number of (possibly quite dense) inequalities can make the (re)solution of the LP relaxation difficult and time consuming.

Several ways can be found in the literature to deal with these problems. Large sets of inequalities on a layered graph might include some subsets which are easier and faster to separate and have a sparser structure. Trying to find violated inequalities of such subsets first before considering the more general sets might significantly reduce separation times, improve cut convergence, and lead to faster LP resolution. For example, when considering the layered graph connectivity cuts (11), one might start by separating the subset of connectivity cuts (6e) in the original graph, and then the subset of two-cycle inequalities (10) in the layered graph [26, 35, 41]. It can also be beneficial to include a small subset of inequalities a priori in the model, e.g., by first running a dual ascent algorithm to (heuristically) solve the LP relaxation and which also gives a promising initial set of layered graph connectivity cuts similar to (11) for hop-constrained tree problems [30].

To avoid a too long tailing off phase, several proposed algorithms consider "early branching", i.e., to go to the branching phase even though there might still be violated inequalities. Stopping criteria for the cutting plane phase can, e.g., be (i) to add only inequalities which are violated by a minimal amount, see, e.g., [26, 35, 41], and / or (ii) too low LP bound improvements in the last iterations, see, e.g., [26, 35].

6.4 Decomposition Approaches

The authors in Abeledo et al. [1], Pessoa et al. [49, 50], Uchoa [58], Uchoa et al. [59], Van den Akker et al. [61] have chosen a different way to avoid large model sizes when considering resource-indexed formulations. Instead of directly using the set of discretized variables, by applying Dantzig-Wolfe reformulation they introduce an alternative, even larger set of variables which represent sets of discretized variables that are added dynamically by column generation. These new variables might correspond to entire (not necessarily feasible) routes in routing problems [1, 49], to subtrees in spanning tree problems [59], and to schedules in machine scheduling problems [50, 61], respectively. To exploit the strength of valid inequalities in the extended discretized variable space, each discretized variable and thus also valid inequalities can be expressed by the sum of a subset of the newly introduced variables. One other advantage of this approach, is that these valid inequalities in the extended space do not change the structure and complexity of the pricing problem which is important when used in a (robust) branch-price-and-cut approach.

7 Conclusions and research perspectives

Layered graphs allow to model comparably complex relations and aspects relatively easy within integer linear programming formulations. The price to pay for this flexibility is a quite large number of variables in these extended formulations. This makes the development of competitive, exact solution algorithms building upon layered graph formulations a challenging and nontrivial task, even though quite tight LP relaxation bounds can often be obtained from these layered graph formulations. In this survey paper, we have pointed out possible advantages of adopting the layered graph view over traditional resource-indexed (and in particular time-indexed) formulations. A case study on the resource constrained capacitated vehicle routing problem has been used to introduce and discuss in detail common layered graph modeling techniques. Introducing three main modeling concepts, we have formalized the latter techniques from a general and abstract perspective. These concepts enable a classification of the majority of existing articles considering layered graph models and solution algorithms. We also discussed in detail additional modeling aspects (and some of their implications) that can be used as further or alternative classifications. The aim of this survey is, however, to not only consider modeling aspects. Thus, we also provided an overview on main ideas developed in order to obtain efficient solution methods relying on large-scale, layered graph formulations. The main observations and messages of this survey include:

(i) Layered graphs are a powerful tool for modeling combinatorial optimization problems by integer linear programming formulations. In particular, certain complex or nonlinear dependencies that are difficult to address by other modeling techniques can be included relatively easily.

(ii) The dual bounds obtained from the LP relaxations of layered graph formulations (with appropriate valid inequalities) are often quite tight.

(iii) Using layered graph formulations to obtain competitive solution algorithms is not at all trivial. Indeed, it is important to well engineer all components of such a method and consider advanced methods such as, e.g., preprocessing, dual ascent, approximations of layered graphs, or early branching, within a developed algorithmic framework.

