



Simultaneous planning for disaster road clearance and distribution of relief goods: a basic model and an exact solution method

Dirk Briskorn¹ · Alf Kimms² · Denis Olschok²

Received: 24 September 2019 / Accepted: 6 April 2020 / Published online: 27 April 2020
© The Author(s) 2020

Abstract

In recent years, more and more disasters occurred. Additionally, the amount of people affected by disasters increased. Because of this, it is of great importance to perform the relief operations efficiently in order to alleviate the suffering of the disaster victims. Immediately after the occurrence of a disaster, there is an urgent need for delivering relief goods to demand locations and affected regions, respectively. Due to roads being blocked or damaged by debris, some demand locations may be out of reach and therefore the delivery of relief goods is hampered. This paper investigates the basic problem of simultaneously unblocking roads in order to make demand locations accessible and delivering relief goods in order to satisfy demand. Strict deadlines for the delivery of relief goods are considered at the demand locations. A formal problem statement is provided, and its computational complexity is analyzed. Additionally, a mixed integer programming model is developed and an exact solution method based on a branch and bound approach is proposed. A computational study investigating the performance of the model formulation and the branch and bound algorithm is conducted.

Keywords Humanitarian logistics · OR in disaster relief · Disaster response · Distribution of relief goods · Disaster road clearance

✉ Alf Kimms
alf.kimms@uni-due.de

Dirk Briskorn
briskorn@uni-wuppertal.de

Denis Olschok
denis.olschok@uni-due.de

¹ Schumpeter School of Business and Economics, University of Wuppertal, Rainer-Gruenter-Str. 21, 42119 Wuppertal, Germany

² Mercator School of Management, University of Duisburg-Essen, Lotharstr. 65, 47057 Duisburg, Germany

1 Introduction

In recent years, the number of natural disasters such as hurricanes, earthquakes and tsunamis has grown. Furthermore, more and more people have been affected by disasters (see Coppola 2015; Guha-Sapir et al. 2016). Hence, an efficient planning of relief operations is of utmost importance. In the response phase, immediately after the occurrence of a disaster, it is an urgent task to supply relief goods to the disaster victims in the affected region. The last mile delivery is mainly carried out by road vehicles. Of course also helicopters are used for last mile delivery in disaster relief. But the number of helicopters is limited, and their operation is costly. Moreover, helicopters are needed for tasks like assessing damages from the air or transporting injured people to hospitals. Therefore, we focus on last mile deliveries using the road network. In such cases, fast delivery and direct routing are hampered by debris which blocks parts of the road network or parts of the road network being destructed. Due to blocked or damaged roads, some locations may be isolated or vehicles may have to make detours to reach victims at these locations. Road clearance operations can be performed in order to clear the blocked roads, so that isolated locations become accessible again or detours are not necessary anymore. Because road clearance is a time-consuming task, the deployment of road clearance operations should be well thought out. From a methodological point of view, two planning problems arise. On the one hand, the delivery of relief goods has to be planned in a way that the demand of the disaster victims is fulfilled in a timely manner. On the other hand, a subset of all blocked roads has to be selected and a sequence in which these blocked roads are cleared has to be determined. The motivation for clearing the selected roads in sequence instead of clearing them in parallel is as follows. By concentrating the scarce resources for road clearance on one blocked road after another, the duration of road clearance operations can be reduced in total as a single road gets cleared faster due to the concentrated resources.

Obviously, there exists an interdependence between the two mentioned planning problems. The sequence and duration of road clearance operations affect the distribution of the relief goods. For example, the later an isolated location is made accessible again, the later relief goods can be distributed to that location. Delivery order, arrival times and deadlines in the context of the distribution of relief goods have an influence on the road clearance operations. If, for example, an isolated location should be supplied in the beginning of the planning horizon, road clearance operations have to focus on blocked roads towards this location first. Because of this interdependence tackling, both planning problems simultaneously may be beneficial.

Consequently, in this paper, we deal with the simultaneous problem, which we name Basic Simultaneous Road Clearance and Distribution Problem (BSRCDP) and focus on the following basic scenario. We consider a region which is hit by a disaster. Often various locations within the considered region are affected differently by the disaster. Also the density of the population varies among the locations in the region. This results in different demand for relief goods such as water,

food, shelter and medical supplies at different locations. Some locations have a higher priority and need their relief goods earlier than other locations. Thus, with every demand location a demand volume and a certain deadline are associated. A deadline determines the point in time when the amount of requested relief goods at a demand location has to be completely fulfilled. Note that the deadlines are strict, a late delivery of the relief goods is not allowed. Certain amounts of goods can be supplied from several locations in the region. The supplies are not subject to any time restrictions and are available from the beginning of the planning horizon. Furthermore, some only slightly affected locations in the region may exist which have neither demand nor supply of relief goods, but they have to be considered as they represent road intersections. Regarding the roads in the region, we distinguish between blocked and unblocked roads because not all roads are blocked. A delivery of relief goods from a location with supply to a location with a demand is possible if these two locations are connected by at least one path of unblocked roads. Note that such an unblocked path can exist at the beginning of the planning horizon. If no path of unblocked roads between a location with supply and a location with a demand exists, a delivery is possible as soon as one path between these two locations is cleared. As we focus on a basic scenario in order to explore the fundamental structure, no vehicles and no related routing are considered. Moreover, we do not incorporate transportation times for the delivery of supply because in comparison with the time needed for road clearance, the transportation time is negligible. In contrast to some other papers dealing with road clearance, e.g., Özdamar et al. (2014), we assume that each blocked road can be reached instantly regardless which other roads are blocked. Generally, we assume all parameters to be deterministic and known at the beginning of the planning horizon. Although a disaster environment is faced, information about road damages and blockages can often be obtained by satellite images and by the use of helicopters and aerial drones. Information regarding the demand of the affected people is provided by aid organizations which are on site first (possibly by helicopter). BSRCDP can generally be applied to disasters occurring in cities as well as in rural areas. But as the road network is dense in cities, the probability of isolated locations is lower in cities than in rural areas. Thus, BSRCDP is more likely to be applied in rural areas.

A feasible solution to BSRCDP ensures that the demand is fulfilled and the corresponding deadlines as well as the supply amounts are respected. We aim to find a feasible solution that minimizes the duration of road clearance. The motivation is twofold: First, a faster road clearance allows for earlier deliveries. Second, after a disaster struck resources are scarce. The faster the road clearance operations are completed, the earlier machines and manpower can be used for further disaster response tasks.

In this paper we provide a mixed integer programming (MIP) model for BSRCDP which we name BSRCDP-MIP. As we will see in the following section, the literature on the simultaneous problem of road clearance and distribution of relief goods is limited. To the best of our knowledge, in papers dealing with the simultaneous problem no strict deadlines are considered. Additionally, only heuristic solution approaches are provided in papers focusing on deterministic variants. In addition to

the consideration of strict deadlines, the main contribution of our paper is a branch and bound (B&B)-based exact solution approach. The paper is organized as follows: In Sect. 2, an overview of the relevant literature is given. A detailed problem description including two formal problem statements and some complexity results are presented in Sect. 3. BSRCD-MIP is explained in Sect. 4. In Sect. 5, the B&B-based exact solution approach is introduced. The performance of BSRCD-MIP and the exact solution approach are evaluated in a computational study. The results are presented in Sect. 6 which is followed by a short conclusion in Sect. 7.

2 Literature overview

We focus on the papers dealing with the simultaneous problem which are most related to BSRCDP. The literature regarding solely relief distribution is reviewed by, e.g., de la Torre et al. (2012), Anaya-Arenas et al. (2014), Zheng et al. (2015) and Özdamar and Ertem (2015). In contrast, Çelik (2016) gives a comprehensive overview of the literature focusing only on road clearance and network restoration, respectively.

Regarding the simultaneous problem, the works of Liberatore et al. (2014) and Ransikarbum and Mason (2016) are long-term oriented and rather belong to the recovery phase. In contrast to the papers considering the response phase and to our study, neither Liberatore et al. (2014) nor Ransikarbum and Mason (2016) incorporate any time dynamics or scheduling aspects. But in the short-term oriented response phase, e.g., points in time when blocked roads get cleared are important for a smooth distribution.

The papers in the following focus on the response phase. Çelik et al. (2015) and Fikar et al. (2018) consider uncertainty. In the stochastic debris clearance problem studied by Çelik et al. (2015), the resource requirements of a blocked road to be cleared are assumed to be stochastic. A partially observable Markov decision process model is proposed to solve small- to medium-sized instances to optimality. Three heuristics are developed to solve larger instances. Additionally, the authors remark that their problem can be modeled using penalties for unsatisfied demand in a given period. In contrast to BSRCDP, the deadline mentioned by Çelik et al. (2015) is not strict. In the problem studied by Fikar et al. (2018), distribution points and a plan how to supply relief goods from a single depot to these points have to be determined. Network disruptions in terms of blocked roads due to aftershocks can occur at random times during the planning horizon. Different vehicle types are available for the transportation of relief goods. One type is additionally equipped with gear for road clearance. Also, interactions between people affected by the disaster are considered. An agent-based simulation and optimization framework is developed.

