A path-generation matheuristic for large scale evacuation planning

Victor Pillac[1], Pascal Van Hentenryck[1,2], and Caroline Even[1]

[1] NICTA Optimisation Research Group*
victor.pillac@nicta.com.au http://org.nicta.com.au
[2] Australian National University

Abstract. In this study we present a general matheuristic that decomposes the problem being solved in a master and a subproblem. In contrast with the column generation technique, the proposed approach does not rely on the explicit pricing of new columns but instead exploits features of the incumbent solution to generate one or more columns in the master problem. We apply this approach to large scale evacuation planning, leading to the first scalable algorithm that comply with emergency services practice.

1 Introduction

Natural and man-made disasters, such as hurricanes, floods, bushfires, or industrial accidents, often affect large populated areas, threatening the lives and welfare of entire populations. In such events, a common contingency is to evacuate the persons at risk to different shelters and safe areas.

Existing work in evacuation planning typically relies on free-flow models in which evacuees are dynamically routed in the network. In contrast, this paper presents an evacuation algorithm that follows recommended evacuation methodologies, which divide the evacuated area in evacuation zones, each being instructed to leave at a specific time and following a pre-defined route [24]. More specifically, it generates evacuation routes for each evacuation zone and uses a lexicographic objective function that first maximizes the number of evacuees reaching safety and then minimizes the total evacuation time. The algorithm can be used for strategic and tactical planning.

From a technical standpoint, the algorithm can be broadly characterized as a Conflict-Based Path-Generation Heuristic (CPG for short), which shares some characteristics with column generation approaches. As in column generation, it decomposes the problem by considering separately the generation of evacuation paths (subproblem) and their selection (master problem). However, a challenge of our application is the spatio-temporal nature of the problem: one evacuation path corresponds to multiple paths in the spatio-temporal graph modeling the actual

* NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

scheduling of the evacuation. As a consequence, the master problem contains interdependent binary selection variables and continuous flow variables for each evacuation path, which means that pricing a single column is not relevant, and we therefore focus on generating evacuation paths. We propose two approaches for the sub-problem. The first explicitly attempts to find a path that will improve the value of the master problem objective function, while the second aims at finding a path of least cost under constraints, where the edge costs are derived from the conflicts and congestion in the incumbent solution.

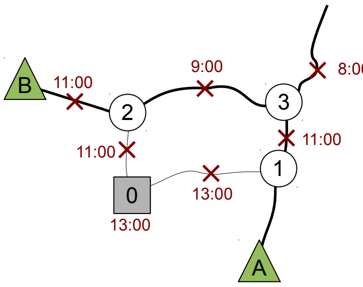
We evaluate the CPG algorithm on real-scale, massive flood scenarios in the Hawkesbury-Nepean river (West Sydney, Australia) which require evacuating in the order of 70,000 persons. Experimental results indicate that the CPG algorithm generates high-quality solutions in limited time. On small instances, where optimal solutions can be found, the CPG algorithm finds optimal or near-optimal solutions. On real-scale instances, the results show that the CPG algorithm is capable of evacuating the entire Hawkesbury-Nepean region in under 10h even if the population grows by 20%.

The remainder of this paper is organized as follows: Section 2 formulates the Evacuation Planning Problem (EPP), Section 3 reviews related work, Section 4 presents the solution approaches, Section 5 compares the performance of the proposed approaches on a set of realistic instances, and, finally, Section 6 concludes this paper.

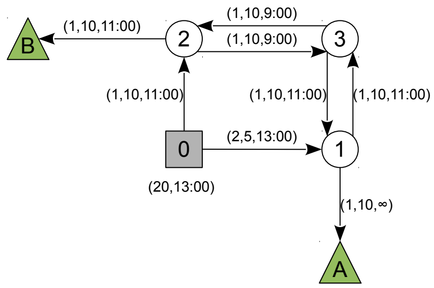
2 Problem Formulation

Figure 1 illustrates an instance of the Evacuation Planning Problem (EPP). Fig. 1(a) presents an evacuation scenario with one evacuated node (0) and two safe nodes (A and B). In this example, the evacuated node 0 has to be evacuated by 13:00, considering that certain links become unavailable at different times (for instance, $(2, 3)$ is cut at 9:00). This evacuation scenario can be represented as a graph $\mathcal{G} = (\mathcal{N} = \mathcal{E} \cup \mathcal{T} \cup \mathcal{S}, \mathcal{A})$ where $\mathcal{E}$, $\mathcal{T}$, and $\mathcal{S}$ are the set of evacuated, transit, and safe nodes respectively, and $\mathcal{A}$ is the set of edges. Each evacuated node i is characterized by a number of evacuees d_i and an evacuation deadline $\bar{f}_i$ (e.g., 20 and 13:00 for node 0 respectively), while each edge e is associated with a triple $(s_e, u_e, \bar{f}_e)$, where s_e is the travel time, u_e is the capacity, and $\bar{f}_e$ is the time at which the edge becomes unavailable.