(iv) Whether or not a layered graph approach for a particular problem seems promising strongly depends on the number of achievable resource values (i.e., layers of the graph) and existing alternative models and solution algorithms. Using a layered graph approach may, for instance, not be a good idea if effective algorithms relying on natural space formulations with dual bounds of similar quality exist. Similarly, the existence of well engineered and sophisticated branch-and-price algorithms based on set covering models can be critical.

These observations also imply the following general (i.e., problem independent) avenues of potential future research related to layered graph approaches, some of which may change the perspective of the latter point.

(i) Further study of existing and development of new approximations for layered graphs that significantly reduce their size. We particularly refer to approaches that allow to derive both lower and upper bounds on the optimal solution value and (at least in principle) converge to an optimal solution. Observe that (to the best of our knowledge) no such methods are known for general layered graphs in which resource states can increase and decrease.

(ii) Obtain a better understanding of the influence and importance of cut initialization methods such as dual ascent in the context of layered graphs and use them in a general setting.

(iii) Develop more efficient methods concerning the use of layered graph cut constraints (or other sets of strong inequalities defined on layered graphs). This particularly involves cut selection strategies and cut convergence issues as several recent studies indicate that the strength of such inequalities is outweighed by the required computing times.

(iv) Obtain a better understanding of the projection of layered graph cuts to the natural space to derive more compact formulations with similar LP relaxations. Preliminary results in this direction have been obtained by Gouveia et al. [30] and indirectly by Godinho et al. [22].

Finally, we would like to point out the following characteristics of problems for which layered graph approaches have not been sufficiently considered yet, despite the fact that they seem well suited for them (in particular compared to other modeling options).

(i) Tight resource constraints in which the feasible values are associated to multiple intervals such as, e.g., multiple, tight time windows. As shown in Section 4, resource constraints with arbitrary sets of feasible values are easily modeled within layered graphs by simply excluding all node copies that correspond to infeasible resource states.

(ii) Synchronization constraints. Synchronizing different components (e.g., routes) of a solution is known to be difficult and is frequently modeled via Big-M terms within Miller-Tucker-Zemlin-like constraints or by explicit synchronization constraints (e.g., one per time point) that are part of the master problem in set-covering formulations. While the former approach results in weak LP bounds, the latter is known to be problematic for (the performance of) corresponding branch-and-price algorithms [15, 16]. Enforcing maximal or minimal differences between resource values of visits of the same node by different components can (in many cases) be modeled on layered graphs by simply restricting the feasible resource values by in-degree constraints that depend on the other visit.

(iii) Non-linear costs and non-linear resource constraints. As briefly discussed for the case of the time-dependent TSP, non-linear costs depending on a certain resource state are easily incorporated through simply changing the costs of each layered graph arc accordingly and lead to natural-space layered graph formulations. Similarly, cases in which the accumulated usage of a resource can depend on the individual arc values in a non-linear way can be easily handled by appropriately adapting the resource consumption of all outgoing arcs of a particular node copy depending on its associated resource state. On the contrary, such relations are not easy to model by (many) alternative approaches.

Acknowledgements

This work is supported by National Funding from FCT - Fundação para a Ciência e a Tecnologia, under the project UID/MAT/04561/2013 and by the Vienna Science and Technology Fund (WWTF) through project ICT15-014.

References

[1] H. Abeledo, R. Fukasawa, A. Pessoa, and E. Uchoa. The time dependent traveling salesman problem: polyhedra and algorithm. *Mathematical Programming Computation*, 5(1):27–55, 2013.

[2] P. Baptiste and R. Sadykov. On scheduling a single machine to minimize a piecewise linear objective function: A compact MIP formulation. *Naval Research Logistics (NRL)*, 56(6):487–502, 2009.

[3] F. Bendali, I. Diarrassouba, A. R. Mahjoub, and J. Mailfert. The k edge-disjoint 3-hop-constrained paths polytope. *Discrete Optimization*, 7(4):222–233, 2010.