The following papers focus on deterministic scenarios within the response phase. In the paper by Yan and Shih (2009), one time–space network for road clearance considering several work teams and one for relief distribution are considered. Both networks are linked in such a way that a flow of relief goods on a damaged or blocked arc is only possible if the arc has been cleared before. Thus, both time–space

networks are integrated into one multi-objective, multi-commodity network flow problem which is formulated as a MIP model. A weighted objective function is used to tackle both objectives (minimizing the duration needed for clearance and for delivering relief goods to all demand locations), and a heuristic is developed to solve the weighted objective problem. Nurre et al. (2012) consider an integrated network design and scheduling problem. An operational network including supply and demand nodes is given. A set of work teams can add arcs to the network that can be used to maximize the cumulative weighted flow of relief goods sent from the supply locations to the demand locations over the planning horizon. The problem is modeled as an integer programming model, and valid inequalities for the model are discussed. Additionally, a heuristic dispatching rule is developed. Based on this work, Nurre and Sharkey (2014) define several integrated network design and scheduling problems with different performance metrics and scheduling objectives. Moreover, a general heuristic dispatching rule framework is developed. Cavdaroglu et al. (2013) consider not only the interdependence between the distribution of relief goods and road clearance tasks in a transportation network, but also the interdependencies between different networks or, more precisely, infrastructures like power, transportation, water and telecommunications. For each infrastructure, the flows of goods and the unmet demands as well as a restoration schedule including a selection of arcs to be repaired, an assignment of the corresponding tasks to work teams and sequences of the assigned tasks for each work team are determined. The problem is formulated as a MIP model, and a heuristic solution method is proposed. Sharkey et al. (2015) continue the work by Cavdaroglu et al. (2013) and consider different classes of restoration interdependencies. Additionally, different decision-making environments including centralized, decentralized and information sharing are discussed. The corresponding integer programming models depicting these environments are solved with a generic solver. González et al. (2016a) and González et al. (2016b) study a similar problem to Cavdaroglu et al. (2013) considering cost reductions associated with recovering multiple co-located components at the same time. Extending the approach from Sharkey et al. (2015), Smith et al. (2017) provide a game theoretical model to obtain recovery strategies depicting decentralized decision making. They extend the time-dependent model of González et al. (2016a). Morshedlou et al. (2018) integrate a work team scheduling problem with a vehicle routing problem. Several work teams are dispatched to restore an infrastructure network in order to increase the flow from supply nodes to demand nodes during the planning horizon. One additional aspect is the consideration of a dynamic restoration process where work teams are able to work cooperatively. Two MIP models are proposed. In the first one a blocked component is not operational unless it is fully cleared, whereas in the second model blocked components can be partially operational during clearance. A lower bound is obtained integrating a relaxed model formulation and valid inequalities. Additionally, a feasibility algorithm is introduced. Shin et al. (2019) consider a scenario containing one work team and one relief vehicle emanating from the same depot. The proposed MIP model simultaneously determines sequences of roads to be cleared by the work team and of demand nodes to be visited by the relief vehicle. An ant colony optimization algorithm is proposed for large-scale data.

3 Problem description and complexity results

In this section, a formal problem statement for BSRCDP is given and some complexity results are presented. In Sect. 3.1 an initial problem formulation with a corresponding example is introduced. A less straightforward but more concise problem formulation is presented in Sect. 3.2. The second problem has a smaller graph representation and is therefore named the reduced problem in the following. Complexity results are derived in Sect. 3.3.

3.1 The initial problem

Let an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \mathcal{S} \cup \mathcal{D} \cup \mathcal{H}$ and $\mathcal{E} = \mathcal{E}^F \cup \mathcal{E}^B$ be given. The elements in the node set \mathcal{S} represent the locations of relief good supply. We associate a positive supply $s_i \in \mathbb{R}_{>0}$ of relief goods with every supply node $i \in \mathcal{S}$. The elements in the node set \mathcal{D} represent the locations with demand for relief goods. With every demand node $j \in \mathcal{D}$, we associate a positive demand $d_j \in \mathbb{R}_{>0}$ for relief goods and a positive deadline $t_j \in \mathbb{N}$. The length of the planning horizon is defined to be $T = \max\{t_j \mid j \in \mathcal{D}\}$. Additionally, the elements in the node set \mathcal{H} represent locations without supply or demand. Such locations may be worth to be considered in order to mark, e.g., the end points of street sections. The edges of the graph are the elements in the sets $\mathcal{E}^B \subseteq \{(\ell, m) \in \mathcal{V} \times \mathcal{V} \mid \ell < m\}$ and $\mathcal{E}^F \subseteq \{(\ell, m) \in \mathcal{V} \times \mathcal{V} \mid \ell < m\}$ with $\mathcal{E}^F \cap \mathcal{E}^B = \emptyset$. An edge $(\ell, m) \in \mathcal{E}^B$ represents a blocked or damaged road or a section of such a road, whereas an edge $(\ell, m) \in \mathcal{E}^F$ stands for an unblocked road. As highlighted in Sect. 1, usually not all roads are blocked by debris. The planning horizon is divided into time units of equal length which we name periods in the following. Only one blocked edge $(\ell, m) \in \mathcal{E}^B$ can be cleared per period. We assume that in one period a predefined length of a real blocked road or, more specifically, a predefined amount of blocking debris can be cleared. This predefined length of a blocked road is represented by one blocked edge $(\ell, m) \in \mathcal{E}^B$. In other words, clearing one blocked edge $(\ell, m) \in \mathcal{E}^B$ takes one period. Note that if in reality a blocked road needs more than one period to be cleared, as many blocked edges $(\ell, m) \in \mathcal{E}^B$ (with clearance time of one period) as periods needed and additional nodes $h \in \mathcal{H}$ are taken to depict such a blocked road with clearance time of more than one period as a sequence. We further assume that all blocked roads require an integer number of periods to be cleared. An example for such a graph is given in Fig. 1. The supply nodes \mathcal{S} are depicted as triangles and we have $\mathcal{S} = \{1, \dots, 6\}$. We have the set of demand nodes $\mathcal{D} = \{7, \dots, 15\}$ which are illustrated as squares. Nodes of set \mathcal{H} are depicted as pentagons. In the example, we have $\mathcal{H} = \{16, 17, 18\}$. The unblocked edges \mathcal{E}^F are drawn as continuous lines and the blocked edges \mathcal{E}^B are drawn as dashed lines in Fig. 1.

A solution to the initial problem gives answers to the following questions: a) How many relief goods are delivered from which supply node to what demand node in what period? b) Which blocked road is cleared in period t such that the transports mentioned in a) can be carried out? The latter can be represented by a

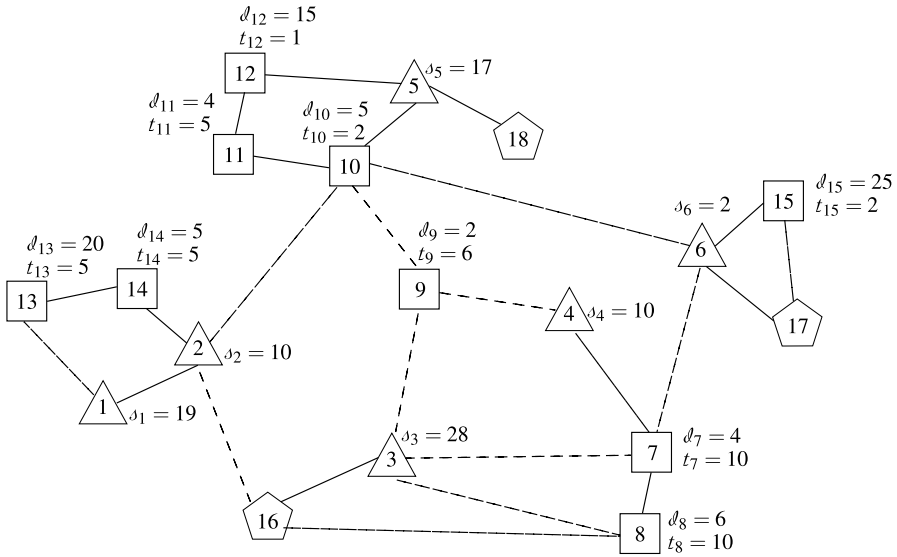


Fig. 1 Graph of the initial problem

sequence Ω of $k \leq T$ edges in \mathcal{E}^B such that each edge is contained at most once. Let $\Omega(t)$ be the edge at position t of Ω . Position t corresponds to the period in which the corresponding blocked edge is to be cleared. Sequence Ω of length k induces graphs $\mathcal{G}_0, \dots, \mathcal{G}_T$ with $\mathcal{G}_0 = (\mathcal{S} \cup \mathcal{D} \cup \mathcal{H}, \mathcal{E}^F)$, $\mathcal{G}_t = (\mathcal{S} \cup \mathcal{D} \cup \mathcal{H}, \mathcal{E}_t^F)$ with $\mathcal{E}_t^F = (\mathcal{E}^F \cup \bigcup_{t'=1}^t \{\Omega(t')\})$ for each $t \leq k$, and $\mathcal{G}_t = \mathcal{G}_k$ for $k < t \leq T$. Clearing blocked edge $\Omega(t) \in \mathcal{E}^B$ unblocks this edge and makes it passable from period t on. Note that the edge set \mathcal{E}_t^B can be obtained by $\mathcal{E}_t^B = (\mathcal{E}^F \cup \mathcal{E}^B) \setminus \mathcal{E}_t^F$ for each $t \leq T$ and that a sequence of graphs can be induced using edge sets \mathcal{E}^B and \mathcal{E}_t^B instead of \mathcal{E}^F and \mathcal{E}_t^F . This idea is used in Sect. 3.2.2.

A solution is feasible if the following restrictions hold. A delivery from a supply node $i \in \mathcal{S}$ to a demand node $j \in \mathcal{D}$ in period t can only be carried out if i and j are connected in \mathcal{G}_t . It is assumed that a delivery is always made at the end of a period. All demands d_j must be fulfilled. A demand d_j is fulfilled if the sum of deliveries of relief goods from supply nodes to this demand node is at least d_j and if these deliveries reach this demand node no later than t_j . Note that a demand node can receive deliveries from a supply node as soon as the demand node and the respective supply node are connected in \mathcal{G}_t . Furthermore, a supply node can only send at most that many relief goods as there are available at this node, that is s_i for $i \in \mathcal{S}$. We assume that the overall supply is equal or greater than the overall demand. Note that not necessarily all blocked edges need to be cleared to obtain a feasible solution and that due to the strict deadlines for some instances no feasible solution exists.

Each feasible solution can be evaluated by the length of a corresponding sequence Ω , i.e., the number of blocked edges to be cleared. Among the feasible

solutions we seek one with a sequence of minimum length which is equivalent to the minimum duration of road clearance.

A solution consists of a sequence of blocked edges to clear and specific deliveries of supply. Note that for a given feasible sequence of blocked edges to clear the specific deliveries can be easily determined. This aspect is used in a reduced problem which is presented in Sect. 3.2.

3.2 A reduced problem formulation

In this section, a formal problem statement for the reduced problem that is equivalent to the initial problem is presented. Before providing the formal problem statement in Sect. 3.2.2, some preliminary considerations on the graph representation are presented in Sect. 3.2.1.

3.2.1 Aggregating nodes into components

The aim of the following considerations is to obtain a smaller but equivalent representation of graph \mathcal{G} of the initial problem.