A common way to deal with the space-time aspects of evacuation problems is to discretize the planning horizon into time steps of identical length, and to work on a *time-expanded graph*, as illustrated in Fig. 2. This graph $\mathcal{G}^d = (\mathcal{N}^d = \mathcal{E}^d \cup \mathcal{T}^d \cup \mathcal{S}^d, \mathcal{A}^d)$ is constructed by duplicating each node from $\mathcal{N}$ for each time step. For each edge $(i, j) \in \mathcal{A}$ and for each time step t in which edge (i, j) is available, an edge$(i_t, j_{t+s_{(i,j)}})$ is created modeling the transfer of evacuees from node i at time t to node j at time $t + s_{(i,j)}$. In addition, edges with infinite capacity are added to model evacuees waiting at evacuated and safe nodes. Finally, all evacuated nodes (resp. safe nodes) are connected to a virtual super-source v_s (super-sink v_t), modeling the inflow (outflow) of evacuees. Note



(a) Evacuation Scenario



(b) Evacuation Graph

Fig. 1. Modeling of an Evacuation Planning Problem.

that some nodes may not be connected to either the super-source or super-sink (in light gray in this example), and can therefore be removed to reduce the graph size. The problem is then to find a flow from v_s to v_t that models the movements of evacuees in space and time.

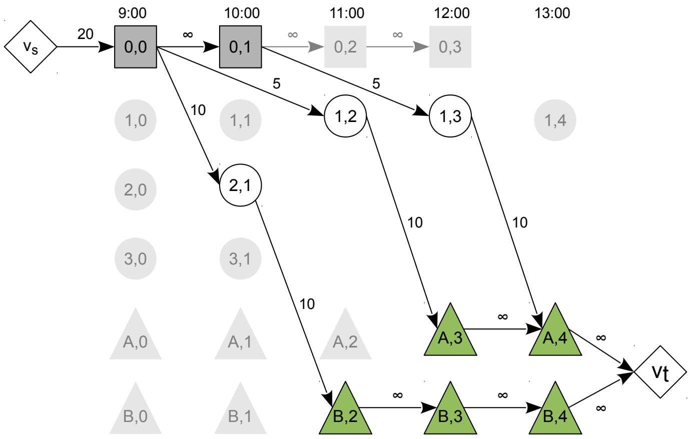


Fig. 2. Time-Expanded Graph for the Evacuation Scenario With 1-hour Time Steps.

In this study, we will make the following assumptions:

1. A decision-maker instructs each evacuee when to leave, which safe node to go to, and which path to follow in the evacuation graph;
2. A single threat scenario is known at decision time;
3. The decision-maker objective is to ensure that all evacuees reach a safe node as early as possible;
4. Each evacuated node should be assigned to a single evacuation path;

5. Edge capacities do not depend on the flow, and no congestion occurs at intersections.

Assumption 1 relates to the fact that our approach is targeted toward emergency or safety services that need to design evacuation plans for buildings or regional areas. Assumption 2 is linked to the deterministic nature of our models and algorithms. Assumption 3 defines the objective function we will use, but it must be noted that the approaches presented here can be adapted to different contexts. Requirement 4 is a practical consideration and reflects the practice in the field of emergency services operations. Finally, requirement 5 is a necessary simplification to solve the models efficiently, and it is compensated by the fact that edge capacities are set to ensure non-congested flow conditions. In that context, the problem is to design an evacuation plan that assigns a single evacuation path to each evacuated node, and to schedule the evacuation over the planning horizon, with the objective of first maximizing the number of evacuees reaching a safe node, and then minimizing the time at which the last evacuee reaches safety.

Model (1-9) presents a Mixed Integer Program modeling the Evacuation Planning Problem (EPP-MIP). Let $x^k_{e_0}$ be a binary variable equal to 1 if and only if edge $e_0 \in \mathcal{A}$ belongs to the evacuation path for evacuated node k, and φ^k_e a continuous variable equal to the flow of evacuees from evacuated node k on edge $e \in \mathcal{A}^d$. Constraints (2) ensure that exactly one path is used to route the flow coming from a same evacuated node in the evacuation graph, while constraints (3) ensure the continuity of the path. Constraints (4) ensure the flow conservation through the time-expanded graph. Constraints (5) enforce the capacity of each edge in the time-expanded graph. Constraints (6) ensure that there is no flow of evacuees coming from an evacuated node k if edge e is not part of the evacuation path for k, and Constraints (7) ensure that all evacuees leave the virtual source.

