Large-Scale Zone-based Evacuation Planning – Part II: Macroscopic and Microscopic Evaluations

Mohd. Hafiz Hasan* Pascal Van Hentenryck†

Abstract

A companion paper [8] introduces models and algorithms for large-scale zone-based evacuation planning in which each evacuation zone is assigned a path to safety and a departure time. It also shows how to combine zone-based evacuations with contraflows and impose additional path-convergence and non-preemptive constraints.

This paper evaluates these algorithms on a real, large-scale case study, both from a macroscopic standpoint and through microscopic simulations under a variety of assumptions. The results quantify, for the first time, the benefits and limitations of contraflows, convergent plans, and non-preemption, providing unique perspectives on how to deploy these algorithms in practice. They also highlight the approaches best suited to capture each of these design features and the computational burden they impose. The paper also suggests new directions for future research in zone-based evacuation planning and beyond in order to address the fundamental challenges by emergency services around the world.

Keywords: Evacuation Planning and Scheduling, Mathematical Optimization, Benders Decomposition, Column Generation, Time-Expanded Graphs, Non-Preemptive and Convergent Evacuations, Contraflow.

1 Introduction

This paper focuses on *zone-based evacuations* in which all evacuees from the same residential zone are assigned a single evacuation route. Most emergency services rely on zone-based evacuations to facilitate the communication of evacuation plans, reduce confusion, increase compliance, and allow for a more reliable control of the evacuation Indeed, zone-based evacuations are probably the only practical method for communicating instructions precisely to the population at risk. However, they are much more computationally challenging to plan and finding scalable algorithms has been one of the foci of recent research.

Part I of this paper presented the core Zone-based Evacuation Planning Problem (ZEPP) which consists in assigning an evacuation path, as well as departure times, to each zone in the region. It also discusses key extensions of the ZEPP: Convergent (C-ZEPP) and Non-Preemptive (NP-ZEPP) zone-based Evacuation Planning Problems. In convergent plans, all evacuees coming to an intersection exit through the same edge, eliminating forks from all evacuation paths and thus reducing driver hesitation at road intersections. In non-preemptive plans, the evacuation of a zone, once it starts, proceeds without interruptions; non-preemptive evacuations are also preferred by emergency services since they are easier to enforce. Contraflows or lane reversals, i.e., the idea of using inbound lanes for outbound traffic in evacuations, can also be used in combination with all these approaches to increase the capacity of the network.

Table 1 summarizes the algorithms presented in Part I and evaluated in this paper. The algorithms consider the ZEPP, C-ZEEP, and NP-ZEPP and will be evaluated with and without the use of contraflows. The algorithms also use different optimization methods, leveraging the structure of the underlying problem type. Benders decomposition as well as heuristic and exact column generation are the two methods of choice, as black-box solvers cannot scale to the size of the case study under consideration.

The purpose of this paper is to provide a systematic comparative study of these algorithms, their design choices, their fidelity, and their computational performance. The algorithms are compared on a real-life case

*University of Michigan, Ann Arbor. University of Michigan, Ann Arbor, Michigan 48109, Email: hasanm@umich.edu.
†Georgia Institute of Technology, Atlanta, Georgia 30332, Email: pvh@isye.gatech.edu.

Table 1: The Zone-based Evacuation Planning Algorithms: Problem Types and Optimization Methods.

Method	Problem Type	Optimization Method
BN	ZEPP	Benders Decomposition
BC	C-ZEPP	Benders Decomposition
CPG	ZEPP	Path Generation
CG	NP-ZEPP	Column Generation

study both at macroscopic and microscopic scales. The case study concerns the Hawkesbury-Nepean (HN) region located north-west of Sydney (Australia) and the associated time-expanded graph has 30,000 nodes and 75,000 arcs. This evaluation quantifies, for the first time, the benefits and limitations of contraflows, convergent plans, and non-preemption, providing unique perspectives on how to deploy these algorithms in practice. It also highlights the approaches that are best suited to capture each of these design features and the computational burden they impose.

The rest of this paper is organized as follows. Section 2 presents the case study, the experimental setting, and some initial observations. Section 3 presents the experimental results from a macroscopic standpoint and Section 4 provides the details of the microscopic simulations. Section 5 gives some perspectives on directions for future research and knowledge gaps that needs to be filled. Finally, Section 6 concludes the paper.

2 The Case Study

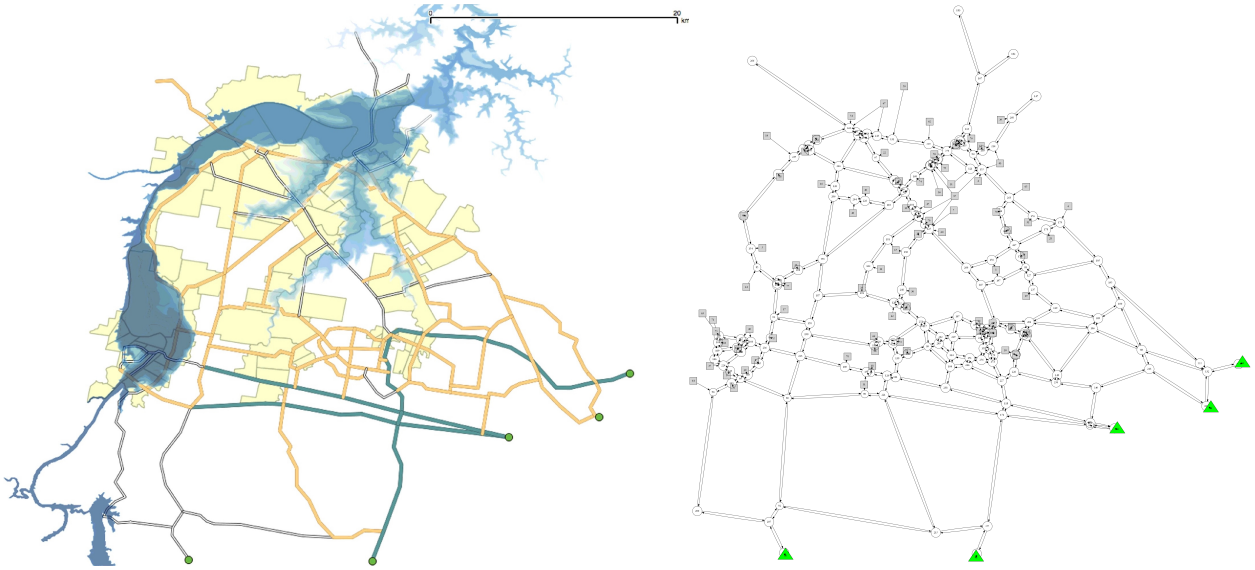
This section presents a case study to investigate the effectiveness and run times of the four methods on a real-world evacuation scenario. It also discusses some preliminary observations on various properties of the evacuation algorithms.

The Case Study The case study is the Hawkesbury-Nepean (HN) region located north-west of Sydney, Australia, which is separated from the Blue Mountains, a catchment area, by the Warragamba dam (see Figure 1). This dam often spills over and, if it breaks, it would create a massive flooding event for West Sydney. The region's evacuation graph consists of 80 evacuation nodes, 184 transit nodes, and 5 safe nodes. Evacuation node deadlines and road block times for the region were obtained from a hydro-dynamic simulation of a 1-in-100 years flood event. The region has a total of 38,343 vehicles to be evacuated in its base instance; however, the results also consider the effect of increasing the population size by linearly scaling the base instance by a factor $x \in [1.0, 3.0]$, as West Sidney is experiencing significant population growth. Figure 2a shows a bird's-eye view of the entire region, whereas Figure 2b shows its corresponding evacuation graph, with squares, circles, and triangles representing evacuation, transit, and safe nodes, respectively. Each instance (or evacuation scenario) is referred to as HN80-Ix from this point forth, where x is the population scaling factor.

Experimental Settings The performance of each method is evaluated under two settings: (a) a deadline setting where the maximum number of evacuees reaching safety is determined within a fixed time horizon $\mathcal{H} = 10$ hours, and (b) a minimum clearance time setting where the smallest amount of time needed to evacuate the entire region is determined. Under both settings, the time horizon is discretized into 5 minute time steps. For the CG method, the set $\mathcal{F}$ of predefined response curves was populated with step response curves having flow rates of $\gamma \in \{2, 6, 10, 25, 50\}$ evacuees per time step. All methods were implemented in C++, with multi-threaded components being handled using OpenMP, used in conjunction with Gurobi 6.5.2 to solve all mathematical programs. All experiments were conducted on the University of Michigan's Flux high-performance computing cluster, using 8 cores of a 2.5 GHz Intel Xeon E5-2680v3 processor and 64 GB of RAM.

Convergent versus Non-Convergent Paths Figures 3a and 3b illustrate examples of generated non-convergent and convergent evacuation paths that do not use contraflow. The paths, whose arcs are represented by red arrows, are overlaid on top of the evacuation graph in these figures. It can be seen in Figure

Figure 1: The Warragamba Dam in New South Wales.