Graph \mathcal{G} of the initial problem contains nodes $\mathcal{V} = \mathcal{S} \cup \mathcal{D} \cup \mathcal{H}$. We focus on subsets of \mathcal{V} that are connected components with respect to the unblocked edges. Because all nodes within a component are directly or indirectly connected by unblocked edges and no transportation times need to be considered, deliveries within a component are immediately possible. Therefore, no further blocked edges need to be cleared to connect any pair of nodes within the same component. Consequently, every component in \mathcal{G} can be replaced by a single node which represents the whole component. The whole node set \mathcal{V} of \mathcal{G} can be replaced by a set of components C_0 . In order to obtain an equivalent representation of the whole node set as a set of components, the supplies s_i of the supply nodes $i \in \mathcal{S}$ as well as the demands d_j and the deadlines t_j of the demand nodes $j \in \mathcal{D}$ have to be transformed for every component as explained in the following.

- Every component $c \in C_0$ has a supply s_c which is the total supply in the component. In the case where there is no supply node in a component s_c is equal to zero.
- With every component $c \in C_0$ a parameter σ_c is associated which represents the number of demands with different deadlines in component c . Note that σ_c can also take value zero if no demand exists in a component. Parameter d_{cp} is the p -th, $p = 1, \dots, \sigma_c$, demand in component $c \in C_0$ and t_{cp} is its corresponding deadline (if $\sigma_c > 0$). Demands with same deadlines in a component are summarized to one demand. Note that $t_{cp} < t_{c,p+1}$.

Graph \mathcal{G} of the initial problem contains edge sets \mathcal{E}^F and \mathcal{E}^B . If the node set \mathcal{V} is replaced by a component set C_0 , also the set of edges changes. Obviously, the whole set of unblocked edges \mathcal{E}^F of the graph of the initial problem does not need to be incorporated as they form the components which are depicted as single nodes.

But also several blocked edges $(\ell, m) \in \mathcal{E}^B$ can be omitted. On the one hand, the blocked edges between nodes of a component in the initial problem can be omitted. The reason is again the depiction of the whole component as a single node. On the other hand, if in the initial graph \mathcal{G} several blocked edges exist between a pair of components, only one edge has to remain if the two components are depicted as single nodes. Clearing any of the blocked edges between the two components in the initial problem means clearing the one edge between the two as single nodes depicted components. Even if in graph \mathcal{G} of the initial problem several blocked edges between two components exist, in a sequence Ω of an optimal solution to the initial problem never more than one blocked edge between a pair of components will be part of this sequence, because we have assumed that transportation times are zero on unblocked edges. Generally, the edge set $\mathcal{E} = \mathcal{E}^F \cup \mathcal{E}^B$ can be replaced by a reduced set of blocked edges E_0^B .

Altogether, graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \mathcal{S} \cup \mathcal{D} \cup \mathcal{H}$ and $\mathcal{E} = \mathcal{E}^F \cup \mathcal{E}^B$ can be substituted by an equivalent, but smaller graph $G_0 = (C_0, E_0^B)$. Only if in graph \mathcal{G} the edge set \mathcal{E}^F is empty, graph G_0 is identical to graph \mathcal{G} . The reduced graph of our example (given in Fig. 1) can be seen in Fig. 2. Nodes 5, 10, 11, 12 and 18 from Fig. 1 are grouped together to form component 19 in Fig. 2. Analogously, nodes 1, 2, 13 and 14 form component 20, nodes 3 and 16 form component 21, nodes 6, 15 and 17 form component 22, and node 4 forms component 23. Completing, node 9 forms component 24 and nodes 7 and 8 form component 25.

3.2.2 Problem definition

Using the considerations from Sect. 3.2.1, the reduced problem can be formally defined as follows. An undirected graph $G_0 = (C_0, E_0^B)$ is given. Node set C_0 represents the components and with each component $c \in C_0$ a non-negative supply $s_c \in \mathbb{R}_{\geq 0}$ is associated. Having σ_c as the number of demands for each $c \in C_0$, parameter $d_{cp} \in \mathbb{R}_{>0}$ is

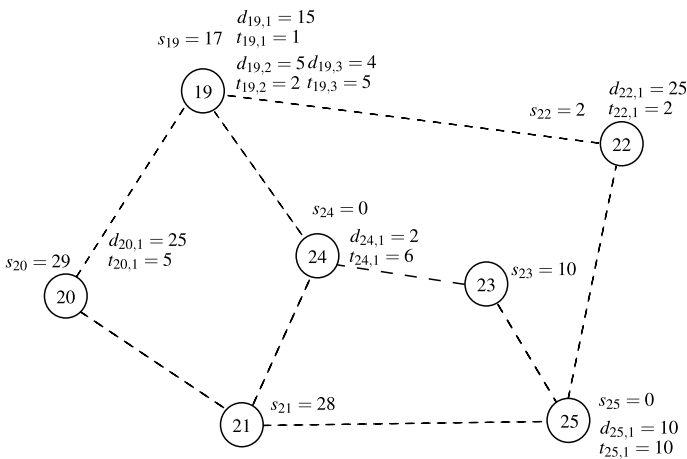


Fig. 2 Graph of the reduced problem

the p -th, $p = 1, \dots, \sigma_c$, positive demand in component $c \in C_0$ and $t_{cp} \in \mathbb{N}$ is its corresponding deadline (if $\sigma_c > 0$). Note that $t_{cp} < t_{cp+1}$. Set $E_0^B \subseteq \{(l, m) \in C_0 \times C_0 \mid l < m\}$ represents the blocked edges between the components. The length of the planning horizon is $T = \max\{t_{cp} \mid c \in C_0, p = 1, \dots, \sigma_c, \sigma_c > 0\}$. Note that the length of the planning horizon is identical to the one in the initial problem. Regarding the road clearance times, the same considerations as in Sect. 3.1 are applied. One blocked edge $(l, m) \in E_0^B$ represents a blocked road segment of predefined length that can be cleared within one period. Again, only one edge can be cleared per period.

A solution to the reduced problem consists only of the decision about which blocked edges to clear in which periods. This decision is represented by a sequence ω of $k \leq T$ edges in E_0^B such that each edge is contained at most once. Let $\omega(t)$ be the edge at position t of ω . Sequence ω of length k induces graphs G_0, \dots, G_T with $G_0 = (C_0, E_0^B)$, $G_t = (C_t, E_t^B)$ for each $t \leq k$, and $G_t = G_k$ for $T \geq t > k$. The sets C_t and E_t^B are defined as follows. Note that in the reduced problem the components are nodes as we have C_0, \dots, C_T as the node sets of graphs G_0, \dots, G_T . Edge $\omega(t)$ is specified by $(l, m) \in E_{t-1}^B$ with $l \in C_{t-1}$ and $m \in C_{t-1}$. Clearing edge $(l, m) \in E_{t-1}^B$ creates an unblocked path of length one between components l and m . Thus, the components l and m form a new component after clearing edge (l, m) which can be depicted as a single node in a graph. Thus, with every edge $\omega(t)$, specified by $(l, m) \in E_{t-1}^B$, the size of the graph G_t is reduced by one node and at least one edge compared to the former graph G_{t-1} . Formally this is depicted by deleting components l and m from the set of components and adding a new component to the set, this is $C_t = (C_{t-1} \setminus \{l, m\}) \cup \{n + t\}$ with n as the highest index of the components in C_0 . Accordingly, the supplies s_c as well as parameters σ_c, d_{cp} and t_{cp} are updated with every G_t . Supply s_{n+t} is the sum of supplies s_l and s_m . σ_{n+t} is the number of demands in components l and m . Note that demands with the same deadline are counted once only, because if in components l and m demands d_{lp} and d_{mp} with the same deadline $t_{lp} = t_{mp}$ exist, they are merged to one demand as described beforehand. Regarding the set of edges E_t^B , it is obvious that the cleared edge (l, m) is omitted. But the new component set C_t requires some additional changes regarding the edges as components l and m are replaced by component $n + t$. The set of blocked edges is defined as given in Eq. (1).

$$\begin{aligned}
 E_t^B = & \left\{ (i, j) \in E_{t-1}^B \mid i \neq l \wedge j \neq l \wedge i \neq m \wedge j \neq m \right\} \\
 & \cup \left\{ (i, n + t) \mid (i, j) \in E_{t-1}^B \setminus \{(l, m)\} \wedge (j = l \vee j = m) \right\} \\
 & \cup \left\{ (j, n + t) \mid (i, j) \in E_{t-1}^B \setminus \{(l, m)\} \wedge (i = l \vee i = m) \right\}
 \end{aligned} \tag{1}$$

The set of blocked edges E_t^B contains edges from set E_{t-1}^B that are not incident to either component $l \in C_{t-1}$ or $m \in C_{t-1}$. This is ensured by the first part of the union. The second part adds edge $(i, n + t)$ to set E_t^B if in E_{t-1}^B edges (i, l) or (i, m) exist. Edge $(j, n + t)$ is added to set E_t^B if in E_{t-1}^B edges (l, j) or (m, j) exist. Note that any edge from any edge set E_t^B represents an edge from edge set E_0^B of the underlying graph G_0 , a small example for this is given later in this section.

A solution ω changes the component structure over the planning horizon and therewith the supplies and demands of components as described beforehand.

Analogously to the initial problem, a solution ω to the reduced problem is feasible if all demands d_{cp} with $c \in C_0$ and $p = 1, \dots, \sigma_c$ (if $\sigma_c > 0$) are fulfilled not after their individual deadlines t_{cp} . This is the case if in every period each component has enough supply to cover the demands in that component that are due in that period or earlier. Thus, a feasible solution ω to the reduced problem changes the component structure in a way that in each graph G_t it holds for each component $c \in C_t$ that the supply s_c is equal or greater than the sum of the demands d_{cp} with $p = 1, \dots, \sigma_c$ (if $\sigma_c > 0$) which are due in period t or earlier because of their deadlines t_{cp} . More formally, in each graph G_1, \dots, G_T inequality (2) must hold for each component $c \in C_t$.

$$s_c \geq \sum_{\substack{p=1 \\ t_{cp} \leq t}}^{\sigma_c} d_{cp} \tag{2}$$

Note that like in the initial problem, not necessarily all blocked edges need to be cleared to obtain a feasible solution and that for some instances no feasible solution exists due to the strict deadlines.