$$
\begin{aligned}
\min \quad & \sum_{e \in \mathcal{A}^d} c_e \varphi_e && && (1)\\
\text{s.t.} \quad & \sum_{e_0 \in \delta_0^+(k)} x^k_{e_0} = 1 && \forall k \in \mathcal{E} && (2)\\
& \sum_{e_0 \in \delta_0^-(i)} x^k_{e_0} - \sum_{e_0 \in \delta_0^+(i)} x^k_{e_0} = 0 && \forall k \in \mathcal{E}, i \in \mathcal{T} && (3)\\
& \sum_{e \in \delta^-(i)} \varphi^k_e - \sum_{e \in \delta^+(i)} \varphi^k_e = 0 && \forall i \in \mathcal{N}^d \setminus \{v_s, v_t\}, k \in \mathcal{E} && (4)\\
& \sum_{k \in \mathcal{E}} \varphi^k_e \le u_e && \forall e \in \mathcal{A}^d && (5)\\
& \varphi^k_e \le u_e * x^k_{e_0} && \forall e \in \mathcal{A}^d, k \in \mathcal{E} && (6)\\
& \varphi_{(v_s,k)} = d_k && \forall k \in \mathcal{E} && (7)\\
& \varphi^k_e \ge 0 && \forall e \in \mathcal{A}^d, k \in \mathcal{E} && (8)
\end{aligned}
$$

$$x_e^k \in \{0,1\} \qquad \forall e \in \mathcal{A}^d, k \in \mathcal{E} \tag{9}$$

The objective (1) is to maximize the number of evacuees reaching a safe node and minimize the weighted evacuation time. For computational efficiency, we associate a penalty with edges arriving to a safe node proportional to the time slice in which they belong. Let $t(i)$ be the time slice of time-node i, and c_{ne} a high penalty for non-evacuated evacuees. The cost $c_{(i,j)}$ of edge $(i,j) \in \mathcal{A}^d$ is defined as:

$$c_{(i,j)} = \begin{cases} c_{ne} & \text{if } i \in \mathcal{E}^d, j = v_t \\ \frac{t(i)}{H} & \text{if } i \in \mathcal{T}^d, j \in \mathcal{S}^d \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

3 Related work

According to Hamacher and Tjandra [10], evacuation planning can be tackled using either *microscopic* or *macroscopic* approaches. Microscopic approaches focus on modeling and simulating the evacuees individual behaviors, movements, and interactions. Macroscopic approaches, such as the one presented in this study, aggregate evacuees and model their movements as a flow in the evacuation graph.

To the best of our knowledge, only a handful of studies attempt to design evacuation plans as we defined them [22]. Huibregtse et al. [14] propose a two stage algorithm that first generates a set of evacuation routes and feasible departure time, and then assigns a route and time to each evacuated area using an ant colony optimization algorithm. A key feature of the approach is the use of traffic simulation to evaluate the quality of solutions. In later work, the authors studied the robustness of the produced solution [13], and strategies to improve the compliance of evacuees [12].

A significant number of contributions attempt to solve flow problems directly derived from the time-expanded graph. For instance, Lu et al. [18, 19] propose three heuristics to design an evacuation plan with multiple evacuation routes per evacuated node, minimizing the time of the last evacuation. The authors show that in the best case the proposed heuristic is able to solve randomly generated instances of up to 50,000 nodes and 150,000 edges in under 6 minutes. Liu et al. [17] propose a Heuristic Algorithm for Staged Traffic Evacuation (HASTE), a similar algorithm that generates augmenting chains in the time-expanded graph. The main difference between HASTE and the previous algorithms is that it relies on a Cell Transmission Model (CTM)[8] to model more accurately the flow of evacuees.

Acknowledging that all evacuated nodes may not be under the same level of threat, Lim et al. [15] consider a short-notice regional evacuation maximizing the number of evacuees reaching safety weighted by the severity of the threat. The authors propose two solution approaches to solve the problem, and present computational experiments on instances derived from the Houston-Galveston region (USA) with up to 66 nodes, 187 edges, and an horizon of 192 time steps.

Other authors have focused on modeling more accurately the transportation network. For example, Bretschneider and Kimms [5, 6] present a free-flow mathematical model that describes in detail the street network and, in particular, the lane configuration at intersections of the network. They present computational experiments on generated instances with a grid topology of up to 240 nodes, 330 edges, and considering 150 times steps. Bish and Sherali [4] present a model based on a CTM that assigns a single evacuation path to each evacuated node. Computational results include instances with up to 13 evacuated nodes, 2 safe nodes, and 72 edges.

Finally, dynamic aspects of evacuation have also been considered. For instance, Lin et al. [16] present a time expanded graph in which they allow for time-dependent attributes such as varying capacity or demand. The authors apply their findings on a case study considering the evacuation of a 11-floor building with approximately 60 nodes, 100 edges, and 60 time steps.

Microscopic approaches include the work by Richter et al. [23] who challenge two assumptions generally made: The existence of a central planning entity with global knowledge, and the ability of this entity to communicate order to evacuees. They propose a decentralized decision making approach supported by smartphones and mobile applications. We note however that our target applications, such as evacuations for floods and hurricanes, use central decision making and have the time and ability to communicate their decisions.