[4] L.-P. Bigras, M. Gamache, and G. Savard. Time-indexed formulations and the total weighted tardiness problem. *INFORMS Journal on Computing*, 20(1):133–142, 2008.

[5] N. Boland, R. Clement, and H. Waterer. A bucket indexed formulation for nonpreemptive single machine scheduling problems. *INFORMS Journal on Computing*, 28(1):14–30, 2016.

[6] N. Boland, M. Hewitt, L. Marshall, and M. Savelsbergh. The continuous-time service network design problem. *Operations Research*, 65(5):1303–1321, 2017.

[7] Q. Botton, B. Fortz, L. Gouveia, and M. Poss. Benders decomposition for the hop-constrained survivable network design problem. *INFORMS Journal on Computing*, 25(1):13–26, 2013.

[8] Q. Botton, B. Fortz, and L. Gouveia. On the hop-constrained survivable network design problem with reliable edges. *Computers & Operations Research*, 64:159–167, 2015.

[9] G. Brandstätter, M. Leitner, and I. Ljubić. Location of charging stations in electric car sharing systems. Technical report, Department of Statistics and Operations Research, University of Vienna, Vienna, Austria, 2016.

[10] F. Clautiaux, S. Hanafi, R. Macedo, M.-É. Voge, and C. Alves. Iterative aggregation and disaggregation algorithm for pseudo-polynomial network flow models with side constraints. *European Journal of Operational Research*, 258(2):467–477, 2017.

[11] A. M. Costa, J.-F. Cordeau, and G. Laporte. Models and branch-and-cut algorithms for the Steiner tree problem with revenues, budget and hop constraints. *Networks*, 53(2):141–159, 2009.

[12] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4):393–410, 1954.

[13] S. Dash, O. Günlük, A. Lodi, and A. Tramontani. A time bucket formulation for the traveling salesman problem with time windows. *INFORMS Journal on Computing*, 24(1):132–147, 2010.

[14] J. De Boeck and B. Fortz. Extended formulation for hop constrained distribution network configuration problems. *European Journal of Operational Research*, 265(2):488–502, 2018.

[15] M. Drexl. *On some generalized routing problems*. PhD thesis, Faculty of Business and Economics, RWTH Aachen University, Aachen, Germany, 2007.

[16] M. Drexl. Synchronization in vehicle routing-a survey of VRPs with multiple synchronization constraints. *Transportation Science*, 46(3):297–316, 2012.

[17] F. Fischer and C. Helmberg. Dynamic graph generation for the shortest path problem in time expanded networks. *Mathematical Programming*, 143(1-2):257–297, 2014.

[18] M. Fischetti, J. J. Salazar González, and P. Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45(3):378–394, 1997.

[19] M. Fischetti, M. Leitner, I. Ljubić, M. Luipersbeck, M. Monaci, M. Resch, and D. Salvagnin. Thinning out Steiner trees: a node based model for uniform edge costs. *Mathematical Programming Computation*, 9(2):203–229, 2017.

[20] L. Fleischer and M. Skutella. Quickest flows over time. *SIAM Journal on Computing*, 36(6): 1600–1630, 2007.

[21] K. R. Fox, B. Gavish, and S. C. Graves. An n-constraint formulation of the (time-dependent) traveling salesman problem. *Operations Research*, 28(4):1018–1021, 1980.

[22] M. T. Godinho, L. Gouveia, and T. Magnanti. Combined route capacity and route length models for unit demand vehicle routing problems. *Discrete Optimization*, 5(2):350–372, 2008.

[23] M. T. Godinho, L. Gouveia, and P. Pesneau. Natural and extended formulations for the time-dependent traveling salesman problem. *Discrete Applied Mathematics*, 164(1):138–153, 2014.

[24] L. Gouveia. A 2n constraint formulation for the capacitated minimal spanning tree problem. *Operations Research*, 43(1):130–141, 1995.

[25] L. Gouveia. Using hop-indexed models for constrained spanning and Steiner tree models. In B. Sanso and P. Soriano, editors, *Telecommunications Network Planning*, pages 21–32. Kluwer Academic Publishers, 1999.