A feasible solution can be evaluated by the length of its sequence ω . Like in the initial problem, this can be interpreted as the number of blocked edges or the number of periods that is necessary to clear parts of the road network so that all demands can be fulfilled. Again we seek a solution with a sequence of minimum length among the feasible ones.

In contrast to the initial problem, a solution to the reduced problem consists only of a sequence of blocked edges to clear and neglects specific deliveries of supply. Nevertheless, each feasible solution to the reduced problem can be easily transformed into an equivalent feasible solution to the initial problem with a sequence of same length and specific deliveries of supply. Regarding the sequence of blocked edges to clear, in Sect. 3.2.1 we explained how to replace edge set \mathcal{E} or, more specifically, edge set \mathcal{E}^B by an equivalent set of blocked edges E_0^B . Using this explanation, it is easily possible to convert a sequence ω of a solution to the reduced problem into an equivalent sequence Ω of a solution to the initial problem. An edge of sequence ω is unambiguously converted into an edge of sequence Ω if only one blocked edge exists between the corresponding pair of connected components in graph \mathcal{G} of the initial problem. If more than one blocked edge exist between a pair of components in graph \mathcal{G} , the corresponding edge of sequence ω can be converted into any of these edges for sequence Ω . Concerning the deliveries of supply, these can be derived from a solution to the reduced problem, which is only a sequence ω of blocked edges to clear, as follows. A feasible sequence ω induces graphs G_0, \dots, G_T with C_0, \dots, C_T and ensures that for each component $c \in C_t$ in each graph G_t inequality (2) holds so that it is guaranteed that in every period each component has enough supply to cover its demands that are due in that period or earlier. Each component $c \in C_t$ with $1 \leq t \leq T$ represents one or more components $c' \in C_{t-1}$ which got connected by clearing blocked edges. Finally, each component $c \in C_0$ represents a subset of nodes from set \mathcal{V} . Following this consideration for every component $c \in C_t$ with $1 \leq t \leq T$, it can be revealed which components of set C_0 and which nodes of set $\mathcal{V} = S \cup D \cup \mathcal{H}$ are represented by the respective component $c \in C_t$ in period

t . We refer to this as the structure of component set C_t . As stated above, a feasible sequence ω ensures that Eq. (2) holds for each component $c \in C_t$. Additionally, deliveries are immediately possible within a component. Specific deliveries can then be easily determined as follows. In period t a demand at demand node $j \in \mathcal{D}$ can be fulfilled by sending the requested supply from any arbitrary supply node $i \in \mathcal{S}$ within the same component to the respective demand node. If the available supply at a supply node is not sufficient, supply from another arbitrary supply node within the same component is delivered to the respective demand node. All deliveries can be performed in the period in which the respective demand is due. Generally, in order to ensure feasible deliveries, in period t only the demands that are due in that period or earlier are required to be fulfilled by deliveries of supply. Demands that are due in a later period must not be fulfilled.

3.3 Complexity results

In the section at hand, we determine the computational complexity of BSRCDP. We, first, derive a feasibility version of the problem defined in the previous sections.

BSRCDP: Given an undirected graph $G_0 = (C_0, E_0^B)$ with a number σ_c of demands for each $c \in C_0$, a demand amount d_{cp} and a deadline t_{cp} for each $c \in C_0$ and $p = 1, \dots, \sigma_c$ (if $\sigma_c > 0$), and a supply s_c for each $c \in C_0$, is there a sequence ω of a subset of edges such that for each position t in the sequence the total demand amount with deadlines not exceeding t in each connected component in $(C_0, \bigcup_{t'=1}^t \{\omega(t')\})$ does not exceed the total supply in this component?

The BSRCDP is closely related to the Steiner-Tree-Problem in Graphs (STG) as we will only sketch shortly in the following.

STG: Given an undirected weighted graph $G^S = (V^S, E^S, w^S)$ with a subset $R^S \subseteq V^S$ is there a subset $F \subseteq E^S$ such that in (V^S, F) there is a path between each pair of nodes in R^S and total weights of edges in F do not exceed a given threshold P ?

STG is strongly NP-complete even if all edge weights equal one, see Garey and Johnson (1979). In this special case, we require the cardinality of F not to exceed the given threshold P . It is now relatively easy to see that BSRCDP generalizes STG with unit weights. We omit a formal proof and shortly sketch the idea for the reduction instead. Given an instance of STG, we construct an instance of BSRCDP with the same node set where all but one node in R^S corresponds to a demand node in C_0 with no supply and unit demand with deadline P . The remaining node in R^S corresponds to a supply node in C_0 with no demand and a supply of $|R^S| - 1$. Nodes in $V^S \setminus R^S$ correspond to a node in C_0 with no supply and no demand. Finally, we set $E_0^B = E^S$. A connected component in (V^S, F) then corresponds to a connected component in $(C_0, \bigcup_{t'=1}^{|F|} \{\omega(t')\})$ if ω contains the edges in F in an arbitrary order. Thus, the answer to the instance of BSRCDP is yes if and only if the answer to the instance of STG is yes.

While the above already establishes NP-completeness of BSRCDP, we can derive a stronger complexity result by reduction from 3-PARTITION which is also known to be strongly NP-complete, see Garey and Johnson (1979).

3-PARTITION: Given $3m + 1$ integers a_1, \dots, a_{3m}, B with $\frac{B}{4} < a_i < \frac{B}{2}$ for each $i = 1, \dots, 3m$ and $\sum_{i=1}^{3m} a_i = mB$. Does there exist a partition of set $\{1, 2, \dots, 3m\}$ into m subsets A_1, \dots, A_m such that $\sum_{i \in A_l} a_i = B$ for each $l = 1, \dots, m$?

Theorem 1 *BSRCDP is strongly NP-complete even if each supply is binary, each demand amount is binary, and all demands have the same deadline.*

Proof For a given instance of 3-PARTITION, we construct the following instance of BSRCDP. We have a set $C = \{1, \dots, (4 + 2B)m\}$ of nodes. We distinguish between supply pooling nodes $1, \dots, 3m$, supply nodes $3m + 1, \dots, (3 + B)m$, demand pooling nodes $(3 + B)m + 1, \dots, (4 + B)m$, and demand nodes $(4 + B)m + 1, \dots, (4 + 2B)m$. Each demand node has a single demand with a demand amount of one and deadline $(3 + 2B)m$. Each supply node has a supply of one. Pooling nodes have neither positive supply nor demands.

Each supply node is connected by an edge to exactly one supply pooling node such that exactly a_i supply nodes are connected to supply pooling node i . Each demand node is connected by an edge to exactly one demand pooling node such that exactly B demand nodes are connected to each demand pooling node. Finally, each demand pooling node is connected to each supply pooling demand. This completes the description of the instance.

Clearly, the construction of the instance of BSRCDP can be done in pseudo-polynomial time. We claim that the answer to the instance of 3-PARTITION is yes if and only if the answer to the instance of BSRCDP is yes.

Consider a sequence ω constituting a yes-certificate for the instance of BSRCDP. The maximum deadline is $(3 + 2B)m$ and, hence, we can assume that ω has a length of no more than $(3 + 2B)m$. Clearly, the only edge incident to any supply node or to any demand node is in ω since otherwise demand remains unsatisfied. Hence, only $3m$ edges between supply pooling nodes and demand pooling nodes can be in ω . In order to deliver the total supply each supply pooling node must be connected to exactly one demand pooling node by one of these $3m$ edges in ω . Hence, these $3m$ edges imply a mapping of supply pooling nodes on demand pooling nodes. Since each demand is satisfied this mapping connects three supply pooling nodes with total supply of exactly B to each demand pooling node. This mapping, hence, constitutes a yes-certificate for the instance of 3-PARTITION.

It is easy to see that we can arrange a sequence ω of length $(3 + 2B)m$ if the answer to the instance of 3-PARTITION is yes. This concludes the proof. □

4 Mathematical model

BSRCD-MIP is based on the definition of the reduced problem. As explained in Sect. 3.2.2, a solution to the reduced problem is completely given by a sequence ω of length $k \leq T$ of blocked edges to clear. Such a sequence ω induces graphs G_0, \dots, G_T so that with every cleared edge $\omega(t)$ and thus with every period $t \leq k$ a new graph G_t

is associated. By clearing a blocked edge $\omega(t)$ between two components from graph G_{t-1} , an unblocked path of length one is created between them so that these components form a new component which is depicted as a single node in graph G_t . Accordingly, the supplies and demands of the components get updated. But graphs G_1, \dots, G_T are not known in advance as they depend on sequence ω which is the solution to the problem. As it is not trivial to implement unknown graphs G_1, \dots, G_T in a deterministic model, we use only the underlying graph $G_0 = (C_0, E_0^B)$ of the reduced problem for the formulation of BSRCD-MIP. Nevertheless, the information about which components of the underlying graph G_0 are connected by paths of unblocked edges in period t is needed to ensure the fulfillment of the demands. Thus, in BSRCD-MIP a path between every pair of components $c \in C_0$ and $q \in C_0$ with $c \neq q$ is determined. Note that this requires the assumption that the underlying graph G_0 is connected by blocked edges E_0^B . Binary variable $a_{cqe} \in \{0, 1\}$ is used to determine these paths. The variable takes value 1 if edge $e \in E_0^B$ is on the determined path from component $c \in C_0$ to component $q \in C_0$ and 0 otherwise. Note that edge $e \in E_0^B$ represents an edge $(l, m) \in E_0^B$ in BSRCD-MIP. The set of blocked edges incident to component $c \in C_0$ is represented by $\delta(c) \subseteq E_0^B$. Binary variable $z_{et} \in \{0, 1\}$ indicates if a blocked edge is cleared in period t . In this case, it takes value 1 and 0, otherwise. If up to period t all edges of the determined path between two components are cleared, these two components are connected from that period on. Additionally, for the flow conservation constraint auxiliary binary variable $r_{cqm} \in \{0, 1\}$ is needed which takes value 1 if component $m \in C_0$ is on the determined path from component $c \in C_0$ to component $q \in C_0$ and 0, otherwise. The aforementioned binary variables are used to determine a solution to the reduced problem, which is sequence ω . It can be derived from variable z_{et} . In the reduced problem, the fulfillment of the demands is ensured by comparing the updated supplies and demands in the components of every graph G_t using inequality (2) as it is described in Sect. 3.2.2. Having only graph G_0 in BSRCD-MIP this is not possible. In order to keep track of the fulfilled demands and the available supplies, continuous variable x_{cqt} defines an amount of supply from component $c \in C_0$ that is used for fulfilling demands in component $q \in C_0$ in period t . Supplies from component $c \in C_0$ can only be used to cover demands in component $q \in C_0$ in period t if between these two components an unblocked path exists in period t as described beforehand. Additionally, parameter M is a sufficient large number. BSRCD-MIP is defined as follows:

$$\min \sum_{t=1}^T \sum_{e \in E_0^B} z_{et} \tag{3}$$

$$\sum_{q \in C_0} \sum_{t=1}^{t_{cp}} x_{cqt} \geq \sum_{i=0}^p d_{ci} \quad c \in C_0; p = 1, \dots, \sigma_c; \sigma_c > 0 \tag{4}$$

$$\sum_{q \in C_0} \sum_{t=1}^T x_{cqt} \leq s_c \quad c \in C_0 \tag{5}$$

$$x_{cqt} \leq M * \left(1 - a_{cqe} + \sum_{\tau=1}^t z_{e\tau} \right) \quad c \in C_0; q \in C_0; t = 1 \dots T; e \in E_0^B; c \neq q \tag{6}$$

$$\sum_{e \in \delta(c)} a_{cqe} = 1 \quad c \in C_0; q \in C_0; c < q \tag{7}$$

$$\sum_{e \in \delta(m)} a_{cqe} = 2 * r_{cqm} \quad c \in C_0; q \in C_0; m \in C_0 \setminus \{c, q\}; c < q \tag{8}$$

$$\sum_{e \in \delta(q)} a_{cqe} = 1 \quad c \in C_0; q \in C_0; c < q \tag{9}$$

$$a_{cqe} = a_{qce} \quad c \in C_0; q \in C_0; c < q; e \in E_0^B \tag{10}$$

$$\sum_{e \in E_0^B} z_{et} \leq 1 \quad t = 1 \dots T \tag{11}$$

$$z_{et} \in \{0, 1\} \quad e \in E_0^B; t = 1 \dots T \tag{12}$$

$$x_{cqt} \geq 0 \quad c \in C_0; q \in C_0; t = 1 \dots T \tag{13}$$

$$a_{cqe} \in \{0, 1\} \quad c \in C_0; q \in C_0; e \in E_0^B; c \neq q \tag{14}$$

$$r_{cqm} \in \{0, 1\} \quad c \in C_0; q \in C_0; m \in C_0 \setminus \{c, q\}; c < q \tag{15}$$

Objective function (3) minimizes the number of cleared edges which is identical to the minimization of the length of sequence ω . Constraint (4) ensures that every demand is fulfilled up to its individual deadline. Remember from Sects. 3.2.1 and 3.2.2 that with $p = 1$ the demand with the lowest deadline, with $p = 2$ the one with the second lowest deadline and so forth are associated. The sum $\sum_{q \in C_0} \sum_{t=1}^{t_{cp}} x_{cqt}$ represents the amount of supply coming from all connected components $q \in C_0$ up to period t_{cp} . It should be at least as much as the demand of component $c \in C_0$ due in period t_{cp} or earlier. By constraint (5) it is forced that the supply of every component $c \in C_0$ is respected as over the planning horizon only as much supply as initially available in component $c \in C_0$ can be used to fulfill demands. Constraints (6)–(10) force that supply from component $c \in C_0$ can only be used to fulfill a demand in component $q \in C_0$ (with $c \neq q$) in period t if between these components at least one path of unblocked edges exists in that period. If at least one edge on the determined path between $c \in C_0$ and $q \in C_0$, which is determined by constraints (7)–(10), is not cleared up to period t , that is $\sum_{\tau=1}^t z_{e\tau} = 0$, then we have $x_{cqt} \leq 0$ in constraint (6). Thus, no supply from component $c \in C_0$ can be used to fulfill demands

in component $q \in C_0$ in period t . If instead all edges on the path are cleared up to period t , we have only $x_{cqt} \leq M$ and supply from component $c \in C_0$ can be used in component $q \in C_0$ in period t . As written above, constraints (7)–(10) determine a path between every pair of components $c \in C_0$ and $q \in C_0$ with $c \neq q$. Note that constraints (7)–(9) determine such paths for pairs c and q with $c < q$, but constraint (10) ensures that a path for pair c and q is also valid for pair q and c . Constraint (7) states that the path starts in component $c \in C_0$. Constraint (8) is a flow conservation constraint which forces the path to enter and leave another component $m \in C_0$ if component m is on the determined path from component $c \in C_0$ to component $q \in C_0$ which is indicated by $r_{cqm} = 1$. Constraint (9) ensures that the path ends in component $q \in C_0$. The assumption that at most one edge is cleared per period is implemented by constraint (11). The domain of the variables is given by constraints (12)–(15).

In order to strengthen the model formulation, some valid inequalities can be incorporated. Some computational tests indicate that the valid inequalities given below can decrease the computational times.

$$\sum_{t=1}^T z_{et} \leq 1 \quad e \in E_0^B \tag{16}$$

If a blocked edge needs to be cleared, it is sufficient to clear it just once during the planning horizon. As the objective function minimizes the sum of cleared edges, no edge will be cleared more than once. But in order to strengthen the formulation, constraint (16) is added which additionally forces that an edge is cleared at most once.

$$\sum_{e \in E_0^B} z_{et} \geq \sum_{e \in E_0^B} z_{e(t+1)} \quad t = 1 \dots (T - 1) \tag{17}$$

Furthermore, there might be more than one optimal solution. The objective function minimizes the number of edges to be cleared. Given the minimum number of edges, there might be a solution—if deadlines allow for it—in which at least in one period no edge is cleared, but in the following periods blocked edges are cleared. To overcome this, inequality (17) is introduced. It states that clearing an edge in period $t + 1$ is only possible if in period t an edge is cleared as well.

5 Branch and bound algorithm

In this section, our exact B&B algorithm to the reduced problem of BSRCDP is presented. In Sect. 5.1, the basic idea is introduced, and in Sect. 5.2, the basic idea is brought into the structure of a B&B algorithm. The branching is described in detail in Sect. 5.3, whereas the bounding procedure of the B&B algorithm is explained in Sect. 5.4.

5.1 Basic idea

The basic idea of the algorithm is to grow a sequence of blocked edges to clear. In an initial step for each component $c \in C_0$ of graph G_0 which has at least one demand, that is $\sigma_c \geq 1$, it is checked if the available supply in this component is sufficient to fully cover the demands in this component. Recall from Sect. 3.2.1 that s_c can take value zero if no supply is available in the whole component. For each component which has not enough supply to cover all demands, that is $s_c < \sum_{p=1}^{\sigma_c} d_{cp}$, the earliest deadline t_{cp} at which the total amount of the demands, which are due at deadline t_{cp} or at an earlier deadline, exceeds the available supply in this component is determined and noted as \tilde{t}_c . Note that if no component with $s_c < \sum_{p=1}^{\sigma_c} d_{cp}$ exists, the optimal solution is a sequence ω of length zero. Otherwise, out of deadlines \tilde{t}_c the lowest deadline \tilde{t} is determined by Eq. (18).

$$\tilde{t} = \min_{\substack{c \in C_0 \\ s_c < \sum_{p=1}^{\sigma_c} d_{cp}}} \tilde{t}_c \tag{18}$$

The corresponding component $c \in C_0$ with the lowest deadline $\tilde{t} = \tilde{t}_c$ is noted as component \tilde{c} and can be interpreted as the component in which the amount of the demands exceeds the supply next. Therefore, \tilde{c} is the next component which needs to get access to a subset of other already-connected components by clearing paths of blocked edges between component \tilde{c} and the components of such a subset. The components of such a subset need to have together a sufficient surplus of supply in period \tilde{t} to cover the remaining demand in component \tilde{c} that is due in period \tilde{t} . Note that such a subset of other components can consist of only one component. Additionally, the surplus of supply of a component in period \tilde{t} is the amount of supply that is left after subtracting the demands of that component which are due in period \tilde{t} or earlier. Demands of that component with deadlines later than \tilde{t} might be fulfilled by supply of other components later. The subset of blocked edges forming the paths to be cleared between component \tilde{c} and the components of such a subset is referred to as partial sequence in the following. Generally, several subsets of components with a sufficient surplus of supply can exist and consequently several partial sequences of blocked edges to clear can exist to connect component \tilde{c} with these subsets. Moreover, even different partial sequences can exist to connect component \tilde{c} with one specific subset of components. But a partial sequence of blocked edges to clear can only contain as many blocked edges as periods that are available to connect component \tilde{c} with other components. $\tilde{\omega}$ is the sequence of blocked edges which have been cleared so far and $\tilde{\delta}$ is its length. As the clearance time of one blocked edge is one period as stated in Sect. 3.2.2, $\tilde{\delta}$ is also the number of periods which have been passed clearing blocked edges. Note that for the initialization the sequence $\tilde{\omega}$ is empty and thus $\tilde{\delta}$ is equal to zero because no edges have been cleared at the beginning of the planning horizon. The number of available periods to connect component \tilde{c} is thus computed by $\tilde{t} - \tilde{\delta}$. Out of all partial sequences with length equal or smaller than $\tilde{t} - \tilde{\delta}$, one is selected. By clearing the blocked edges of the selected partial sequence, an unblocked path between the components adjacent to

the blocked edges of the selected partial sequence emerges. Applying the definition of a component from Sect. 3.2.1, the components along the emerged path can be summarized and depicted as a single component or, more specifically, node with the corresponding update of blocked edges as well as supplies, demands and deadlines in a new graph as described in Sect. 3.2.2. Note that the transformation procedure in Sect. 3.2.2 is only described for one cleared edge. If a partial sequence contains more than one blocked edge to clear, the transformation procedure has to be repeated as many times as the cardinality of the selected partial sequence. $\tilde{\omega}$ is updated by adding the selected partial sequence, and its length is added to $\tilde{\delta}$. The procedure described beforehand is repeated for the new graph in order to connect the component which demands exceed its supply next with a subset of components which have a sufficient surplus of supply by clearing another partial sequence of blocked edges.

As described beforehand, our approach is to connect the component which needs supply most urgently first before connecting the second most urgent component. This is justified as follows. If, although all feasible partial sequences are considered, the most urgent component cannot get connected to a subset of components with a sufficient surplus of supply, no feasible solution exists, and considering any other less urgent component is unnecessary.