In contrast with the cited studies, the approach proposed in this work is the first to produce evacuation plans that are actionable from an emergency service perspective. It generates a plan that assigns one evacuation route to each evacuated area, and optimizes both the evacuation routes and schedule globally.

Column generation is an optimization technique which consists in considering only a subset of columns in a master problem and then iteratively generating columns of negative reduced cost (assuming minimization) by solving a pricing subproblem. It has been widely used to solve large-scale MIP problems, and we refer the interested reader to the book by Desaulniers et al. [9] and the study by Lübbecke and Desrosiers [20] for a recent review of techniques and applications of column generation. In particular, it has been used to solve multi-commodity network flow problems (MCNF) [1], integer MCNF [3], origin-destination MCNF [2], and MCNF with side constraints on paths [11].

However, a distinctive feature of evacuation planning is the dependency between paths in the time-expanded network. More precisely, a commodity (i.e., evacuees from a specific evacuated node) can only follow paths that correspond to the same physical path (sequence of edges in the evacuation graph). Therefore classical MCNF approaches cannot be applied directly, as one path in the evacuation model introduces multiple variables in the master problem. In addition, it is worth noting that heuristic column generation have mainly focused on solving the pricing subproblem heuristically. In contrast, our approach does not consider the pricing problem explicitly, but heuristically generates new paths. Similar ideas were also used by Coffrin et al. [7] and, to a lesser extent, in Massen et al. [21].

4 Proposed approaches

Computational experiments show that the EPP-MIP model (1-9) becomes intractable for instances with more than eight nodes and an horizon of 10h divided in 5 minutes steps. Therefore, we propose a conflict-based heuristic path generation approach (CPG) that separates the generation of evacuation paths from the scheduling of the evacuation.

Figure 3 gives an overview of the CPG approach. First, the algorithm generates an initial set of paths Ω' (1) and solves a master problem to find an evacuation schedule optimizing the objective function (2). Then it identifies *critical* evacuated nodes $\mathcal{E}'$ (3), which are not fully evacuated, or evacuated late, and considers nodes that are potentially in conflict (5) with the objective of generating new paths (6). Finally, it solves the scheduling problem including the newly generated paths (2). The steps are repeated for a fixed number of iterations.

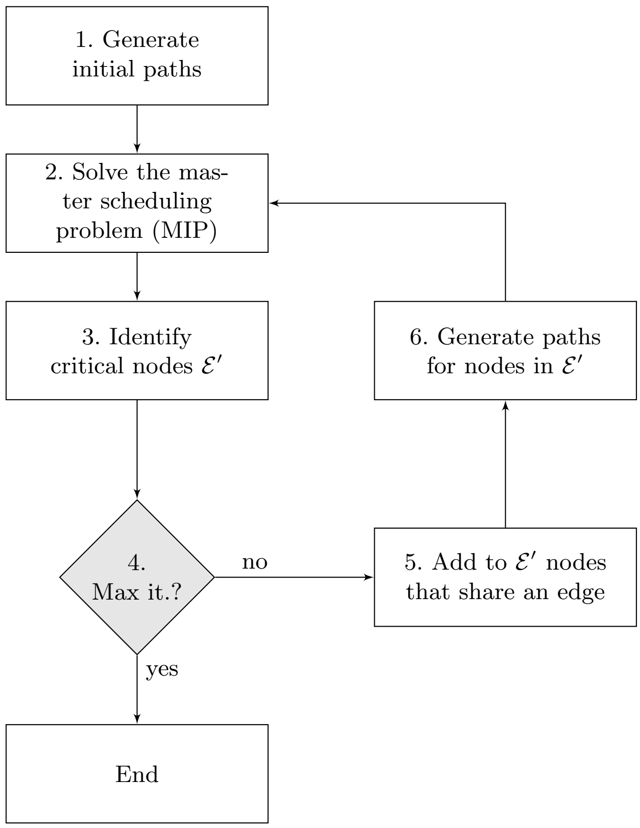


Fig. 3. Overview of the conflict-based heuristic path generation approach

4.1 Master problem

Let Ω be the set of all feasible paths between evacuated nodes and safe nodes and Ω_k be the subset of evacuation paths for evacuated node k. We define a binary variable x_p which takes the value of 1 if and only if path $p \in \Omega$ is selected, a continuous variable φ_p^t representing the number of evacuees to start evacuating on path p at time t, and a continuous variable φ_k accounting for the number of non-evacuated evacuees in node k. In addition, we denote by $\omega(e)$ the subset of paths that contain edge e and by τ_p^e the number of time steps required to reach edge e when following path p. Finally, we denote by $\mathcal{H}_p \subseteq \mathcal{H}$ the subset of time steps in which path p is usable, and u_p the capacity of path p.