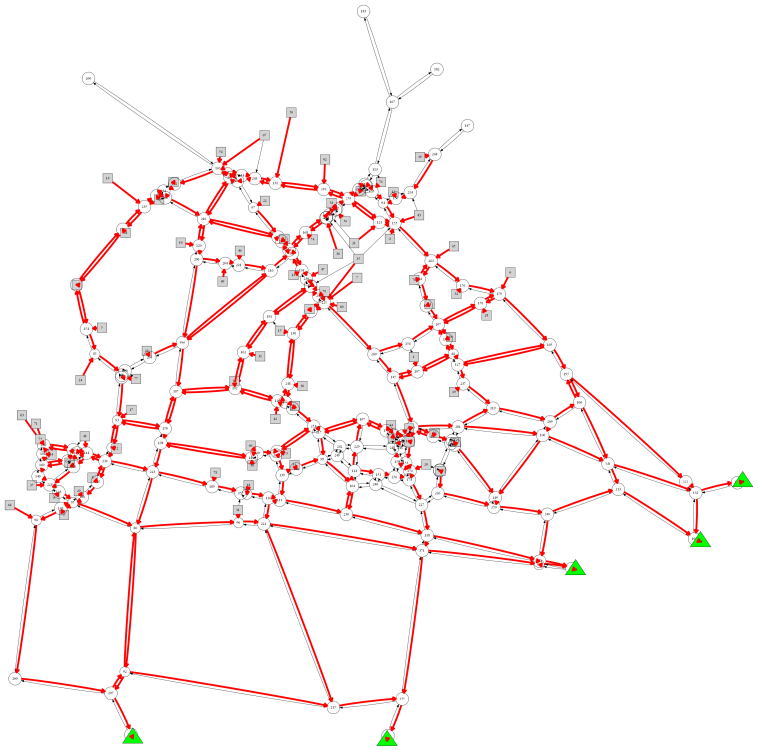


(a) HN map (from [13])

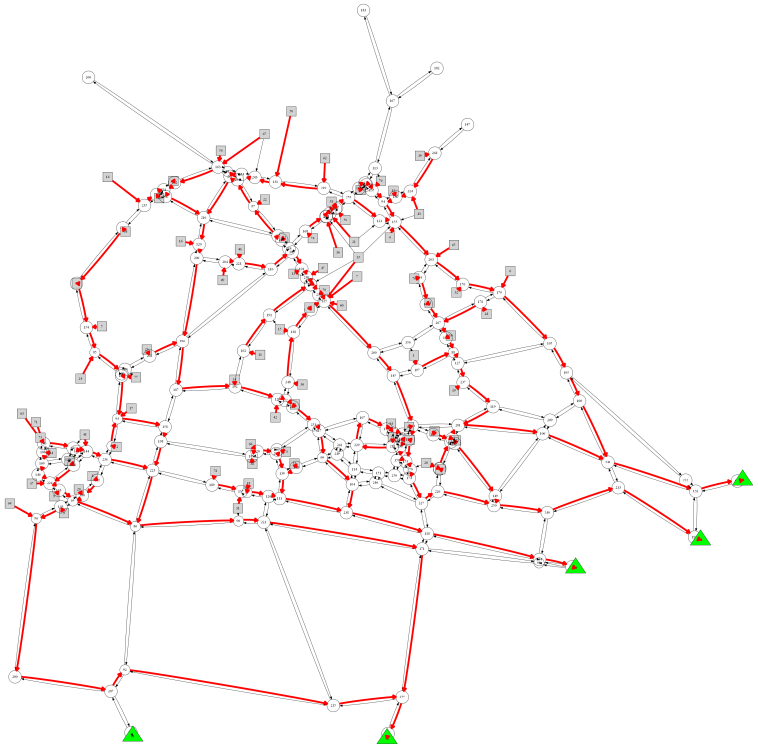
(b) HN evacuation graph

Figure 2: The Map and Evacuation Graph of the Hawkesbury-Nepean Region. The Dam is at the South-West Corner of the Left Map.

3b that convergent paths form a tree with leaves at the evacuation nodes (squares in the graph) and rooted at the safe nodes (triangles in the graph). The non-convergent paths in Figure 3a do not share this property; however, not being constrained by this property allows more arcs to be utilized for evacuation (at the expense introducing potential delays caused by driver hesitation when multiple paths fork at road intersections).



(a) Non-Convergent Evacuation Paths.



(b) Convergent Evacuation Paths.

Figure 3: Non-Convergent and Convergent Evacuation Paths without Contraflow Generated by the BN and BC Methods.

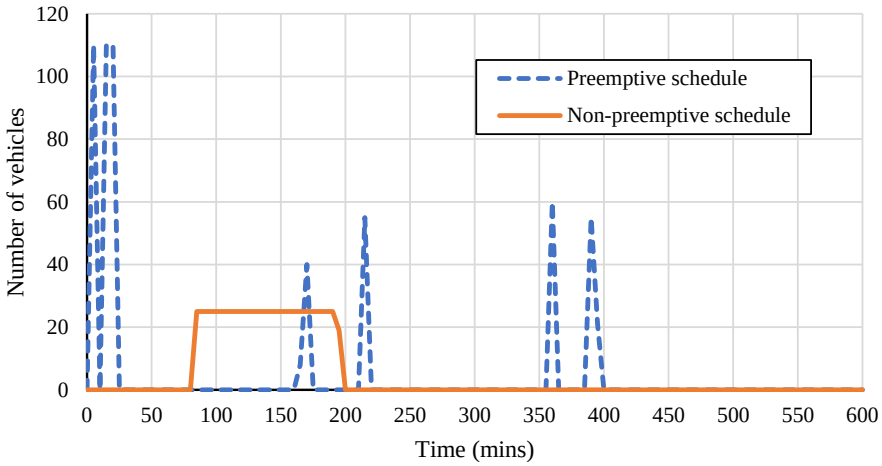
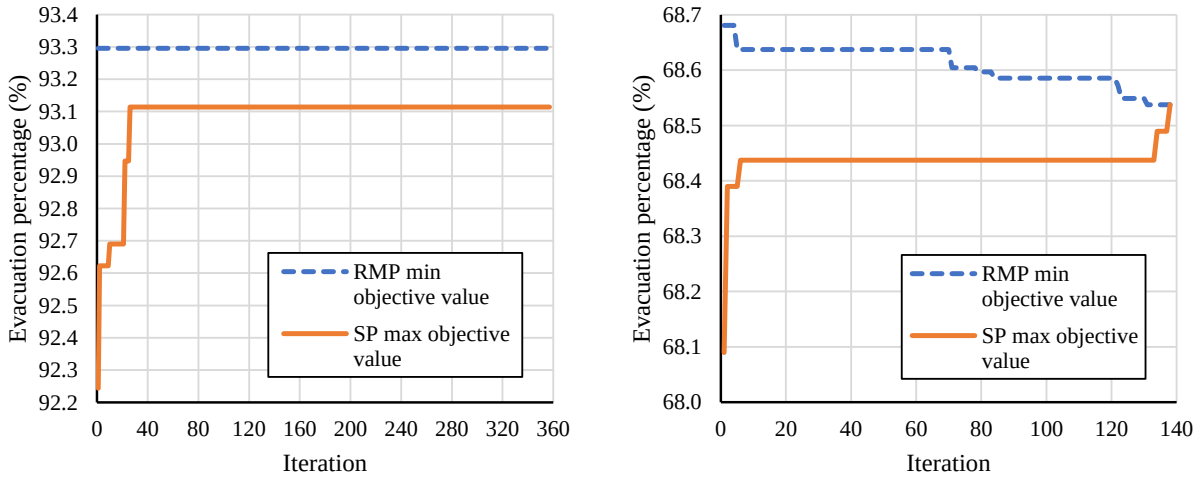


Figure 4: Preemptive and Non-Preemptive Evacuation Schedules for an Evacuation Node with 569 Evacuees (Generated by the BN and CG Methods, Respectively).

Preemptive Versus Non-Preemptive Schedules Figure 4 highlights the difference between preemptive and non-preemptive evacuation schedules generated for an evacuation node with 569 evacuees. The preemptive schedule is characterized by multiple spikes followed by interruptions in evacuee departure rates over time, which may lead to some challenges in the enforcement of the schedule. The preemptive evacuation contrasts with the non-preemptive schedule, which uses a step response curve with a flow rate of 25 evacuees every 5 minutes. Evacuation is started at the 85^{th} minute, and a constant departure rate is maintained until the node has been completely evacuated, making the schedule arguably easier to enforce compared to the preemptive one.

Convergence of the Benders Decomposition Figures 5a and 5b reveal how the upper and lower bounds of the BN and BC methods converge over time for a particular experiment in the deadline setting. For this instance, the BC method converged in less than 140 iterations, whereas the BN method did not, even after 360 iterations, at which point the algorithm was terminated as it exceeded a set time limit (time limits for each method are elaborated further in Section 3). Nevertheless, it can be seen that the final optimality gap is very small ($\approx 0.2\%$) and this gap was attained in less than 40 iterations.



(a) Convergence of the BN Method

(b) Convergence of BC Method

Figure 5: Convergence characteristics of the BN and BC methods under the deadline setting for the HN80-I3.0 instance. RMP=Restricted Master Problem; SP=Subproblem.

Elementary and Non-Elementary Paths Table 2 compares results of the column-generation phase of the CG method without and with the elementary path restriction. The key insight is that the two formulations produce the same optimal values. Minor differences in the last two instances may be due to the column-generation phase being terminated before convergence as a CPU time limit of 5760 minutes was reached (CPU time limits applied to all experiments are detailed further in Section 3.1). Restricting attention to elementary paths increases the CPU times, which is not surprising, since finding shortest paths subject to resource constraints is an NP-hard problem. Even though the hybrid strategy employed for finding elementary paths is highly effective, it still cannot compete with the polynomial time Bellman-Ford algorithm used in the original formulation. Nevertheless, it is interesting to observe that the CPU time advantage of the original formulation diminishes as the population size increases. Observe that CG with elementary paths reaches optimality in fewer iterations and has fewer columns in its final RMP in almost all instances.

Table 2: Results of column generation phase of CG method using original shortest path and new elementary shortest path PSP formulations (when no contraflow is allowed).

Instance	**Original Shortest Path PSP**				**New Elementary Shortest Path PSP**			
	Iter #	**Column #**	**CPU Time (mins)**	**Optimal Obj Val**	**Iter #**	**Column #**	**CPU Time (mins)**	**Optimal Obj Val**
HN80-I1.0	79	12251	39	8816	79	11678	124	8816
HN80-I1.1	104	15072	136	10405	95	13853	218	10405
HN80-I1.2	229	22571	799	12116	190	19543	834	12116
HN80-I1.4	152	20184	690	15935	108	17048	404	15935
HN80-I1.7	178	21871	1312	22635	120	19883	2760	22635
HN80-I2.0	197	31418	5760	30490	145	25051	5760	30490
HN80-I2.5	121	22806	5760	46189	129	23513	5760	46188
HN80-I3.0	132	31726	5760	1.960×10^9	87	21233	5760	1.961×10^9

Convergence of the Column Generation Figure 6 demonstrates how the objective function value (y-axis on the left) of the RMP of the CG method evolves over time during its column-generation phase. It also shows the evolution of the objective function value of the best incumbent solution found for the RMP when it is solved as an IP in its last iteration (y-axis on the left), together with the progression of its optimality gap over time (y-axis on the right). It can be seen that there is a steep decline in the objective function value of the restricted master problem within the first 100 seconds, after which the value slowly approaches a minimum. The same trend is observed when the restricted master problem is solved as a MIP, with the optimality gap of the best incumbent solution settling to a value of $\approx 13.5\%$ when the algorithm reached its time limit.