5.2 General structure

Bringing the basic idea into the structure of a B&B algorithm, with root node P_0 of the search tree always graph $G_0 = (C_0, E_0^B)$ of the reduced problem and with each other node P_b of the search tree a graph $G_t = (C_t, E_t^B)$ is associated. For the sake of clarity, we name each node of the search tree B&B node in the following. Additionally, for every B&B node \tilde{t} , \tilde{c} , $\tilde{\omega}$ and $\tilde{\delta}$ exist. \tilde{t} and \tilde{c} are determined as described in Sect. 5.1. As no edge has been cleared at the beginning of the planning horizon $\tilde{\omega}$ is empty for P_0 and thus the value of $\tilde{\delta}$ is zero for P_0 . Every B&B node P_b has as many branches as feasible partial sequences exist to connect component \tilde{c} within $\tilde{t} - \tilde{\delta}$ periods to subsets of other components which each has a sufficient surplus of supply in period \tilde{t} . Obviously, each branch represents one partial sequence of blocked edges to clear and vice versa. Thus, branch and partial sequence can be used interchangeably. In Sect. 5.3 it is described in detail how the branches of a B&B node are obtained. The search strategy is a depth first search. The algorithm starts without a feasible solution, because as stated in Sects. 3.1 and 3.2.2, it is not known in advance if a feasible solution exists or not. The aim of conducting a depth first search is to find a feasible solution as fast as possible which then can be used for further bounding of the whole search process. The bounding procedure of the B&B algorithm is described in Sect. 5.4. In order to obtain a good feasible solution, i.e., a solution with a number of cleared edges as small as possible, always the branch representing the shortest partial sequence is selected. Taking the parent B&B node, its associated graph and the selected partial sequence of blocked edges to clear, the child B&B node and the associated graph of the child B&B node are formed by (repeatedly, if the partial

sequence contains more than one blocked edge) applying the transformation described in Sect. 3.2.2. For the child B&B node $\tilde{\omega}$ is updated by adding the selected partial sequence and $\tilde{\delta}$ is updated by adding the length of the selected partial sequence. The procedure of identifying component \tilde{c} and period \tilde{t} as well as determining all feasible partial sequences, selecting one and forming the next B&B node as described beforehand is repeated. If for a B&B node P_b or, more specifically, in its associated graph no component with uncovered demand, that is $s_c \geq \sum_{p=1}^{\sigma_c} d_{cp}$ for each $c \in C_t$, can be identified, a feasible solution is found and this B&B node does not have to be branched. The sequence $\tilde{\omega}$ of this B&B node P_b is a feasible solution ω to the reduced problem. For each graph induced by sequence ω Eq. (2) holds and thus sequence ω is a feasible solution. By considering all possible and feasible partial sequences to connect a component \tilde{c} with subsets of components with sufficient surplus of supply during the search process, it is ensured that an optimal solution is found if a feasible solution exists. If no feasible solution exists, all B&B nodes and branches are fathomed by the bounding rules described in Sect. 5.4 without having found a feasible solution.

Figure 3 shows the complete search tree of the B&B algorithm to the example given in Fig. 2. The number at each branch is the length of its partial sequence, and the letter is an index for the partial sequence. Each partial sequence can be found in the second column of Table 1 by this index. Note that the edges in Table 1 refer to edge set E_0^B because any edge from an edge set E_t^B of a graph G_t can be represented as an edge of graph G_0 as stated in Sect. 3.2.2. The first column of the table contains the B&B nodes. In the third column sequences $\tilde{\omega}$ for every B&B node P_b are shown.

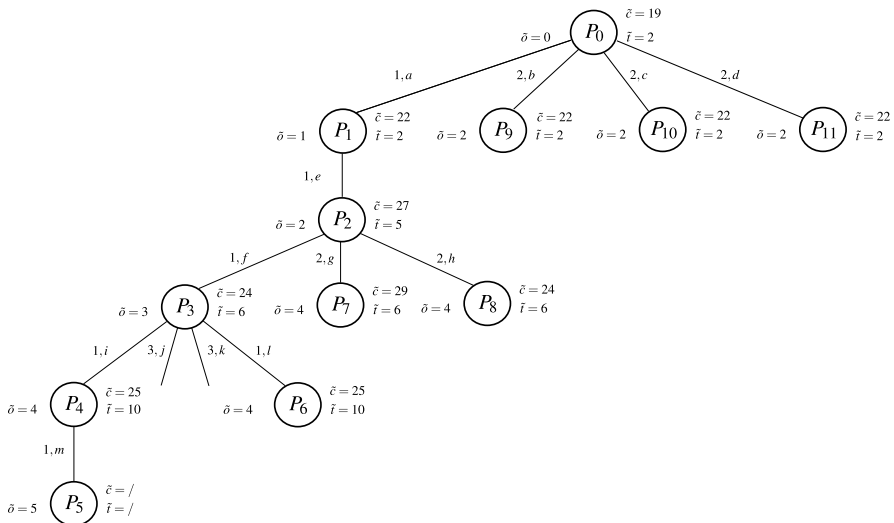


Fig. 3 Search tree of the example

Table 1 Branches and sequences $\tilde{\omega}$ of the example

Branches	$\tilde{\omega}$
P_0 a: ((19, 20)) b:((19, 24), (23, 24)) c:((19, 24), (21, 24)) d:((19, 20), (20, 21))	–
P_1 e: ((19, 22))	((19, 20))
P_2 f: ((20, 21)) g:((19, 24), (21, 24)) h:((21, 25), (22, 25))	((19, 20), (19, 22))
P_3 i:((21, 24)) j:((23, 24), (23, 25), (21, 25)) k:((21, 24 , (21, 25), (23, 25)) l: ((23, 24))	((19, 20), (19, 22), (20, 21))
P_4 m: ((23, 25))	((19, 20), (19, 22), (20, 21), (21, 24))
P_5 –	((19, 20), (19, 22), (20, 21), (21, 24), (23, 25))
P_6 –	((19, 20), (19, 22), (20, 21), (23, 24))
P_7 –	((19, 20), (19, 22), (19, 24), (21, 24))
P_8 –	((19, 20), (19, 22), (22, 25), (21, 25))
P_9 –	((19, 24), (23, 24))
P_{10} –	((19, 24), (21, 24))
P_{11} –	((19, 20), (20, 21))

5.3 Branching

For each B&B node P_b , \tilde{t} , \tilde{c} , $\tilde{\omega}$ and $\tilde{\delta}$ are determined as described beforehand. Given these information and a graph G_t which is associated with P_b , all feasible branches or, more specifically, partial sequences can be computed for B&B node P_b as follows.

Step 1 Determine all necessary subsets of components in graph G_t which have a sufficient surplus of supply to cover the remaining demand in component \tilde{c} that is due in period \tilde{t} .

Step 2 For each of these subsets

(a) and for each component in a subset compute all elementary paths with feasible length smaller or equal $\tilde{t} - \tilde{\delta}$ from that component to component \tilde{c} . If for at least one component of the treated subset no feasible path can be computed, Step 2(b) can be skipped for the treated subset.

(b) determine all combinations of paths computed in Step 2(a) so that in each combination all components of the treated subset are connected to component \tilde{c} . If in a combination one edge appears several times, only one copy is kept. Additionally, if after keeping only one copy of edges appearing several times the number of edges in such a combination is greater than $\tilde{t} - \tilde{\delta}$, this combination is infeasible and can be discarded. Each feasible combination represents a partial sequence which is a branch of B&B node P_b . If the treated subset contains only one component, obviously no combinations need to be determined. Instead, each feasible elementary path determined in Step 2(a) for such a subset is a partial sequence.

Step 3 If in graph G_t component \tilde{c} is the only component left with uncovered demand, out of all feasible partial sequences only one with a minimum number of blocked edges is kept and all other partial sequences are discarded.

In *Step 1* let C_t^+ be the set of components in graph G_t with a surplus of supply in period \tilde{t} . So for each component $c \in C_t^+$ inequality (19) must hold.

$$s_c > \sum_{p=1}^{\sigma_c} d_{cp} \quad t_{cp} \leq \tilde{t} \tag{19}$$

Each of these components can contribute to cover the demand in component \tilde{c} that is due in period \tilde{t} . In order to find all necessary feasible partial sequences and thus to guarantee the optimality of the B&B algorithm, initially all subsets of C_t^+ except the empty set are considered. Out of these the subsets whose combined surplus of supply is smaller than the uncovered demand in component \tilde{c} that is due in period \tilde{t} are not further considered for the computation of the partial sequences. The remaining subsets have a sufficient surplus of supply, but considering all of them could lead to the computation of superfluous partial sequences. Therefore, from the remaining subsets for each subset with cardinality greater than one is checked if at least one of its own strict subsets has a combined surplus of supply which is already greater or equal than the uncovered demand in component \tilde{c} that is due in period \tilde{t} . All subsets having at least one of these strict subsets are removed from the remaining subsets. The remaining subsets are forwarded to Step 2.

The aim of *Step 2* is to generate partial sequences of blocked edges to clear for each subset which connect component \tilde{c} with the components of the respective subset.

In *Step 2(a)* for each component of the treated subset, all elementary paths with feasible length from that component to component \tilde{c} are computed as a preparation for Step 2(b). It is not sufficient to compute only the shortest path of blocked edges to clear for each component because it can be easily shown that in order to obtain a feasible solution it might be necessary to clear a longer path of blocked edges between component \tilde{c} and a component with surplus of supply. A subset cannot be considered for generating partial sequences in Step 2(b) if for at least one of its components no feasible path to component \tilde{c} can be generated, because of this connecting all components of this subset to component \tilde{c} is impossible.

In *Step 2(b)* the partial sequences are generated as described above.

The aim of *Step 3* is to avoid generating branches in the search tree which cannot lead to an optimal solution. If component \tilde{c} is the last component that needs to be connected to a subset of components with a sufficient surplus of supply and at least one feasible partial sequence of blocked edges to clear exist, the resulting child B&B node does not need to get further branched regardless the length of the chosen partial sequence as no component in the graph associated with the child B&B node will have a lack of supply and will need to get connected to other components. Thus, as we seek a sequence of minimum length for the overall solution, it is optimal to consider only the partial sequence with minimum length in this case.