$$\min \quad \sum_{k\in\mathcal{E}} \varphi_k c_{ne} + \sum_{p\in\Omega}\sum_{t\in\mathcal{H}_p} \varphi_p^t c_p^t \tag{11}$$

$$\text{s.t.} \quad \sum_{p\in\Omega_k} x_p = 1 \qquad \forall k \in \mathcal{E} \tag{12}$$

$$\sum_{p\in\Omega_k}\sum_{t\in\mathcal{H}_p} \varphi_p^t + \varphi_k = d_k \qquad \forall k \in \mathcal{E} \tag{13}$$

$$\sum_{\substack{p\in\omega(e)\\ t-\tau_p^e\in\mathcal{H}_p}} \varphi_p^{t-\tau_p^e} \le u_e \qquad \forall e \in \mathcal{A}, t \in \mathcal{H} \tag{14}$$

$$\sum_{t\in\mathcal{H}_p} \varphi_p^t \le |\mathcal{H}_p| x_p u_p \qquad \forall p \in \Omega \tag{15}$$

$$\varphi_p^t \ge 0 \qquad \forall p \in \Omega, t \in \mathcal{H}_p \tag{16}$$

$$\varphi_k \ge 0 \qquad \forall k \in \mathcal{E} \tag{17}$$

$$x_p \in \{0,1\} \qquad \forall p \in \Omega \tag{18}$$

Model (11-18) presents the evacuation scheduling problem CPG-MP. The objective (11) minimizes the cost of the solution as defined previously. Constraints (12) ensure that exactly one path is selected for each evacuated node, while constraints (13) account for the number of evacuated and non-evacuated evacuees. Constraints (14) enforce the capacity on the edges of the graph. Finally, constraints (15) ensures that there is no flow on paths that are not selected. It is interesting to observe that the master problem does not use a variable for each edge e and time step t. Instead, it reasons in terms of variables φ_p^t which indicate how many evacuees leave along path p at time t.

In practice, we only consider a subset of evacuation paths $\Omega' \subseteq \Omega$ each time we solve CPG-MP. Fig. 4 depicts the structure of the master problem matrix. Horizontal blocks represent groups of constraints numbered as in model (11-18), while the shaded areas represent non-null coefficients in the matrix. Note that each constraint in group (15) only involves variables associated with the corresponding path and must be dynamically added to the model whenever a new path is considered. Nonetheless, a solution of CPG-MP considering the

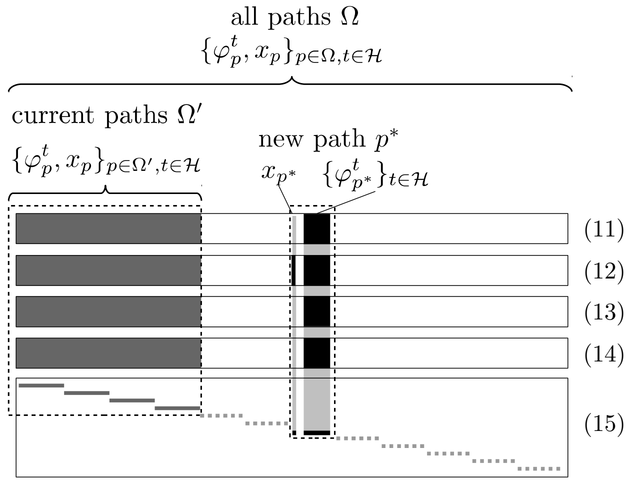


Fig. 4. The Structure of the Evacuation Scheduling Master Problem Matrix.

subset of paths $\Omega'' \subset \Omega'$ is feasible when considering the set Ω'. Hence the solution from the previous iteration is used as starting solution for the current iteration.

4.2 Subproblem

Considering the spatio-temporal nature of this application, and the fact that a path corresponds to multiple columns and introduces a new constraint, we do not rely on traditional column generation techniques to generate new paths. Instead, we use problem-specific knowledge to generate new columns that will potentially improve the objective function of the master problem. First, we identify the subset $\mathcal{E}' \subseteq \mathcal{E}$ of critical evacuated nodes, i.e., nodes that are not fully evacuated in the current solution or evacuated late. Then, we include in $\mathcal{E}'$ all the evacuated nodes whose evacuation paths share at least one edge with a node from $\mathcal{E}'$. Finally, we generate new paths for the critical evacuated nodes $\mathcal{E}'$.

Improving Path Generation (IPG) The first path generation approach we propose borrows ideas from the Large Neighborhood Search algorithm [25]. Conceptually, we transform a solution of CPG-MP into a solution of EPP-MIP, fixing all flow variables in EPP-MIP to their value derived from CPG-MP, except for the subset of variables corresponding to the evacuation area $k^* \in \mathcal{E}'$ for which a new path is to be generated. In other words, we solve EPP-MIP for a single evacuated area (k^*) reducing the capacity on the edges of the time-expanded graph. Model (19-26) presents a MIP formulation of the resulting problem. The model is very similar to the original problem, at the difference that only one evacuation area is considered. In addition, Constraints (23) limit the flow on every edge to the residual capacity u'_e which is equal to the original capacity minus the value of the flow on the incumbent solution (excluding the flow corresponding to k^*).