3 Macroscopic Evaluation

This section provides a summary of the results obtained from all four methods under the deadline and minimum clearance time settings, together with the specific conditions under which each method is applied.

3.1 The Deadline Setting

Under the deadline setting, each method maximizes the number of evacuees reaching safety for the HN80-Ix instances (with x $\in [1.0, 3.0]$) within a fixed time horizon $\mathcal{H} = 10$ hours.

The BN Method The results of the BN method are summarized in Tables 3 and 4 without and with contraflow, respectively. As shown in Algorithm 1 (Part I), BN first searches for the tightest time horizon

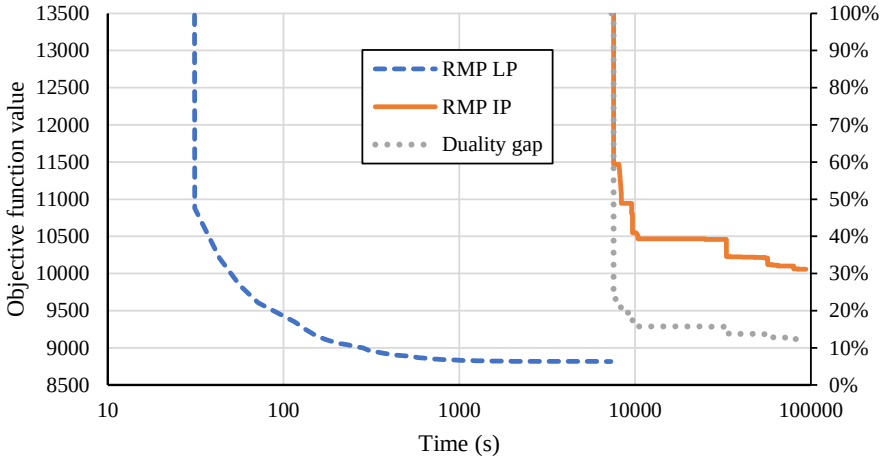


Figure 6: Evolution of the solution quality of the CG method over time for the HN80-I1.0 instance. The RMP LP and RMP IP objective functions are shown in the scale on the left side. The duality gap is expressed in percentage using the scale on the right side.

Table 3: Results of the BN method under the deadline setting without contraflow.

Instance	**Iter #**	**t^* CPU Time (mins)**	**Total CPU Time (mins)**	**z_{RMP} (%)**	**$z_{\text{SP,max}}$ (%)**	**Optimality Gap (%)**
HN80-I1.0	1	131	135	100.0	100.0	0.00
HN80-I1.1	1	111	117	100.0	100.0	0.00
HN80-I1.2	1	104	110	100.0	100.0	0.00
HN80-I1.4	1	92	96	100.0	100.0	0.00
HN80-I1.7	1	103	110	100.0	100.0	0.00
HN80-I2.0	1	79	84	100.0	100.0	0.00
HN80-I2.5	4	38	98	100.0	100.0	0.00
HN80-I3.0	358	5	1449	93.3	93.1	0.20

t^* that preserves the optimal solution of the RMP, $z(\text{RMP}(\mathcal{G}, \mathcal{H}))$. The tables show the CPU time taken for this first phase in column "t^* CPU Time". Each RMP instance in this procedure is given a CPU time limit of 10 minutes. The tables also show the total number of iterations required to complete the entire method as well as the corresponding total CPU time taken. The complete method is allocated a CPU time limit of 24 hours. Finally, the tables show the minimum objective function value of the RMP z_{RMP} and the maximum objective function value of the SP, $z_{\text{SP,max}}$ at termination in terms of evacuation percentage, as well as the optimality gap, i.e., the percentage difference between the upper and lower bounds of the solution given by $\frac{z_{\text{RMP}} - z_{\text{SP,max}}}{z_{\text{SP,max}}}$.

There are three key observations from Tables 3 and 4: (a) Without contraflows, the method converges to an evacuation percentage of 100% for all instances except for HN80-I3.0, for which the method produces a 93.1% evacuation percentage. This instance was also the only one where the method did not converge within the allocated CPU time limit; however, the final optimality gap of 0.20% assures that the obtained solution is very close to being optimal. (b) The method converges after only 1 iteration for all instances except HN80-I2.5 and HN80-I3.0 without contraflow. For instances that took 1 iteration, the bulk of the CPU time is spent on the search for t^*. (c) When using contraflows, the method converges faster on all instances. The increased network capacity provided by contraflows makes the evacuation instances easier to solve.

The BC Method Tables 5 and 6 show results of the BC method under the deadline setting without and with contraflow, respectively. The tables are presented in the same way as Tables 3 and 4 because of the

Table 4: Results of the BN method under the deadline setting with contraflow.

Instance	Iter #	t^* CPU Time (mins)	Total CPU Time (mins)	z_{RMP} (%)	$z_{\text{SP,max}}$ (%)	Optimality Gap (%)
HN80-I1.0	1	43	47	100.0	100.0	0.00
HN80-I1.1	1	43	45	100.0	100.0	0.00
HN80-I1.2	1	41	43	100.0	100.0	0.00
HN80-I1.4	1	48	50	100.0	100.0	0.00
HN80-I1.7	1	42	45	100.0	100.0	0.00
HN80-I2.0	1	42	45	100.0	100.0	0.00
HN80-I2.5	1	36	39	100.0	100.0	0.00
HN80-I3.0	1	35	37	100.0	100.0	0.00

similarities between the BN and BC methods. However, the CPU time limits allocated for the BC method are different. Each RMP instance in the search for t^* is given a limit of 10 minutes, whereas the entire method is allocated only 2 hours of CPU time due its faster convergence.

The key observations from these two tables are as follows: (a) The method evacuates everyone for all instances when contraflow is allowed, and for instances HN80-Ix with x $\in [1.0, 1.7]$ when contraflow is not allowed. However, unlike the BN method, this method converges to optimal solutions for all instances as evidenced by their 0% optimality gaps. (b) For instances in which a 100% evacuation rate is achieved, the method converges in just 1 iteration, and the bulk of the CPU time in these instances is spent searching for t^*. (c) Except for instance HN80-I1.4, the method converges faster when contraflow is allowed. As mentioned earlier, the BC method is extremely effective in finding high-quality evacuations quickly.

Table 5: Results of the BC method under the deadline setting without contraflow.

Instance	Iter #	t^* CPU Time (mins)	Total CPU Time (mins)	z_{RMP} (%)	$z_{\text{SP,max}}$ (%)	Optimality Gap (%)
HN80-I1.0	1	0.35	0.51	100.0	100.0	0.00
HN80-I1.1	1	10.19	10.28	100.0	100.0	0.00
HN80-I1.2	1	10.15	10.24	100.0	100.0	0.00
HN80-I1.4	1	1.13	1.21	100.0	100.0	0.00
HN80-I1.7	1	10.21	10.32	100.0	100.0	0.00
HN80-I2.0	14	0.09	1.80	96.9	96.9	0.00
HN80-I2.5	22	0.28	3.43	81.4	81.4	0.00
HN80-I3.0	138	0.01	16.22	68.5	68.5	0.00

Table 6: Results of the BC method under the deadline setting with contraflow.

Instance	Iter #	t^* CPU Time (mins)	Total CPU Time (mins)	z_{RMP} (%)	$z_{\text{SP,max}}$ (%)	Optimality Gap (%)
HN80-I1.0	1	0.19	0.27	100.0	100.0	0.00
HN80-I1.1	1	0.22	0.30	100.0	100.0	0.00
HN80-I1.2	1	0.28	0.37	100.0	100.0	0.00
HN80-I1.4	1	2.31	2.42	100.0	100.0	0.00
HN80-I1.7	1	0.50	0.60	100.0	100.0	0.00
HN80-I2.0	1	0.61	0.70	100.0	100.0	0.00
HN80-I2.5	1	0.38	0.48	100.0	100.0	0.00
HN80-I3.0	1	0.20	0.30	100.0	100.0	0.00

The CPG Method Results for the CPG method are outlined in Tables 7 and 8, respectively. The tables show the number of iterations, the CPU time, and the evacuation percentage. The CPG algorithm is allowed a maximum of 10 iterations, after which it was terminated even if it still had critical nodes which are not completely evacuated. Additionally, the RMP and SP are each allocated a CPU time limit of 1 hour.

The tables show that a 100% evacuation rate is achieved in all instances but HN80-I2.5 and HN80-I3.0 without contraflow. These two instances were terminated by the iteration limit. The results also show relatively short CPU times of less than a minute for instances HN80-Ix with x $\in [1.0, 2.0]$, and a spike in CPU time for instance HN80-I3.0 both with and without contraflow. This spike is possibly caused by the increase in difficulty of solving these instances. The positive correlation between the number of iterations required and the value of x for the instances provides further evidence that the heuristic requires more iterations to evacuate more evacuees. Similar to the BN and BC methods, the CPU times for each instance are smaller when contraflow is allowed. On almost all instances, the CPG method is very effective as well.

Table 7: Results of deadline setting CPG method without contraflow.

Instance	Iter #	CPU Time (mins)	Evacuation Percentage (%)
HN80-I1.0	2	0.05	100.0
HN80-I1.1	3	0.07	100.0
HN80-I1.2	3	0.08	100.0
HN80-I1.4	3	0.10	100.0
HN80-I1.7	3	0.15	100.0
HN80-I2.0	4	0.69	100.0
HN80-I2.5	10	11.04	96.7
HN80-I3.0	10	96.48	86.3

Table 8: Results of the CPG method under the deadline setting with contraflow.