5.4 Bounding

As explained in Sect. 5.2 the B&B algorithm starts without a feasible solution. Before obtaining a feasible solution during the search process which can be used for further bounding, the following bounding rules are used to fathom branches or B&B nodes.

A branch of a B&B node P_b is fathomed if the current length \bar{o} of sequence $\tilde{\omega}$ of B&B node P_b plus the number of edges of the respective branch is larger than an initial upper bound. The value of the summation can be interpreted as the number of blocked edges cleared after choosing the respective branch. The initial upper bound is computed as $|C_0| - 1$. A cleared sequence ω of length $|C_0| - 1$ forming a spanning tree connects all components $c \in C_0$ directly or indirectly and thus all demands can be covered if a feasible solution exists. Clearing an additional blocked edge only deteriorates the objective value, that is the length of sequence ω .

A B&B node cannot be branched and is therefore fathomed if no partial sequence exists, i.e., no partial sequence with length smaller or equal $\tilde{t} - \bar{o}$ that connects component \tilde{c} with a subset of components with sufficient surplus in period \tilde{t} exists.

A B&B node P_b with $b \neq 0$ is also fathomed if the structure of the component set C_t of graph G_t which is associated with B&B node P_b is identical to the structure of component set $C_{t'}$ of a graph $G_{t'}$ which is associated with a B&B node $P_{b'}$ with $b' < b$. Note that this can be observed even if different sequences $\tilde{\omega}$ have been cleared so far. The structure of component set C_t with $t > 0$ reveals which components of graph G_0 of the root B&B node P_0 are depicted by a single component $c \in C_t$ (see Sect. 3.2.2). If the component structure of B&B node P_b is identical to the one of $P_{b'}$, the sub-tree of B&B node P_b would be identical to the one of B&B node $P_{b'}$ which has already been fully explored. Therefore, B&B node P_b can be fathomed.

After obtaining a feasible sequence ω during the search process, additional bounding rules are applied.

A branch of a B&B node is fathomed if the current length \bar{o} of sequence $\tilde{\omega}$ of B&B node P_b plus the number of edges of the respective branch is larger than the length of the shortest feasible sequence found so far. Obviously, this bounding rule is similar to the first one presented in this section. Only the initial upper bound $|C_0| - 1$ is replaced by a regularly better upper bound given by the shortest feasible sequence found so far.

A B&B node P_b is fathomed if a lower bound computed for this B&B node is equal or larger than the length of the shortest feasible sequence found so far. A lower bound for B&B node P_b is computed as follows. To the number of cleared edges \bar{o} of B&B node P_b , the number of components with in total less supply than demand, that are all components $c \in C_t$ in graph G_t associated with B&B node P_b for which inequality $s_c < \sum_{p=1}^{\sigma_c} d_{cp}$ hold, is added. Each of these components needs to get connected to at least one other component. For this at least one edge per component with in total less supply than demand has to be cleared. Additionally, the minimum length of the shortest paths from each of these components with in total less demand than supply to each nearest component with in

total more supply than demand, that is $s_c > \sum_{p=1}^{\sigma_c} d_{cp}$, is added. Finally, value one is subtracted to avoid a double counting of an edge.

6 Computational study

In this section, the main results of a comprehensive computational study are presented. In Sect. 6.1 the performances of BSRCD-MIP and of the B&B algorithm are compared. The performance of the B&B algorithm with respect to bigger instances and to the structure of the tested instances is further examined in Sect. 6.2.

BSRCD-MIP is implemented in AMPL and solved by the commercial solver Gurobi (version 8.1). The B&B algorithm is coded in C. All studies are conducted on a computer equipped with an Intel(R) Core(TM) i5-3470 CPU (3.2 GHz), 16 GB memory and Windows 7 Professional (64 bit).

In our computational study, we randomly generated instance sets for 8, 10, 12, 16, 20 and 25 components. Regarding the parameter settings, we consider different parameters with different levels. The first parameter is the number of components with a positive supply. The total number of demands is the second parameter. The third parameter is the number of components to which the demands are randomly assigned during instance generation. For each of the three aforementioned parameters, two different levels are considered. The fourth parameter is the density of the graph. For the instance set with 8 components, only one level of density is considered. For the instance sets with 10 and 12 components, two different levels are taken into account while for the instance sets with 16, 20 and 25 components three different levels are considered. One can refer to Table 4 for more details about these parameter settings. Dozens of instances for each combination of these parameters, which totals to several hundreds of instances, are generated for each number of components. For each of these instances, the B&B algorithm without all bounding rules is run for 10 s and the number of feasible solutions obtained, if any, is counted. Obviously, the B&B algorithm also detects many instances as infeasible due to the strict deadlines of the demands within 10 s. These instances are not considered for the test bed. The remaining instances with varying numbers of feasible solutions (including the instances with no feasible solutions returned by the B&B algorithm within 10 s) are sorted with respect to the number of feasible solutions in descending order. Then, for each number of components and each combination of parameter settings, the top five instances are assigned to the first instance group, while the bottom five are assigned to the second instance group. Results regarding the two instance groups are discussed in Sect. 6.2. For more details on the parameter settings and the instance structure, respectively, we refer to Sects. 6.1 and 6.2. Additionally, a detailed documentary of the instance structure can be inquired from the authors. Note that the results are presented in a highly aggregated form in the following sections.

6.1 Comparison of BSRCD-MIP and the B&B algorithm

The aim of the computational study presented in this section is to demonstrate the computational advantage of the B&B algorithm over BSRCD-MIP which is solved including the presented valid inequalities.

Table 2 shows the main results of this study. The first column gives the number of components of the tested instances. 80 (160/160) instances with 8 (10/12) components are tested using a time limit of 1800 s for each instance. In Table 2 three columns are depicted for BSRCD-MIP and the B&B algorithm. The first out of these three columns gives the number of instances solved to optimality within the time limit. The second and third columns depict the average computational time given in seconds and the percentage standard deviation of the computational time. Both are computed based on the solved instances. The percentage standard deviation is not given if the results would cause a division by zero.

Comparing BSRCD-MIP with the B&B algorithm, we can state that the B&B algorithm clearly outperforms BSRCD-MIP. Whereas all of the instances with 8 components and most of the instances with 10 components can be solved by BSRCD-MIP, it is obvious that the number of instances solved within the time limit decreases significantly for instances with 12 components. Computational tests show that instances with more than 12 components are generally not solvable within the time limit and are thus not depicted. Throughout the tested instances regularly a high standard deviation in terms of computational time can be observed for BSRCD-MIP. Further analysis reveals that at least 90% (50%) of the instances with 10 (12) components are solved in less or equal 10 s. 10% (50%) or less of the instances have partially high computational times which cause the high standard deviation. In contrast, the B&B algorithm is able to solve all instances shown in Table 2 to optimality instantly.

6.2 Performance of the B&B algorithm

In this section, the performance of the B&B algorithm is examined in terms of the number of solved instances and of the computational time for different instance sizes. Additionally, the influence of different instance groups as well as the effect of different parameter settings and the quality of feasible solutions obtained during a run of the B&B algorithm are investigated.

The main results are depicted in Table 3. The first column of Table 3 gives the number of components. The second column presents the number of solved instances.

Table 2 Results for BSRCD-MIP and the B&B algorithm

$ C_0 $	BSRCD-MIP			B&B		
	# Solved	Time	SD time	# Solved	Time	SD time
8	80	0.0	-	80	0.0	-
10	158	26.2	592%	160	0.0	-
12	114	68.1	336%	160	0.0	-

Table 3 Aggregated results for the B&B algorithm

$ C_0 $	# Solved	Time	Time first	Gap	SD time (%)	SD time first (%)	SD gap (%)
Instance group 1							
16	120	2.5	0.6	0.6	340	433	133
20	84	88.7	0.8	0.9	312	325	89
25	17	485.4	1.9	0.9	122	126	100
Instance group 2							
16	112	61.8	29.7	0.3	364	422	200
20	52	155.4	26.7	0.9	198	252	89
25	5	660.1	15.1	0.8	107	156	88

Columns three and four give the average computational time and the average computational time until a first feasible solution is found. Both are computed based on the number of solved instances and are given in seconds. The fifth column depicts the average absolute gap between the objective value of the first feasible solution and the optimal objective value for the solved instances. Columns six to eight of Table 3 show the percentage standard deviation for the computational time, the computational time until the first feasible solution and the gap based on the solved instances. The time limit is 1800 s for each instance.

Regarding the number of instances solved to optimality, all 120 tested instances with 16 components are solved in the first instance group and in the second instance group a vast majority (112) of 120 instances with 16 components is solved. In terms of the instances with 20 components, in the first instance group 84 out of 120 instances are solved, whereas in the second instance group 52 out of 120 instances are solved. Regarding instances with 25 components, only 17 instances in the first instance group and 5 instances in the second instances group are solved at a total of 120 instances per group. Out of these figures it appears that for instances with 16, 20 as well as 25 components significantly more instances are solved for the first instance group than for the second instance group. Obviously, instances of the second instance group seem to be harder to solve. This observation is supported by the average computational times. In the first instance group, instances with 16 components are solved in 2.5 s on average, whereas in the second instance group the average computational time is 61.8 s. Similar results are obtained for instances with 20 and 25 components where the average computational times are 88.7 s and 485.4 s for the first instance group and 155.4 s and 660.1 s for the second instance group. However, studying the percentage standard deviation of the computational times reveals that on the first glance the computational times vary intensely. But further analysis shows that in the first instance group 95% (72%) and in the second instance group 87% (56%) of the tested instances with 16 (20) components are solved in 10 s or less.

Considering the quality of the feasible solutions obtained for an instance during a run of the B&B algorithm, we first investigate the gap as well as its percentage standard deviation depicted in the fifth and eighth column of Table 3. It appears

that the gap is quite small as for each number of components and for both instance groups the average gap does not exceed 0.9 which is approximately only one blocked edge. Although the percentage standard deviation of the gap is relatively high, its absolute values seldom exceed one blocked edge. Thus, a first conclusion is that the B&B algorithm is able to find (first) feasible solutions of high quality in terms of the objective value. As the test bed for the computational study contains mostly instances for which at least one feasible solution can be obtained within 10 s of computational time, it is not remarkable that the average computational time until a first feasible solution is found is comparatively low. But like for the average computational times, the average computational times until a first feasible solution are higher for the second instance group and the percentage standard deviations are comparatively high as one can see in the fourth and seventh column of Table 3. Nevertheless, not only because of the promising results for the first feasible solutions, one assumption is that the B&B algorithm used as a heuristic, which terminates as soon as a feasible solution is found, is deployable for bigger instances. Computational tests disprove this assumption as out of 3120 instances with 25 components only 573 instances are immediately detected as infeasible, and for 590 instances a first feasible solution is found within the time limit of 600 s.