$$\min \quad \sum_{e \in \mathcal{A}^d} c_e \varphi_e \tag{19}$$

$$\text{s.t.} \quad \sum_{e_0 \in \delta_0^+(k^*)} x_{e_0} = 1 \tag{20}$$

$$\sum_{e_0 \in \delta_0^-(i)} x_{e_0} - \sum_{e_0 \in \delta_0^+(i)} x_{e_0} = 0 \qquad \forall k \in \mathcal{E}, i \in \mathcal{T} \tag{21}$$

$$\sum_{e \in \delta^-(i)} \varphi_e - \sum_{e \in \delta^+(i)} \varphi_e = 0 \qquad \forall i \in \mathcal{N}^d \setminus \{v_s, v_t\} \tag{22}$$

$$\varphi_e \leq u'_e * x_{e_0} \qquad \forall e \in \mathcal{A}^d \tag{23}$$

$$\varphi_{(v_s,k^*)} = d_{k^*} \tag{24}$$

$$\varphi_e \geq 0 \qquad \forall e \in \mathcal{A}^d \tag{25}$$

$$x_e \in \{0,1\} \qquad \forall e \in \mathcal{A}^d \tag{26}$$

The values of the $(x_{e_0})_{e0 \in \mathcal{E}}$ variables in each solution of the IPG problem define a new path for the CPG-MP. If the IPG problem does not admit a solution, or if the value of the solution is greater than the contribution of k^* to the incumbent solution value, we select up to $n = 5$ evacuation areas which share some edges with k^* and remove their contribution to residual capacity. This is equivalent to assuming that the removed areas do not need to be evacuated, and it relaxes the IPG by increasing the capacity on the edges. This process is repeated for all evacuated nodes identified as critical.

Heuristic Path Generation (HPG) The second path generation approach attempts to generate diverse evacuation paths by solving a series of independent shortest path problems from each evacuated node to all safe nodes. The cost c_e of edge e is adjusted at each iteration using the following linear combination of the edge's travel time s_e, the number of occurrences of e in the current set of paths, and the utilization of e in the current solution:

$$c_e = \alpha_t \frac{s_e}{\max_{e \in \mathcal{E}} s_e} + \alpha_c \frac{\sum_{\substack{p \in \Omega' \\ e \in p}} 1}{|\Omega'|} + \alpha_u \frac{\sum_{\substack{p \in \Omega' \\ e \in p}} \sum_{t \in \mathcal{H}_p} \varphi_p^t}{u_e} \tag{27}$$

where α_t, α_c, and α_u are positive weights which sum is equal to 1.

5 Computational experiments

We consider the evacuation of the Hawkesbury-Nepean (HN) floodplain, located North-West of Sydney, for which a 1-in-200 years flood will require the evacuation of 70,000 persons. The resulting evacuation graph contains 50 evacuated nodes, 6 safe nodes, 153 transit nodes, and 485 edges. We consider a horizon of 10 hours with a time step of 5 minutes (starting at 00h00). In addition, we generate the instances HN-Rx with a subset of x $\in [2, 50]$ evacuation nodes and a reduced graph, and HN-Ix which have the same evacuation graph but a number of evacuees scaled by a factor of x $\in [1.1, 3.0]$.

All approaches were implemented using Java 7 and Gurobi 5.5, and experiments were conducted on a cluster of 64bits machines with 2.8GHz AMD 6-Core Opteron 4184 and 16Gb of RAM. Results are an average over 10 runs given the randomized nature of parts of the algorithms and of Gurobi internal heuristics. We set a limit of 10 iterations for CPG, which generally converges quickly. The IPG subproblem is solved using the Gurobi solver, while HPG relies on the Dijsktra algorithm to evaluate the shortest paths.

Table 1 compares the percentage of evacuees reaching safety (Perc. Evac) and the time at which the last evacuees reaches safety (Evac. End) in the solutions produced by CPG and by solving the EPP-MIP with the Gurobi solver. The figures in bold indicate proved optimum solution, figures in italics represent incumbent solution at the 30 min time limit. The results indicate that for instances of up to 5 nodes, the three approaches find the optimal solution. However, for larger instances, EPP-MIP does not terminate in the time limit.

	CPG					
	IPG		**HPG**		**EPP-MIP**	
	Perc. Evac	**Evac. End**	**Perc. Evac**	**Evac. End**	**Perc. Evac**	**Evac. End**
HN-R02	100%	02h30	100%	02h30	**100%**	**02h30**
HN-R03	100%	01h55	100%	01h55	**100%**	**01h55**
HN-R05	100%	02h25	100%	02h25	**100%**	**02h25**
HN-R08	100%	02h50	100%	02h50	*100%*	*02h50*
HN-R10	100%	02h50	100%	02h50	*100%*	*04h25*
HN-R20	100%	02h25	100%	02h50	*78%*	*10h00*
HN-R30	100%	02h45	100%	03h05	*82%*	*10h00*
HN-R40	100%	09h15	100%	09h15	76%	10h00
HN-R50	100%	09h15	100%	09h15	-	-

Table 1. Comparison of solution quality on reduced size instances.