Instance	Iter #	CPU Time (mins)	Evacuation Percentage (%)
HN80-I1.0	1	0.04	100.0
HN80-I1.1	1	0.03	100.0
HN80-I1.2	2	0.04	100.0
HN80-I1.4	2	0.05	100.0
HN80-I1.7	2	0.06	100.0
HN80-I2.0	3	0.15	100.0
HN80-I2.5	4	2.44	100.0
HN80-I3.0	4	21.23	100.0

The CG Method The results of the CG method without and with contraflow are summarized in Tables 9 and 10, respectively. They show the number of iterations and the CPU time taken by the column generation phase to converge, the total number of columns generated by the phase, the total CPU time taken by the entire CG method, the optimality gap of the final integer solution of the RMP, calculated using formula $\frac{z_{\text{RMP,MIP}} - z_{\text{RMP,LP}}}{z_{\text{RMP,MIP}}}$ where $z_{\text{RMP,LP}}$ and $z_{\text{RMP,MIP}}$ are the final objective function values of the column generation and the MIP, respectively, and the evacuation percentage achieved by the method. The following CPU time limits are applied to the method: 96 hours for the column generation phase and 24 hours for the final MIP.

The tables show that the final MIP reaches its 24 hour time limit in all instances. The column-generation phase also reaches its 96 hour time limit in the most challenging instances, HN80-Ix with x $\in [2.0, 3.0]$ without contraflow and HN80-I3.0 with contraflow. A 100% evacuation rate is achieved in instances HN80-Ix with x $\in [1.0, 1.4]$ without contraflow and with x $\in [1.0, 2.0]$ with contraflow. Correspondingly, the optimality gaps of these instances are relatively low, being $< 20\%$ when contraflow is not allowed and $< 10\%$ when it is allowed. These results indicate that the quality of the solutions obtained for these instances are relatively

high. The large optimality gap of the other instances can be explained by the large penalty incurred in their objective function values because not everyone is evacuated safely in the final integer solutions. A steady increase in CPU times is also observed as the population scaling factor x increases. For the same instance, the CPU times are smaller when contraflow is allowed, and so are the optimality gaps when 100% evacuation is achieved.

Table 9: Results of the CG method under the deadline setting without contraflow.

Instance	**Iter #**	**Column #**	**Column Generation CPU Time (mins)**	**Total CPU Time (mins)**	**Optimality Gap (%)**	**Evacuation Percentage (%)**
HN80-I1.0	75	11626	153	1593	13.5	100.0
HN80-I1.1	88	13463	261	1701	14.9	100.0
HN80-I1.2	191	18020	792	2232	15.0	100.0
HN80-I1.4	116	17412	681	2121	16.9	100.0
HN80-I1.7	119	19415	1275	2715	100.0	96.1
HN80-I2.0	163	25883	5760	7200	100.0	92.5
HN80-I2.5	126	23185	5760	7200	100.0	89.7
HN80-I3.0	95	23146	5760	7200	63.0	81.1

Table 10: Results of the CG method under the deadline setting with contraflow.

Instance	**Iter #**	**Column #**	**Column Generation CPU Time (mins)**	**Total CPU Time (mins)**	**Optimality Gap (%)**	**Evacuation Percentage (%)**
HN80-I1.0	60	3757	25	1465	4.2	100.0
HN80-I1.1	69	4636	55	1495	4.5	100.0
HN80-I1.2	105	5398	137	1577	5.1	100.0
HN80-I1.4	152	8015	304	1744	5.0	100.0
HN80-I1.7	226	11880	829	2269	7.1	100.0
HN80-I2.0	310	14971	1899	3339	9.9	100.0
HN80-I2.5	453	21474	5741	7181	99.9	99.7
HN80-I3.0	201	17094	5760	7200	99.9	99.8

Comparison of the Evacuation Rates Figures 7 and 8 compare the evacuation percentages achieved by all four methods. Figure 8 compares the performance of each method when contraflow is allowed, and it can be seen that all methods achieve 100% evacuation for all instances, except for the CG method which achieves 99.7% and 99.8% evacuation for HN80-I2.5 and HN80-I3.0, respectively. The performance comparison of each method without contraflow is shown in Figure 7 which paints a slightly different picture. All methods achieve 100% evacuation only up to instance HN80-I1.4. For instances with larger population scaling factors, the evacuation percentage starts to drop off significantly for some methods and slightly for others. A common trend prevalent in these instances (HN80-Ix with $x \in [1.7, 3.0]$) is that the BN method consistently produces the highest evacuation percentage, followed by CPG. Although both methods generate evacuation plans with the same characteristics, their performance disparity could be attributed to the heuristic nature of CPG. The BN method, although not strictly optimal, returns optimal results when the integrality constraints on flows are relaxed. The lower evacuation rates of the BC and CG methods can be explained by the additional constraints imposed on their respective evacuation plans. The BC method produces convergent paths and the CG method generates non-preemptive evacuation schedules. These constraints limit their ability to match the performance of the BN and CPG methods in a macroscopic analysis. Indeed, the benefits of these

methods cannot be captured in a macroscopic analysis, as they concern human behavior and the realities of enforcing evacuation plans. Note also that the CG method is unique in that the evacuation rates of its plans are limited to the set of predefined flow rates γ that was specified in Section 2, whereas the same limitation does not apply to the other methods. The method's performance is dependent on these preset γ values, and its performance would change given different sets of γ values.

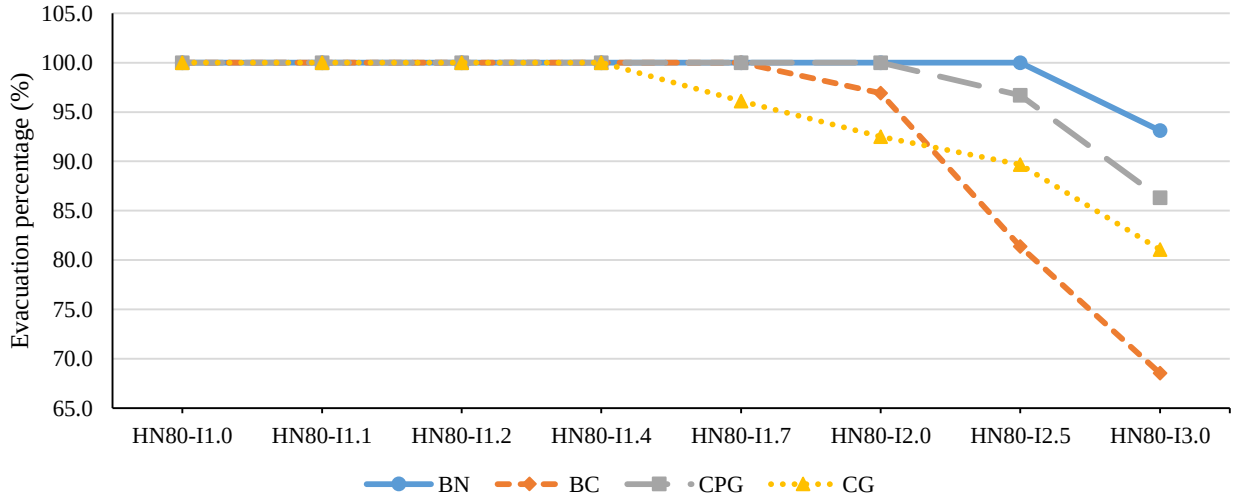


Figure 7: Summary of evacuation percentage under deadline setting without contraflow.

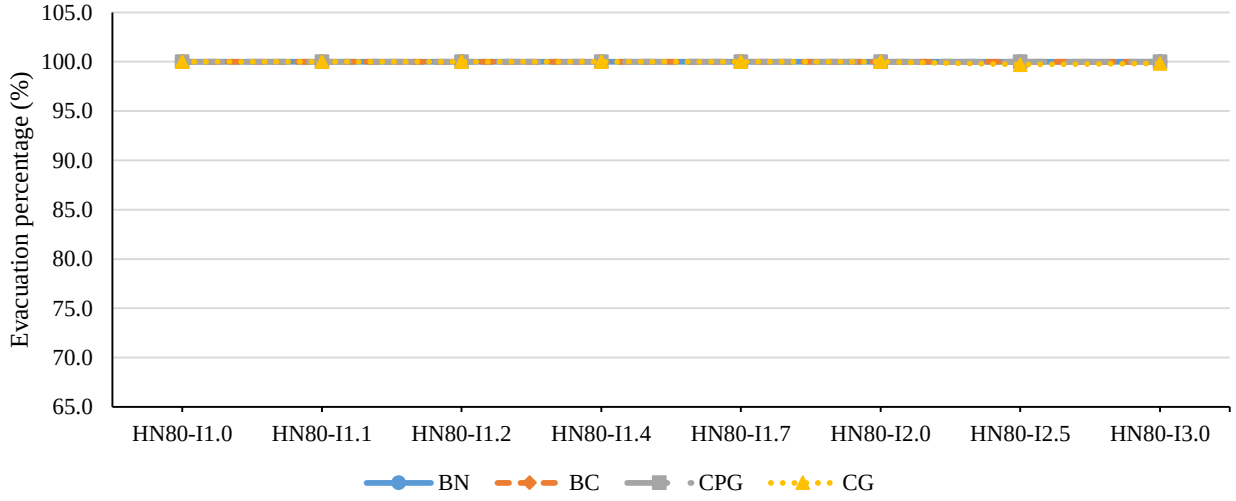


Figure 8: Summary of evacuation percentage under deadline setting with contraflow.

Comparison of the CPU Times Tables 11 and 12 compile the total CPU times of all four methods without and with contraflow, respectively. The tables also show average CPU times for each method across all instances with their associated standard errors representing uncertainty. The standard errors are relatively large for most methods due to the spread in CPU times across the various instances. Nevertheless, a quantitative comparison can still be made. Regardless of whether contraflow is allowed or not, the BC and CPG methods consistently consume the smallest amount of CPU time, followed by the BN method, and finally the CG method. In addition to being the most expensive method, the CG method reaches its CPU time limit in a few of the more challenging instances. The difference in run times across the methods is not surprising due to the different algorithms employed as well as the varying constraints imposed on their evacuation plans.