[26] L. Gouveia and M. Ruthmair. Load-dependent and precedence-based models for pickup and delivery problems. *Computers & Operations Research*, 63:56–71, 2015.

[27] L. Gouveia, P. Patrício, and A. de Sousa. Compact models for hop-constrained node survivable network design: An application to mpls. In *Telecommunications Planning: Innovations in Pricing, Network Design and Management*, pages 167–180. Springer, 2006.

[28] L. Gouveia, P. Patrício, and A. de Sousa. Hop-constrained node survivable network design: An application to MPLS over WDM. *Networks and Spatial Economics*, 8(1):3–21, 2008.

[29] L. Gouveia, P. Patrício, and A. de Sousa. Models for optimal survivable routing with a minimum number of hops: comparing disaggregated with aggregated models. *International Transactions in Operational Research*, 18(3):335–358, 2011.

[30] L. Gouveia, L. G. Simonetti, and E. Uchoa. Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs. *Mathematical Programming*, 128(1):123–148, 2011.

[31] L. Gouveia, M. Leitner, and I. Ljubić. Hop constrained Steiner trees with multiple root nodes. *European Journal of Operational Research*, 236(1):100–112, 2014.

[32] L. Gouveia, M. Leitner, and I. Ljubić. The two-level diameter constrained spanning tree problem. *Mathematical Programming*, 150(1):49–78, 2015.

[33] L. Gouveia, P. Patrício, and A. de Sousa. Lexicographical minimization of routing hops in hop-constrained node survivable networks. *Telecommunication Systems*, 62(2):417–434, 2016.

[34] L. Gouveia, M. Ruthmair, and D. Santos. Um modelo de fluxo para o electric traveling salesman problem. In C. Antunes, D. Cardoso, and F. da Silva, editors, *A Investigação Operacional em Portugal - novos desafios novas ideias, homenagem ao Professor Luís Valadares Tavares*, pages 133–143. IST Press, Lisboa, 2016.

[35] L. Gouveia, M. Leitner, and M. Ruthmair. Extended formulations and branch-and-cut algorithms for the black-and-white traveling salesman problem. *European Journal of Operational Research*, 262(3):908–928, 2017.

[36] G. Laporte, M. Desrochers, and Y. Nobert. Two exact algorithms for the distance-constrained vehicle routing problem. *Networks*, 14(1):161–172, 1984.

[37] M. Leitner. Layered graph models and exact algorithms for the generalized hop-constrained minimum spanning tree problem. *Computers & Operations Research*, 65:1–18, 2016.

[38] M. Leitner, I. Ljubić, J.-J. Salazar-González, and M. Sinnl. An algorithmic framework for the exact solution of tree-star problems. *European Journal of Operational Research*, 261(1):54–66, 2017.

[39] M. Leitner, I. Ljubić, M. Luipersbeck, and M. Sinnl. A dual-ascent-based branch-and-bound framework for the prize-collecting Steiner tree and related problems. *INFORMS Journal on Computing*, 2018. to appear.

[40] M. Leitner, I. Ljubić, M. Luipersbeck, and M. Sinnl. Decomposition methods for the two-stage stochastic Steiner tree problem. *Computational Optimization and Applications*, 69(3):713–752, 2018.

[41] M. Leitner, I. Ljubić, M. Riedler, and M. Ruthmair. Exact approaches for the directed network design problem with relays. Technical Report AC-TR-18-001, Algorithms and Complexity Group, Vienna University of Technology, Vienna, Austria, 2018.

[42] A. N. Letchford and J.-J. Salazar-González. Projection results for vehicle routing. *Mathematical Programming*, 105(2-3):251–274, 2006. ISSN 0025-5610.

[43] I. Ljubić and S. Gollowitzer. Layered graph approaches to the hop constrained connected facility location problem. *INFORMS Journal on Computing*, 25(2):256–270, 2013.