Table 2 presents the percentage of evacuees reaching safety (Perc. Evac), the time at which the last evacuees reaches safety (Evac. End) and the number of paths generated (Num. Paths) for both the IPG and HPG path generation. For comparison, we include results obtained when solving the master problem with only the three shortest paths from each evacuated area to the closest safe nodes. The results show that for instances with up to 20% additional evacuees, the whole area can be evacuated in under 10h by both approaches. It is worth noting that on these instances IPG and HPG give the same solutions, but IPG generates more than 10 times less paths. When the population increases, HPG performs increasingly better than IPG, evacuating more people in the 10h limit. Both approaches dominate the 3 shortest paths baseline.

Instance	HPG			IPG			3 Shortest Paths		
	Perc. Evac.	Evac. Time	Num. Paths	Perc. Evac.	Evac. Time	Num. Paths	Perc. Evac.	Evac. Time	Num. Paths
HN-I	100%	08h05	1057	100%	08h05	65	100%	09h55	150
HN-I1.1	100%	08h45	1058	100%	09h10	80	100%	09h55	150
HN-I1.2	100%	09h25	1117	100%	09h25	79	98%	10h00	150
HN-I1.4	99%	10h00	741	98%	10h00	69	95%	10h00	150
HN-I1.7	97%	10h00	1211	93%	10h00	80	84%	10h00	150
HN-I2.0	94%	10h00	1118	91%	10h00	85	75%	10h00	150
HN-I2.5	84%	10h00	1193	74%	10h00	87	65%	10h00	150
HN-I3.0	75%	10h00	995	61%	10h00	85	58%	10h00	150
Average	**94%**	**09h31**	**1061**	**90%**	**09h35**	**79**	**84%**	**09h58**	**150**

Table 2. Comparison of solution quality.

Table 3 provides further insights on the relative performance of both approaches. It presents the CPU time (in seconds) spent in the master problem (MP) and path generation subproblem (SP) for the different instances. It appears that in the IPG approach solving the master problem takes less than 10 second in total, while it requires 67 minutes in the HPG. This difference can be explained by the smaller number of paths generated by IPG, which greatly reduces the size of the MP. On the other hand, IPG spends 285 minutes on the generation of new paths, compared with 2 seconds for HPG. This is explained by the algorithm used to solve the subproblem: HPG relies on the Dijsktra algorithm which is polynomial in the size of the time expanded graph, while IPG uses a MIP solver.

Instance	HPG		IPG	
	MP	SP	MP	SP
HN	2,617	1	5	2,533
HN-I1.1	2,684	1	9	15,094
HN-I1.2	2,652	1	9	8,622
HN-I1.4	1,459	1	3	11,868
HN-I1.7	3,623	2	5	17,367
HN-I2.0	5,156	2	9	17,121
HN-I2.5	7,000	2	9	25,357
HN-I3.0	6,906	2	7	39,195
Average	**4,012**	**2**	**7**	**17,145**

Table 3. Comparison of average CPU times (in seconds) for the master problem (MP) and subproblem (SP)

6 Conclusions

In this paper, we presented a generic matheuristic which borrows ideas from column generation and large neighborhood search. This general framework decomposes a complex problem in a master and subproblem. Similarly to column generation approaches, the master problem provides information to the subproblem to generate new variables. The first major difference is that the subproblem does not seek to generate a single column, but a set of columns linked by a constraint. The second major difference is that the subproblem does not explicitly minimize the reduced cost of the new columns, which means that it does not require access to the dual variables of the master problem. This is particularly an advantage when the master problem is a MIP. Instead, it attempts to identify features of the current incumbent that can be improved to lead to a better solution.

We applied the proposed matheuristic to the planning of large scale evacuations and demonstrated that it was able to produce high quality evacuation plans in reasonable time. In that application, the master problem selects evacuation paths and schedule the flow of evacuees, while the subproblem generates new paths for nodes identified as critical. We compared two approaches to solve the subproblem. The first, namely IPG, transforms the incumbent solution of the master problem in a solution of the original problem, and relaxes the variables corresponding to the evacuation node for which a new evacuation is to be generated. The second, HPG, relies on a randomized heuristic that generates new paths solving a shortest path problem on the evacuation graph where edges are penalized depending on their usage in the incumbent solution.

Preliminary computational results indicate that HPG has an advantage in terms of solution quality and CPU time. However, it appears that IPG produces competitive results with significantly less paths, which translates into lower CPU times for the master problem.

Future work will focus on the development of ad-hoc algorithms to solve the IPG problem more efficiently to speed up the algorithm. In addition, we are investigating ways to control the set of paths included in the master problem to improve the solution quality without affecting the computational time.