Table 11: CPU times of all methods under deadline setting without contraflow.

Instance	**Total CPU Time (mins)**			
	BN	**BC**	**CPG**	**CG**
HN80-I1.0	135	0.51	0.05	1593
HN80-I1.1	117	10.28	0.07	1701
HN80-I1.2	110	10.24	0.08	2232
HN80-I1.4	96	1.21	0.10	2121
HN80-I1.7	110	10.32	0.15	2715
HN80-I2.0	84	1.80	0.69	7200
HN80-I2.5	98	3.43	11.04	7200
HN80-I3.0	1449	16.22	96.48	7200
Average	275 ± 168	6.75 ± 2.04	13.58 ± 11.92	3995 ± 946

Table 12: CPU times of all methods under deadline setting with contraflow.

Instance	**Total CPU Time (mins)**			
	BN	**BC**	**CPG**	**CG**
HN80-I1.0	47	0.27	0.04	1465
HN80-I1.1	45	0.30	0.03	1495
HN80-I1.2	43	0.37	0.04	1577
HN80-I1.4	50	2.42	0.05	1744
HN80-I1.7	45	0.60	0.06	2269
HN80-I2.0	45	0.70	0.15	3339
HN80-I2.5	39	0.48	2.44	7181
HN80-I3.0	37	0.30	21.23	7200
Average	44 ± 2	0.68 ± 0.25	3.00 ± 2.62	3284 ± 880

The disparity in CPU times between the BN and BC methods deserves special attention as their algorithms share a lot of similarities, both utilizing Benders decomposition. This discrepancy is explained by the sizes of the RMP and SP of both methods, which are summarized in Table 13. Although the BC method solves three problems (the RMP, SP, and DMWP) during each iteration, and the BM method solves only two, the number of variables and constraints in the latter is significantly larger, leading to correspondingly larger CPU times. Moreover the nature of the subproblem is fundamentally different. The Benders subproblems of the BC method are maximum flows, whereas those of the BN method are multi-commodity flow problems, where evacuees from an evacuation node can be seen as a single commodity.

Table 13: Number of constraints and variables in the problems of the BC and BN methods.

		BC Method	**BN Method**
Restricted Master Problem	# Constraints	$\|\mathcal{T}\| + \|\mathcal{E} \cup \mathcal{T}\| + \|\mathcal{A}\| + \|\mathcal{E}\| + \|\mathcal{C}\| + 1$	$\|\mathcal{E}\|^2 + 2\|\mathcal{T}\|\|\mathcal{E}\| + \|\mathcal{E}\| + \|\mathcal{A}\|\|\mathcal{E}\| + \|\mathcal{A}\| + \|\mathcal{A}_c\| + \|\mathcal{C}\| + 1$
	# Variables	$2\|\mathcal{A}\| + 1$	$2\|\mathcal{A}\|\|\mathcal{E}\| + \|\mathcal{A}_c\| + 1$
Subproblem	# Constraints	$\|\mathcal{T}\|\|\mathcal{H}\| + \|\mathcal{A}\|\|\mathcal{H}\| + \|\mathcal{E}\|$	$\|\mathcal{T}\|\|\mathcal{H}\|\|\mathcal{E}\| + \|\mathcal{E}\| + \|\mathcal{A}\|\|\mathcal{H}\|\|\mathcal{E}\| + \|\mathcal{A}\|\|\mathcal{H}\|$
	# Variables	$\|\mathcal{A}\|\|\mathcal{H}\|$	$\|\mathcal{A}\|\|\mathcal{H}\|\|\mathcal{E}\|$
Dual of Magnanti-Wong Problem	# Constraints	$\|\mathcal{T}\|\|\mathcal{H}\| + \|\mathcal{A}\|\|\mathcal{H}\| + \|\mathcal{E}\|$	-
	# Variables	$\|\mathcal{A}\|\|\mathcal{H}\| + 1$	-

3.2 The Minimum Clearance Time Setting

In the minimum clearance time experiments, each method finds the smallest amount of time needed to evacuate the entire region, i.e., to achieve 100% evacuation. A precise definition of minimum clearance time h^* is given by Equation (98) in Part I of this paper.

As explained in Section 9 (Part I), the CG method's multi-objective function simultaneously minimizes the overall evacuation time and maximizes the evacuation percentage. As such, h^* can be determined from its deadline setting results for instances where 100% evacuation is achieved by identifying the time at which the last evacuee reaches its safe node. For the BN and BC methods, Algorithm 3 (Part I) is applied to find h^*. A binary search procedure is first applied to find a lower bound $h^\dagger$ to the minimum clearance time. A sequential search procedure is then applied to find h^*. For both methods, a CPU time limit of 10 minutes is applied when solving each RMP instance in the binary search procedure. However, in the subsequent sequential search, a CPU time limit of 10 hours is used for each search step of the BN method, whereas a time limit of 2 hours is used for that of the BC method. The different time limits are selected to cater to the correspondingly different convergence times of each method. The approach outlined in Section 9 (Part I) is used to find h^* for the CPG method. Under the minimum clearance time setting, the method's RMP and SP are each allocated only 2 minutes of CPU time.

The total CPU times and the value of h^* for each method are presented in Tables 14 and 15 for cases without and with contraflow, respectively. For the CG method, results are only shown for instances where the method achieved 100% evacuation under the deadline setting. Figures 9 and 10 compare each method without and with contraflow, respectively. As expected, the clearance times increase as the population size grows and using contraflows helps to reduce them. Across all instances, the BN method consistently produces the smallest clearance time, followed by CPG, BC, and CG. A slight anomaly is observed in the results of the CG method for instances HN80-I1.7 and HN80-I2.0 when contraflow is allowed, where the clearance time is smaller for the latter instance. This result is likely caused by the early termination of the RMP's last iteration, in which it reached its 24 hour CPU time limit.

Finally, Table 16 shows the average CPU times consumed by each method under the minimum clearance time setting. Whether contraflow is allowed or not, the BC method consistently consumes the least amount of CPU time, followed by CPG, BN, and CG.

Table 14: Minimum clearance times of all methods without contraflow.

Instance	BN		BC		CPG		CG	
	CPU Time (mins)	Min Clear Time (mins)	CPU Time (mins)	Min Clear Time (mins)	CPU Time (mins)	Min Clear Time (mins)	CPU Time (mins)	Min Clear Time (mins)
HN80-I1.0	814	260	11	335	29	280	1593	455
HN80-I1.1	1936	290	12	365	27	315	1701	555
HN80-I1.2	650	300	10	395	38	330	2232	570
HN80-I1.4	1440	350	13	450	35	380	2121	600
HN80-I1.7	690	405	10	535	55	445	-	-
HN80-I2.0	705	470	10	630	41	520	-	-
HN80-I2.5	1374	575	2	770	94	625	-	-
HN80-I3.0	1128	675	16	925	67	815	-	-

Table 15: Minimum clearance times of all methods with contraflow.

Instance	BN		BC		CPG		CG	
	CPU Time (mins)	Min Clear Time (mins)	CPU Time (mins)	Min Clear Time (mins)	CPU Time (mins)	Min Clear Time (mins)	CPU Time (mins)	Min Clear Time (mins)
HN80-I1.0	500	195	26	225	46	200	1465	235
HN80-I1.1	624	205	27	240	56	210	1495	255
HN80-I1.2	713	225	47	260	57	235	1577	285
HN80-I1.4	49	250	3	285	79	260	1744	320
HN80-I1.7	207	290	1	335	56	320	2269	575
HN80-I2.0	49	335	10	390	63	385	3339	535
HN80-I2.5	52	405	10	475	99	440	-	-
HN80-I3.0	59	475	0	560	106	500	-	-

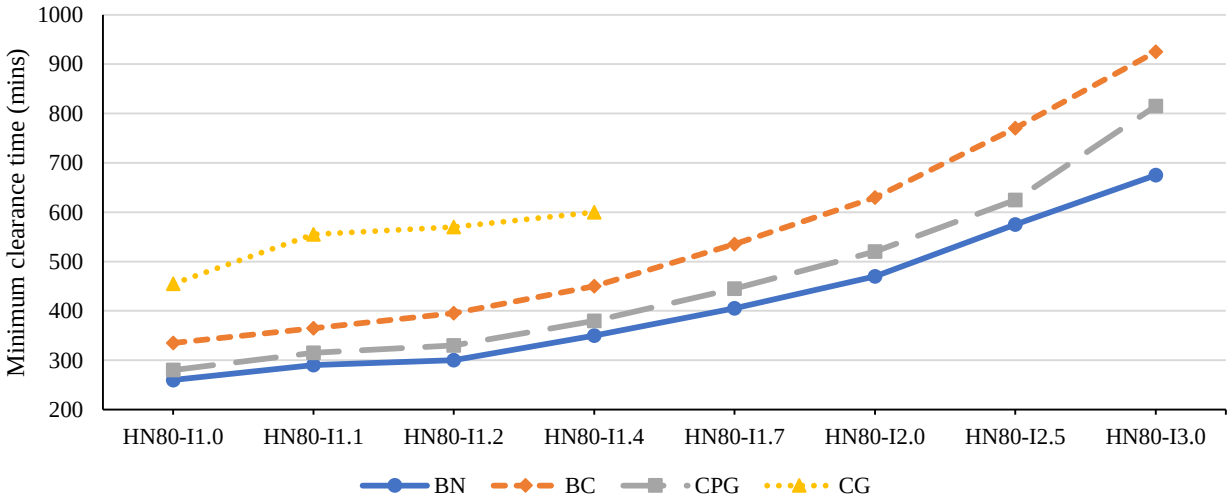


Figure 9: Summary of minimum clearance time without contraflow.