[44] R. Macedo, C. Alves, J. V. de Carvalho, F. Clautiaux, and S. Hanafi. Solving the vehicle routing problem with time windows and multiple routes exactly using a pseudo-polynomial model. *European Journal of Operational Research*, 214(3):536–545, 2011.

[45] A. R. Mahjoub, L. Simonetti, and E. Uchoa. Hop-level flow formulation for the survivable network design with hop constraints problem. *Networks*, 61(2):171–179, 2013.

[46] A. R. Mahjoub, M. Poss, L. Simonetti, and E. Uchoa. Distance transformation for network design problems. Technical report, 2017.

[47] T. Oncan, I. Altinel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36(3):637–654, 2009.

[48] M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, 33(1):60–100, 1991.

[49] A. Pessoa, E. Uchoa, and M. P. de Aragão. A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. *Networks*, 54(4):167–177, 2009.

[50] A. Pessoa, E. Uchoa, M. P. de Aragão, and R. Rodrigues. Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation*, 2(3-4):259–290, 2010.

[51] J. C. Picard and M. Queyranne. The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, 26(1): 86–110, 1978.

[52] M. Riedler, T. Jatschka, J. Maschler, and G. R. Raidl. An iterative time-bucket refinement algorithm for a high-resolution resource-constrained project scheduling problem. *International Transactions in Operational Research*, 2017.

[53] R. Roberti and P. Toth. Models and algorithms for the asymmetric traveling salesman problem: an experimental comparison. *EURO Journal on Transportation and Logistics*, 1(1):113–133, 2012.

[54] M. Ruthmair. *On Solving Constrained Tree Problems and an Adaptive Layers Framework*. PhD thesis, Vienna University of Technology, Institute of Computer Graphics and Algorithms, Vienna, Austria, 2012.

[55] M. Ruthmair and G. R. Raidl. A layered graph model and an adaptive layers framework to solve delay-constrained minimum tree problems. In O. Günlük and G. Woeginger, editors, *Proceedings of the 15th Conference on Integer Programming and Combinatorial Optimization (IPCO XV)*, volume 6655 of *LNCS*, pages 376–388. Springer, 2011.

[56] M. Ruthmair and G. R. Raidl. On solving the rooted delay- and delay-variation-constrained steiner tree problem. In A. Mahjoub et al., editors, *Proceedings of the 2nd International Symposium on Combinatorial Optimization*, volume 7422 of *LNCS*, pages 225–236. Springer, 2012.

[57] M. Sinnl and I. Ljubić. A node-based layered graph approach for the Steiner tree problem with revenues, budget and hop-constraints. *Mathematical Programming Computation*, 8(4): 461–490, 2016.

[58] E. Uchoa. Cuts over extended formulations by flow discretization. In A. R. Mahjoub, editor, *Progress in Combinatorial Optimization*, pages 255–282. ISTE-Wiley, 2011.

[59] E. Uchoa, R. Fukasawa, J. Lysgaard, A. Pessoa, M. P. de Aragão, and D. V. Andrade. Robust branch-cut-and-price for the capacitated minimum spanning tree problem over a large extended formulation. *Mathematical Programming*, 112(2):443–472, 2008.

[60] E. Uchoa, T. A. Toffolo, M. C. de Souza, A. X. Martins, and R. Fukasawa. Branch-and-cut and hybrid local search for the multi-level capacitated minimum spanning tree problem. *Networks*, 59(1):148–160, 2012.

[61] J. Van den Akker, C. A. Hurkens, and M. W. Savelsbergh. Time-indexed formulations for machine scheduling problems: Column generation. *INFORMS Journal on Computing*, 12(2): 111–124, 2000.

[62] X. Wang and A. C. Regan. Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, 36(2):97–112, 2002.

[63] X. Wang and A. C. Regan. On the convergence of a new time window discretization method for the traveling salesman problem with time window constraints. *Computers & Industrial Engineering*, 56(1):161–164, 2009.

[64] R. T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical Programming*, 28(3):271–287, 1984.