Bibliography

[1] Alvelos, F. and Valrio De Carvalho, J. (2000). Solving multicommodity flow problems with branch-and-price. *Technical Report*.

[2] Barnhart, C., Hane, C., and Vance, P. (1997). Integer multicommodity flow problems. *Lecture Notes in Economics and Mathematical Systems*, 450:17–31.

[3] Barnhart, C., Hane, C. A., and Vance, P. H. (2000). Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326.

[4] Bish, D. R. and Sherali, H. D. (2013). Aggregate-level demand management in evacuation planning. *European Journal of Operational Research*, 224(1):79–92.

[5] Bretschneider, S. and Kimms, A. (2011). A basic mathematical model for evacuation problems in urban areas. *Transportation Research Part A: Policy and Practice*, 45(6):523–539.

[6] Bretschneider, S. and Kimms, A. (2012). Pattern-based evacuation planning for urban areas. *European Journal of Operational Research*, 216(1):57–69.

[7] Coffrin, C., Van Hentenryck, P., and Bent, R. (2011). Strategic stockpiling of power system supplies for disaster recovery. In *2011 IEEE Power and Energy Society General Meeting*, pages 1–8. IEEE.

[8] Daganzo, C. F. (1994). The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287.

[9] Desaulniers, G., Desrosiers, J., and Solomon, M. M., editors (2005). *Column Generation*. Mathematics of Decision Making. Springer.

[10] Hamacher, H. W. and Tjandra, S. A. (2001). Mathematical modelling of evacuation problems: A state of art. Technical report, Fraunhofer Institut für Techno und Wirtschaftsmathematik.

[11] Holmberg, K. and Yuan, D. (2003). A multicommodity network-flow problem with side constraints on paths solved by column generation. *INFORMS Journal on Computing*, 15(1):42–57.

[12] Huibregtse, O., Hegyi, A., and Hoogendoorn, S. (2012). Blocking roads to increase the evacuation efficiency. *Journal of Advanced Transportation*, 46(3):282–289.

[13] Huibregtse, O. L., Bliemer, M. C., and Hoogendoorn, S. P. (2010). Analysis of near-optimal evacuation instructions. *Procedia Engineering*, 3(0):189–203. ¡ce:title¿1st Conference on Evacuation Modeling and Management¡/ce:title¿.

[14] Huibregtse, O. L., Hoogendoorn, S. P., Hegyi, A., and Bliemer, M. C. J. (2011). A method to optimize evacuation instructions. *OR Spectrum*, 33(3):595–627.

[15] Lim, G. J., Zangeneh, S., Baharnemati, M. R., and Assavapokee, T. (2012). A capacitated network flow optimization approach for short notice evacuation planning. *European Journal of Operational Research*, 223(1):234–245.

[16] Lin, P., Lo, S., Huang, H., and Yuen, K. (2008). On the use of multi-stage time-varying quickest time approach for optimization of evacuation planning. *Fire Safety Journal*, 43(4):282–290.

[17] Liu, H. X., He, X., and Ban, X. (2007). A cell-based many-to-one dynamic system optimal model and its heuristic solution method for emergency evacuation. In *Proc. 86th Annual Meeting Transportation Res. Board*.

[18] Lu, Q., George, B., and Shekhar, S. (2005). Capacity constrained routing algorithms for evacuation planning: A summary of results. In Bauzer Medeiros, C., Egenhofer, M., and Bertino, E., editors, *Advances in Spatial and Temporal Databases*, volume 3633 of *Lecture Notes in Computer Science*, pages 291–307. Springer Berlin Heidelberg.

[19] Lu, Q., Huang, Y., and Shekhar, S. (2003). Evacuation planning: A capacity constrained routing approach. In Chen, H., Miranda, R., Zeng, D., Demchak, C., Schroeder, J., and Madhusudan, T., editors, *Intelligence and Security Informatics*, volume 2665 of *Lecture Notes in Computer Science*, pages 111–125. Springer Berlin Heidelberg.

[20] Lübbecke, M. and Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, 53(6):1007–1023.

[21] Massen, F., Deville, Y., and Hentenryck, P. V. (2012). Pheromone-based heuristic column generation for vehicle routing problems with black box feasibility. In *CPAIOR*, pages 260–274.

[22] Pillac, V., Hentenryck, P. V., and Even, C. (2013). A conflict-based path-generation heuristic for evacuation planning. Technical Report VRL-7393, NICTA. arXiv:1309.2693.

[23] Richter, K.-F., Shi, M., Gan, H.-S., and Winter, S. (2013). Decentralized evacuation management. *Transportation Research Part C: Emerging Technologies*, 31(0):1–17.

[24] SES-NSW (2005). Hawkesbury nepean flood emergency sub plan. Technical report, State Emergency Service - New South Wales.

[25] Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming – CP98*, volume 1520 of *Lecture Notes in Computer Science*, pages 417–431. Springer Berlin / Heidelberg.