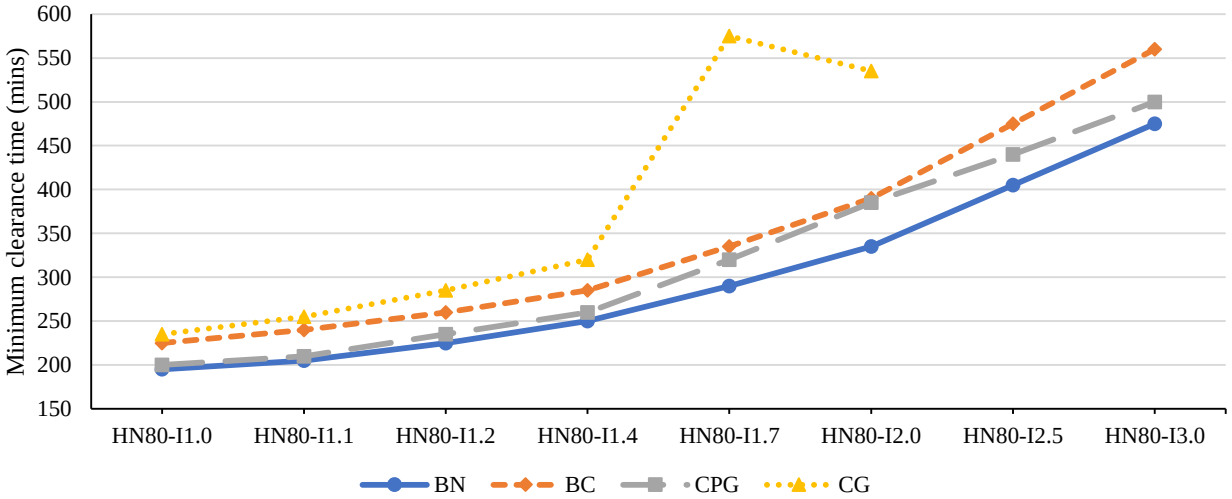


Figure 10: Summary of Minimum clearance time with contraflow.

Table 16: Average CPU times of all methods under minimum clearance time setting.

Method	**Average CPU Time (mins)**	
	Without Contraflow	**With Contraflow**
BN	1092 ± 164	282 ± 101
BC	11 ± 1	16 ± 6
CPG	48 ± 8	70 ± 8
CG	1912 ± 156	1981 ± 297

4 Microscopic Evaluation

The four evacuation methods presented in this paper are macroscopic and do not capture individual evacuee behaviors, movements, and interactions, as well as vehicle dynamics such as acceleration and deceleration, lane changing, and collision avoidance which are all reflective of what would actually happen in a real-world evacuation scenario. These factors could introduce unanticipated delays or induce congestion, both of which would negatively affect the performance of an evacuation plan. Unfortunately, these factors are not captured in the macroscopic evaluation.

This section presents a microscopic evaluation of the four evacuation algorithms. Each evacuation plan was simulated, using a road traffic simulation package SUMO (Simulation of Urban Mobility) [9]. SUMO is a full-featured, open-source, microscopic traffic flow simulation suite developed primarily by researchers of the Institute of Transportation Systems at the German Aerospace Center. Its traffic simulator realistically models the traversal behavior of each vehicle in a road network by computing each vehicle's instantaneous speed according to the speed limit, a safe distance to be maintained from a leading vehicle, and the leading vehicle's speed according to a car-following model described by Krauß [10] and a lane-changing model described by Erdmann [5]. This simulator not only allows for ascertaining and evaluating the robustness of the evacuation plans generated in terms of how they would actually perform in a real-world evacuation scenario; it can also reveal the benefits of convergent and non-preemptive plans.

The simulations utilize actual road network information of the HN region, including speed limits, lane counts, and GPS coordinates of each node to construct the road network for SUMO. They also define the demand by specifying, for each vehicle, an evacuation path and a departure time derived from plans generated in the deadline and minimum clearance time experiments.

4.1 The Deadline Setting

Table 17 shows the evacuation percentages obtained from simulating the evacuation plans generated by the deadline setting experiments. Figures 11 and 12 present the same information in graphical form for settings without and with contraflow, respectively. To get a better perspective on the results shown in these figures, they should be compared with their counterparts from the deadline experiments, Figures 7 and 8, which show corresponding evacuation percentages produced by the four algorithms. When contraflows are not allowed, the evacuation percentages start to decrease sooner as the population scaling factor increases. More importantly, the clear performance advantage of the BN and CPG methods are not preserved in these simulations. In some instances, they are outperformed by the other two methods whereas, for some other instances, they could barely outperform the CG method. When contraflows are allowed, the methods are no longer able to achieve a 100% evacuation rate. In fact, the CPG method cannot achieve a 100% evacuation on any instance, whereas the evacuation percentage of other methods start decreasing after instance HN80-I2.0, with the BN method having the steepest drop.

Table 17: Evacuation percentages obtained from SUMO simulations of deadline setting evacuation plans.

Instance	Evacuation Percentage (%)							
	Without Contraflow				With Contraflow			
	BN	BC	CPG	CG	BN	BC	CPG	CG
HN80-I1.0	100.0	100.0	100.0	100.0	100.0	100.0	97.6	100.0
HN80-I1.1	100.0	100.0	98.8	100.0	100.0	100.0	95.0	100.0
HN80-I1.2	100.0	100.0	97.6	100.0	100.0	100.0	94.3	100.0
HN80-I1.4	100.0	100.0	95.1	100.0	100.0	100.0	98.3	100.0
HN80-I1.7	91.1	95.6	94.9	96.1	100.0	100.0	94.8	100.0
HN80-I2.0	94.4	85.2	90.6	92.2	100.0	100.0	94.2	100.0
HN80-I2.5	80.2	70.9	84.8	80.2	94.3	100.0	89.7	98.9
HN80-I3.0	70.4	61.4	67.7	67.4	70.2	92.3	77.7	91.3

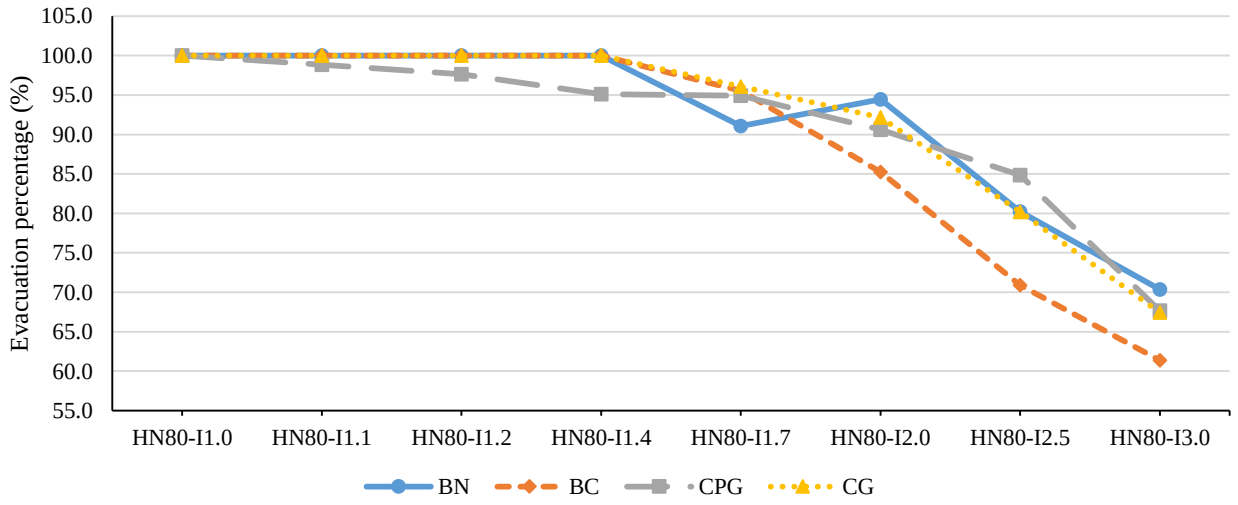


Figure 11: Evacuation percentages obtained from SUMO simulations of deadline setting evacuation plans without contraflow.

To obtain a better understanding of how closely the simulated results match those of deadline experiments, it is interesting to normalize the evacuation percentage obtained from the simulations relative to those from the deadline settings. More precisely, the normalized evacuation percentage is calculated as follows:

$$\text{Normalized evacuation percentage} = \frac{\text{microscopic evacuation percentage}}{\text{macroscopic evacuation percentage}} \quad (1)$$

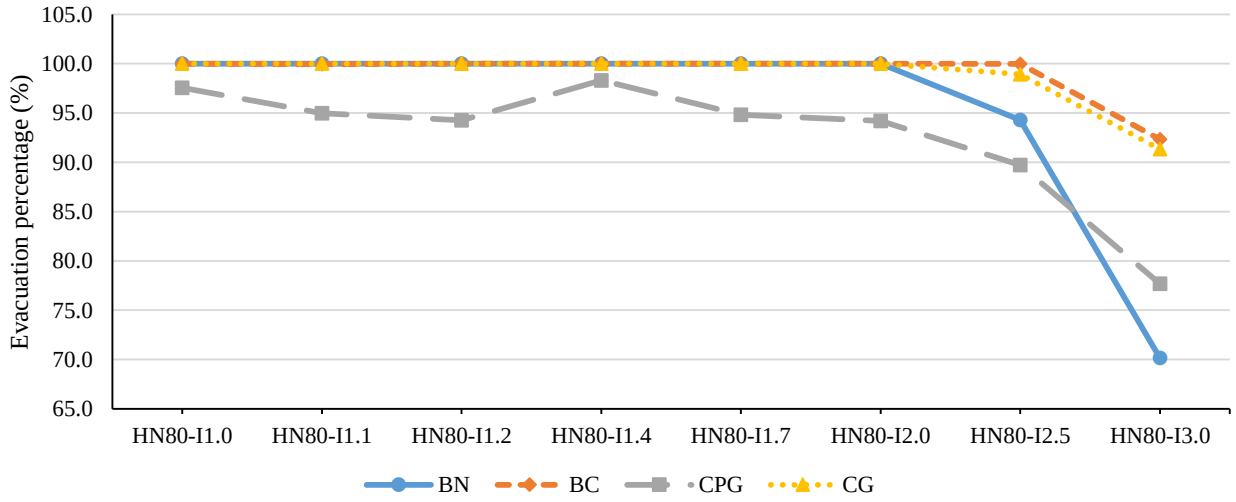


Figure 12: Evacuation percentages obtained from SUMO simulations of deadline setting evacuation plans with contraflow.