In order to analyze the influence of different parameter settings, we refer to Table 4. The instances with 16, 20 and 25 components (120 instances each) presented in Table 3 are shown in a less aggregated way in Table 4. For instances with

Table 4 Results of the parameter analysis for the B&B algorithm

C ₀	%	# Solved		Time	
		Instance group 1	Instance group 2	Instance group 1	Instance group 2
16					
Supply	20	60	60	1.3	23.8
	40	60	52	3.8	105.6
Density	15	40	40	0.0	0.0
	20	40	40	1.7	13.2
	25	40	32	6.0	199.8
20					
Supply	20	53	33	57.5	100.1
	40	31	19	141.9	251.5
Density	15	40	37	35.4	114.7
	20	25	12	112.3	281.7
	25	19	3	169.7	152.8
25					
Supply	20	15	5	346.5	660.1
	40	2	0	1527.3	–
Density	15	11	3	468.0	507.4
	20	6	2	517.3	889.1
	25	0	0	–	–

16, 20 and 25 components two of the four introduced parameters, namely the number of components with a positive supply given as a percentage of $|C_0|$ and the density of the graph given as a percentage of the maximum number of possible edges in the graph of the reduced problem, are depicted in the first two columns. The maximum number of possible edges is $\frac{|C_0|*(|C_0|-1)}{2}$. Although the road system mostly is a planar graph, BSRCDP is not restricted to planar graphs so that non-planar road systems with bridges and tunnels can also be handled. Columns three and four give the number of solved instances for each instance group, and columns five and six depict the average computational times given in seconds. Following observations can be made. One main observation is that the density of the graph has a comparatively large influence on the number of solved instances as well as on the average computational times. Regarding the number of solved instances with 16 components, the density does not have a real influence, but it has on the computational time as the average computational time increases from 0.0 to 6.0 s in the first instance group and from 0.0 to 199.8 s in the second instance group, respectively. With respect to the instances with 20 and 25 components, the increasing density regularly causes a comparatively strong decrease in the number of solved instances in both instance groups. Correspondingly, the average computational time of these instances regularly increases significantly with the density of the graph. Another observation is that the number of components with supply has a considerably impact on the performance of the B&B algorithm. Regarding the number of solved instances, it stands out that with exception of the instances with 16 components of the first instance group the number of solved instances decreases significantly when 40% instead of 20% of the components have a positive supply. A similar effect is observable for the average computational times of both instance groups. A doubling of the percentage of the number of components with positive supply leads to an increase of the average computational times of at least two times. Note that while the percentage of components with positive supply is doubled, the amount of total supply but also the amount of total demand is doubled. Also the other two parameters, namely the number of demands and the number of components to which these demands are randomly assigned, are investigated. But as we observe no significant influence on the number of solved instances or the average computational times, the results are not depicted.

7 Conclusion

This paper investigates a basic scenario of the deterministic simultaneous problem of disaster road clearance and relief distribution taking strict deadlines for the fulfillment of demands into account. Two equivalent problem statements with different graph representations are introduced. Aspects of the inherent structure of the problem are used to transform an initial problem into a reduced problem with a smaller graph representation. One MIP model based on the reduced problem is presented. Further focusing on the reduced problem, we propose a B&B-based exact solution

approach. In a computational study, the performances of the MIP model and the B&B algorithm are tested.

A first result of the computational study is that the B&B algorithm outperforms the MIP model. Second, next to the number of components, the number of components having a strictly positive supply and the density of the graph have a relatively strong impact on the performance of the B&B algorithm. A third result is that the first feasible solution found by the B&B algorithm tends to be of relatively good quality. The deployment of a heuristic based on the B&B algorithm for bigger instances, which terminates with a first feasible solution, lacks in stable computational times.

This paper provides algorithmic ideas and structural insights rather than a decision support system that can be readily applied. Thus we encourage other researchers to step further ahead and develop such a system. A further aspect of future research is the additional consideration of multiple work teams for road clearance and cooperation between them. On the one hand, cooperation of work teams leads to faster clearance of single blocked roads, but on the other hand such cooperation hampers the simultaneous clearance of different blocked roads due to work teams working cooperatively on a blocked road at the same time. This aspect is known from the literature focusing only on road clearance, but to the best of our knowledge it is only scarcely investigated for the simultaneous problem. Another field of future research is the investigation of other factors than the distribution of relief goods that affect the road clearance activities. One factor is the accessibility of restoration crews to the disrupted sectors of critical infrastructures.

Acknowledgements Open Access funding provided by Projekt DEAL. The work of Alf Kimms and Denis Olschok was financially supported by the Deutsche Forschungsgemeinschaft Grant KI 1272/9-1.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Anaya-Arenas AM, Renaud J, Ruiz A (2014) Relief distribution networks: a systematic review. *Ann Oper Res* 223:53–79. <https://doi.org/10.1007/s10479-014-1581-y>
- Cavdaroglu B, Hammel E, Mitchell JE, Sharkey TC, Wallace WA (2013) Integrating restoration and scheduling decisions for disrupted interdependent infrastructure systems. *Ann Oper Res* 203:279–294. <https://doi.org/10.1007/s10479-011-0959-3>
- Çelik M (2016) Network restoration and recovery in humanitarian operations: framework, literature review, and research directions. *Surv Oper Res Manag Sci* 21(2):47–61. <https://doi.org/10.1016/j.sorms.2016.12.001>
- Çelik M, Ergun Ö, Keskinocak P (2015) The post-disaster debris clearance problem under incomplete information. *Oper Res* 63(1):65–85. <https://doi.org/10.1287/opre.2014.1342>

- Coppola DP (2015) Introduction to international disaster management, 3rd edn. Butterworth-Heinemann, Oxford
- de la Torre LE, Dolinskaya IS, Smilowitz KR (2012) Disaster relief routing: integrating research and practice. *Socio-Econ Plan Sci* 46(1):88–97. <https://doi.org/10.1016/j.seps.2011.06.001>
- Fikar C, Hirsch P, Nolz PC (2018) Agent-based simulation optimization for dynamic disaster relief distribution. *Cent Eur J Oper Res* 26(2):423–442. <https://doi.org/10.1007/s10100-017-0518-3>
- Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman and Company, New York
- González AD, Dueñas-Osorio L, Medaglia AL, Sánchez-Silva M (2016a) The time-dependent interdependent network design problem (td-INDP) and the evaluation of multi-system recovery strategies in polynomial time. In: Huang H, Li J, Zhang J, Chen J (eds) Proceedings of the 6th Asian-Pacific symposium on structural reliability and its applications. Tongji University Press, Shanghai, pp 544–550
- González AD, Dueñas-Osorio L, Sánchez-Silva M, Medaglia AL (2016b) The interdependent network design problem for optimal infrastructure system restoration. *Comput Aided Civ Infrastruct Eng* 31(5):334–350. <https://doi.org/10.1111/mice.12171>
- Guha-Sapir D, Hoyois P, Wallemaecq P, Below R (2016) Annual disaster statistical review 2016: the numbers and trends. CRED, Brussels
- Liberatore F, Ortuño MT, Tirado G, Vitoriano B, Scaparra MP (2014) A hierarchical compromise model for the joint optimization of recovery operations and distribution of emergency goods in humanitarian logistics. *Comput Oper Res* 42:3–13. <https://doi.org/10.1016/j.cor.2012.03.019>
- Morshedlou N, González AD, Barker K (2018) Work crew routing problem for infrastructure network restoration. *Transp Res Part B Methodol* 118:66–89. <https://doi.org/10.1016/j.trb.2018.10.001>
- Nurre SG, Sharkey TC (2014) Integrated network design and scheduling problems with parallel identical machines: complexity results and dispatching rules. *Networks* 63(4):306–326. <https://doi.org/10.1002/net.21547>
- Nurre SG, Cavdaroglu B, Mitchell JE, Sharkey TC, Wallace WA (2012) Restoring infrastructure systems: an integrated network design and scheduling (INDS) problem. *Eur J Oper Res* 223(3):794–806. <https://doi.org/10.1016/j.ejor.2012.07.010>
- Özdamar L, Ertem MA (2015) Models, solutions and enabling technologies in humanitarian logistics. *Eur J Oper Res* 244(1):55–65. <https://doi.org/10.1016/j.ejor.2014.11.030>
- Özdamar L, Aksu DT, Ergünes B (2014) Coordinating debris cleanup operations in post disaster road networks. *Socio-Econ Plan Sci* 48(4):249–262. <https://doi.org/10.1016/j.seps.2014.08.001>
- Ransikarbum K, Mason SJ (2016) Multiple-objective analysis of integrated relief supply and network restoration in humanitarian logistics operations. *Int J Prod Res* 54(1):49–68. <https://doi.org/10.1080/00207543.2014.977458>
- Sharkey TC, Cavdaroglu B, Nguyen H, Holman J, Mitchell JE, Wallace WA (2015) Interdependent network restoration: on the value of information-sharing. *Eur J Oper Res* 244(1):309–321. <https://doi.org/10.1016/j.ejor.2014.12.051>
- Shin Y, Kim S, Moon I (2019) Integrated optimal scheduling of repair crew and relief vehicle after disaster. *Comput Oper Res* 105:237–247. <https://doi.org/10.1016/j.cor.2019.01.015>
- Smith AM, González AD, Dueñas-Osorio L, D'Souza RM (2017) Interdependent network recovery games. *Risk Anal*. <https://doi.org/10.1111/risa.12923>
- Yan S, Shih YL (2009) Optimal scheduling of emergency roadway repair and subsequent relief distribution. *Comput Oper Res* 36(6):2049–2065. <https://doi.org/10.1016/j.cor.2008.07.002>
- Zheng YJ, Chen SY, Ling HF (2015) Evolutionary optimization for disaster relief operations: a survey. *Appl Soft Comput* 27:553–566. <https://doi.org/10.1016/j.asoc.2014.09.041>