These normalized evacuation percentages are summarized in Table 18. Average values for each method are also shown with their corresponding uncertainties calculated using standard errors. A normalized value of 1.0 is achieved by most methods for the smaller population instances. However, as the population increases, the normalized values start to decrease, in some instances down to 0.70 in the case of the BN method with contraflow. The general observation is that, regardless of what method is used and whether contraflows are allowed, microscopic results tend to diverge from macroscopic results as the population grows. *However, what is most interesting is how the microscopic results change the ranking of the methods. The BC and CG methods not only have the highest normalized ratios. They also have the highest evacuation percentages, especially when contraflows are allowed.* By using convergent plans and non-preemptive evacuations, BC and CG produce plans whose objectives are realistic from a microscopic standpoint.

Table 18: SUMO evacuation percentages normalized relative to deadline setting evacuation percentages.

Instance	**Normalized Evacuation Percentage**							
	Without Contraflow				**With Contraflow**			
	BN	**BC**	**CPG**	**CG**	**BN**	**BC**	**CPG**	**CG**
HN80-I1.0	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00
HN80-I1.1	1.00	1.00	0.99	1.00	1.00	1.00	0.95	1.00
HN80-I1.2	1.00	1.00	0.98	1.00	1.00	1.00	0.94	1.00
HN80-I1.4	1.00	1.00	0.95	1.00	1.00	1.00	0.98	1.00
HN80-I1.7	0.91	0.96	0.95	1.00	1.00	1.00	0.95	1.00
HN80-I2.0	0.94	0.88	0.91	1.00	1.00	1.00	0.94	1.00
HN80-I2.5	0.80	0.87	0.88	0.89	0.94	1.00	0.90	0.99
HN80-I3.0	0.76	0.90	0.78	0.83	0.70	0.92	0.78	0.91
Average	0.93 ± 0.03	0.95 ± 0.02	0.93 ± 0.03	0.97 ± 0.02	0.96 ± 0.04	0.99 ± 0.01	0.93 ± 0.02	0.99 ± 0.01

4.2 The Minimum Clearance Time Setting

This section presents the microscopic results for the minimum clearance time setting. Table 19 shows minimum clearance times obtained by the simulations, whereas Figures 13 and 14 show the same information in graphical form without and with contraflows, respectively. To place the results into context, Figure 13 is compared with Figure 9 which contain corresponding algorithmic results, whereas Figure 14 is compared to

Figure 10. The main insight from these results is that the gaps between the various methods either decrease or disappear entirely. For instance, looking at Figure 14, there is no clear winner in terms of minimum clearance time from the simulations.

Table 19: SUMO minimum clearance time results.

Instance	Minimum Clearance Time (mins)							
	Without Contraflow				With Contraflow			
	BN	BC	CPG	CG	BN	BC	CPG	CG
HN80-I1.0	343	399	373	428	274	249	227	252
HN80-I1.1	359	438	408	539	286	276	280	288
HN80-I1.2	410	486	432	533	333	297	358	315
HN80-I1.4	526	563	477	583	385	343	305	341
HN80-I1.7	539	672	566	-	398	424	440	555
HN80-I2.0	695	792	753	-	529	478	494	539
HN80-I2.5	833	967	797	-	571	592	648	-
HN80-I3.0	998	1148	1109	-	830	705	778	-

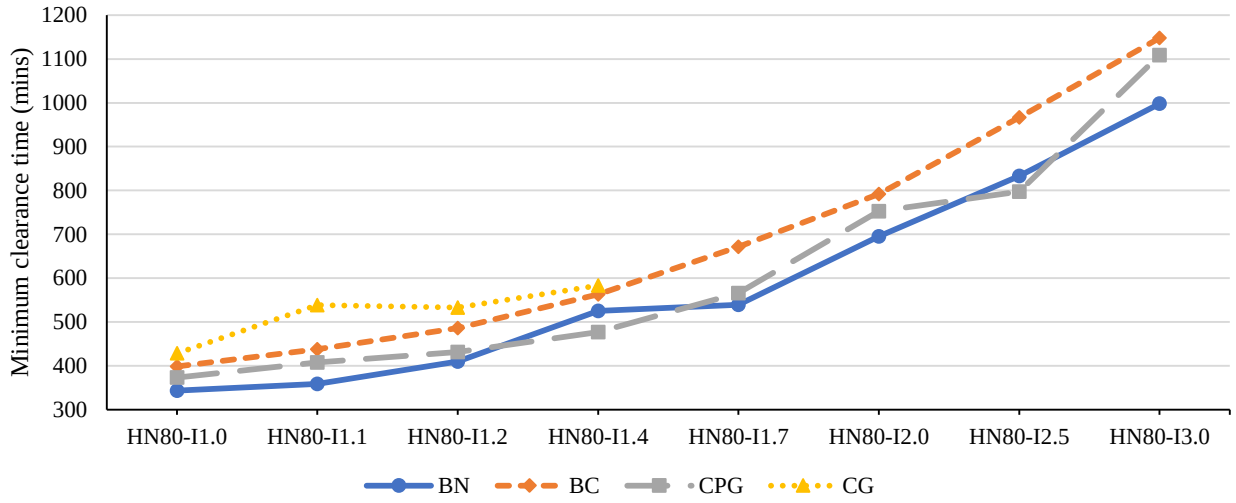


Figure 13: SUMO minimum clearance time without contraflow.

Table 20 summarizes the normalized values for this setting. The table reveals statistically significant differences in the average normalized values of each method.

The CG method has normalized values which are closest to 1.0, followed by the BC, CPG, and BN methods. These results indicate that the CG method produces minimum clearance times that are most reproducible in simulations, i.e., their times are the least optimistic and most precisely reflect what could be achieved in a real-world setting. The results are interesting since the CG method is the only method that utilizes non-preemptive evacuation schedules. A possible rationale for this result is that the step response curves representing evacuation flow rates over time prevent the transportation network from being utilized to its full capacity systematically. Indeed, the flow rate is fixed to a constant value and cannot be increased to saturate available arc capacities at any given time unlike the flow rates of preemptive schedules. While this characteristic of the CG hampers its macroscopic performance, its conservative utilization of the transportation network has a positive side benefit in that it is less likely to cause congestion. The fact that the CG method's normalized values are occasionally less than 1.0 warrants further explanation. This result is due to the discretization into 5 minute intervals in the time-expanded graph. This discretization forces travel time along each arc to be rounded up to the nearest 5 minute multiple when the arc is represented in the time-expanded graph, even though the actual travel time may be less than this multiple. This quantization

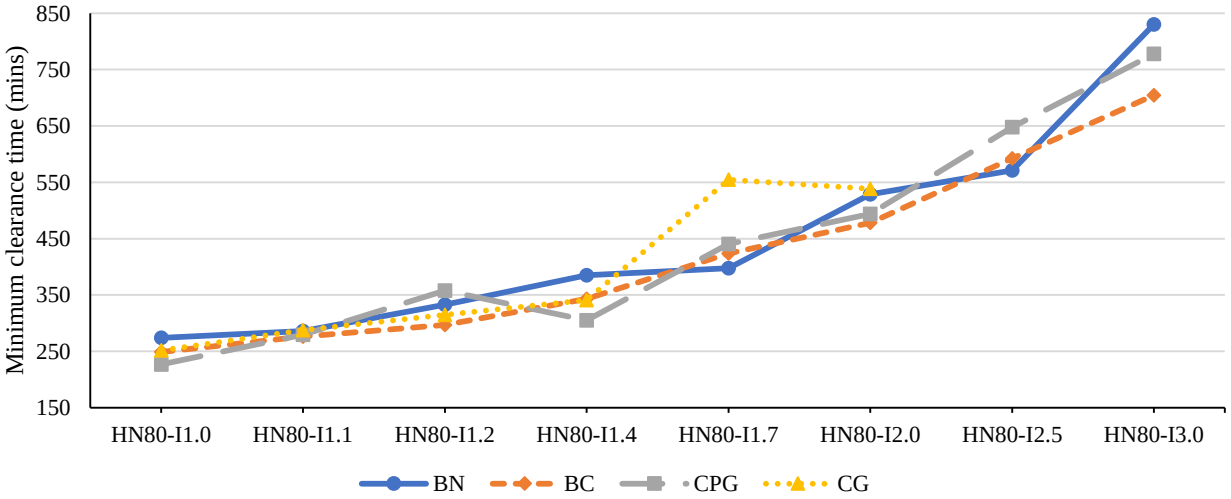


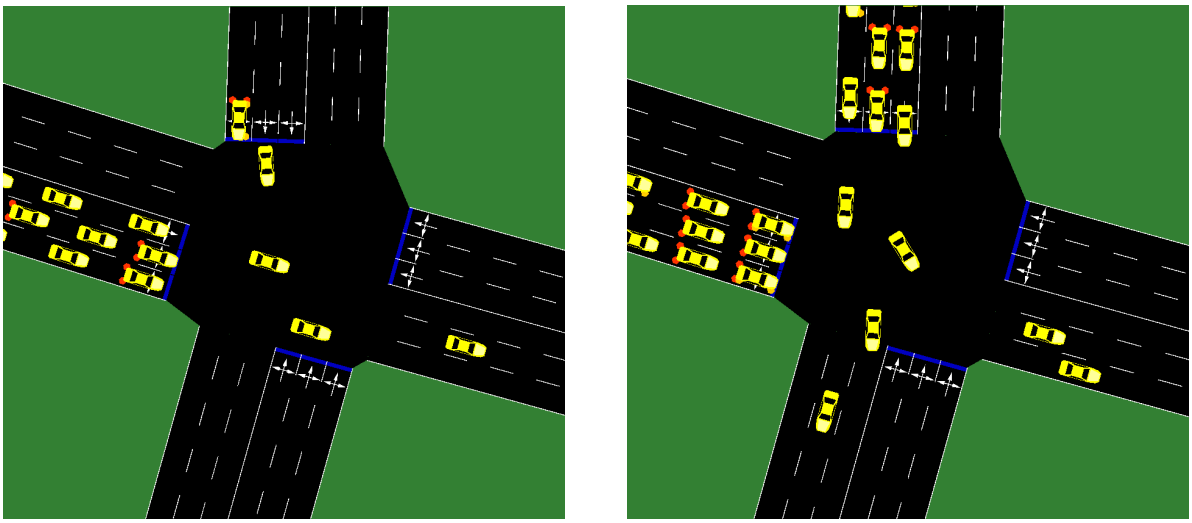
Figure 14: SUMO minimum clearance time with contraflow.

Table 20: SUMO minimum clearance time normalized results.

Instance	**Normalized Minimum Clearance Time**							
	Without Contraflow				**With Contraflow**			
	BN	**BC**	**CPG**	**CG**	**BN**	**BC**	**CPG**	**CG**
HN80-I1.0	1.32	1.19	1.33	0.94	1.41	1.11	1.13	1.07
HN80-I1.1	1.24	1.20	1.29	0.97	1.40	1.15	1.33	1.13
HN80-I1.2	1.37	1.23	1.31	0.94	1.48	1.14	1.52	1.10
HN80-I1.4	1.50	1.25	1.26	0.97	1.54	1.20	1.17	1.06
HN80-I1.7	1.33	1.26	1.27	-	1.37	1.26	1.38	0.96
HN80-I2.0	1.48	1.26	1.45	-	1.58	1.23	1.28	1.01
HN80-I2.5	1.45	1.26	1.28	-	1.41	1.25	1.47	-
HN80-I3.0	1.48	1.24	1.36	-	1.75	1.26	1.56	-
Average	1.40 ± 0.03	1.23 ± 0.01	1.32 ± 0.02	0.95 ± 0.01	1.49 ± 0.05	1.20 ± 0.02	1.36 ± 0.06	1.06 ± 0.03

causes total travel times along an evacuation path to be a slight overestimate of actual times. In the simulation of other methods, this overestimation gets canceled out by congestion-induced delays. However, the CG method produces flows that tend to underutilize the transportation network, which result in less congestion and delays. As such, the travel time overestimation does not get canceled out and is reflected in the normalized clearance times.

Method BC also produces normalized values close to 1.0. To understand why this was the case, it is useful to take a closer look at the visualizations of the simulations and, in particular, busy road intersections and to compare the congestion severity at these intersections for the BN and CPG methods. Figures 15a and 15b show visualizations of the same intersection taken from simulations of the BC and BN methods, respectively. The inspections reveal that, even though congestion is present at these intersections in the BC simulations, it was consistently less severe than those of the other two methods. The BC method forces traffic flows at intersections to converge onto a single outgoing road. This limitation is not present in the other methods, which allow for convergent, divergent, and sometimes even crossing paths at any given intersection which often leads to more severe congestion. These observations support the normalized values obtained for the BC method which indicate its microscopic and macroscopic results are much closer.



(a) Convergent traffic flow of BC method

(b) Non-convergent traffic flow of BN method

Figure 15: Visualization of simulated traffic flow at an intersection produced by evacuaton plans of the BC and BN methods. Red vehicles are stopped.

5 Perspectives

Extensions of the Optimization Algorithms The macroscopic and microscopic evaluations contain two important findings.

1. Adding constraints on the evacuations, e.g., path convergence and non-preemption, increases the fidelity and overall effectiveness of the evacuation plans. In particular, for large population sizes, problems C-ZEPP and NP-ZEPP produce solutions which are dominated in macroscopic evaluations but become superior when validated in microscopic simulators.

2. Adding constraints on the evacuations may lead to more elegant optimization approaches. Indeed, problems C-ZEPP and NP-ZEPP are solved using exact Benders decomposition (method BC) and column generation (method CG); this is not the case of Problem ZEPP, which is approximated by methods BN and CPG. This elegance may lead to improved computational performance: method BC is the most efficient method studied in this paper, but method CG is the most demanding.

An obvious direction for future work is to study convergent, non-preemptive zone-based evacuation planning. However, enhancing method BC with non-preemption is not an easy task. The subproblem becomes a complex scheduling problem that was studied by Even et al. [6]. The key technical issue is to find effective combinatorial Benders cuts to link the subproblem and the restricted master problem. Method CG seems easier to extend with convergent paths, since they only affect the master problem. It remains to evaluate whether the resulting restricted master problem can be solved effectively.

Fidelity of the Macroscopic Approaches This study demonstrated that macroscopic approaches such as methods BC and CG are of high fidelity. Their evacuation plans do not degrade in any significant way when evaluated with a microscopic simulator. This finding is interesting on its own. Indeed, it indicates that, for the case study, it is not necessary to consider more complex models of the transportation network, such as the Cell Transmission Model (CTM) of Daganzo [3] which has raised significant interest due to its elegance and practicality in various settings. However, the CTM is a mixed integer formulation and hence it adds additional computational complexity to the models. Some work has considered the linear relaxation of the CTM. However, this linearization makes it possible to delay vehicles everywhere in the network, which is not practical in evacuations and makes the model overly optimistic.

Methods BN and CPG have been shown to be overly optimistic by the microscopic simulations. This finding is important as well. It suggests that flow-based modeling of evacuations will also be significantly optimistic. Indeed, flow-based methods decide the route and timing of each individual evacuee and hence they have much more flexibility than zone-based evacuations. Similarly, methods not based on time-expanded graphs and not reasoning about time are unlikely to be of high fidelity in practice.

Evacuations Under Uncertainty *Moving from deterministic evacuation planning algorithms to optimization under uncertainty is a critical extension to the methods presented in this study.* There are many sources of uncertainty in evacuations, from the natural or human-made disaster itself to evacuee compliance and potential accidents. Large-scale zone-based evacuation planning under uncertainty is largely unexplored. The work of Andreas and Smith [1] is a notable exception. They study a stochastic evacuation planning problems, where scenarios specify the uncertainty (e.g., the loss of an arc or an increased travel time). They assign penalties to each arc in the network and define the cost of a plan as the sum of the penalties on the arcs used in the evacuation. The resulting problem is to find a convergent evacuation plan that minimizes the expected cost. Andreas and Smith apply a Benders decomposition, but their algorithm does not scale beyond a dozen nodes.

It is important to emphasize that, in evacuations, the uncertainty is not only exogenous (e.g., the flood may be more severe than expected). It also depends on the evacuation plan itself. Indeed, the probability of an accident along an arc increases with the flow being scheduled on the arc. This type of endogenous uncertainty is particularly difficult to handle efficiently.

Reliable Evacuation Plans An interesting direction in evacuation planning is to borrow some ideas from power systems engineering and to define various notions of reliability. For instance, the dispatches of transmission systems satisfy a property known as n-1 reliability which means that the network can sustain the loss of a line or a generator and redispatch flows appropriately. Defining and finding n-1 reliable evacuation plans is an interesting avenue for further research. Within this framework, it would be typical to have a master problem to find a robust plan and a subproblem for each contingency. The subproblem will then aim at rerouting the flow of evacuees affected by the contingency without altering the rest of the evacuation significantly.

Evacuating Low-Mobility Populations and Multi-Modal Evacuations The problems studied in this paper assume that each evacuee (or each household) has a vehicle available for evacuation and do not consider low-mobility populations. Bus evacuations (e.g., [2, 4, 7, 11, 14]) have been studied rather extensively for evacuating low-mobility populations. However, it is important to consider the multi-modal evacuations that integrate both vehicle owners and low-mobility populations, since they share the same transportation network and may affect the design of contraflows. On-demand multimodal transit systems

[12] may bring some significant benefits in that context, since these novel transit systems solve the first/last mile problem. They will be able to pick up riders at their location.

Additional Functionalities There are a number of additional and critical issues that also deserve to be mentioned. The problem of reversed evacuations, e.g., bringing residents back to the evacuation area, is also of great importance and needs to take into account various cleaning efforts and priorities. The evacuation process itself should also take into account lodging and other amenities when scheduling an evacuation for a large area that will not be livable for a significant period of time. Taking into account these considerations may significantly reduce the financial burden of many families.

6 Conclusion

This paper presented a systematic comparative study of zone-based evacuation planning, their design choices, their fidelity, and their computational performance. It synthesized existing algorithms, complemented them with some new ones to fill some of the gaps in the design space, and compared them on a real-life case study both at macroscopic and microscopic scales. In particular, the evaluation was performed on the Hawkesbury-Nepean (HN) region located north-west of Sydney, Australia, whose associated time-expanded graph has 30,000 nodes and 75,000 arcs.

In zone-based evacuation planning, the region to evacuate is divided into zones and each zone must be assigned a path to safety and departure times along the path. Zone-based evacuations are highly desirable in practice because they allow emergency services to communicate evacuation orders and to control the evacuation more precisely. This study quantified, for the first time, the benefits and limitations of contraflows, convergent plans, and non-preemption, providing unique perspectives on how to deploy these algorithms in practice. It also highlighted the approaches best suited to capture each of these design features and the computational burden they impose.

A key insight of this study is the success of mathematical programming for designing macroscopic evacuation planning algorithms that are effective when evaluated by microscopic simulators on large-scale scenarios. The algorithms thus provide both primal solutions and a performance guarantee. A second key insight is the importance of constraining the optimization algorithms to produce realistic plans. By imposing the non-preemption constraints, the optimization algorithm produces evacuation plans that are easy to enforce and of high fidelity. By imposing convergence properties, the optimization algorithm avoids congestion and driver hesitations. The optimization algorithms have been shown to be highly practical and some of them can even be used in real-time settings.

This paper also listed a number of directions for future work and perspectives on the field. Perhaps the most pressing issues center around delivering plans that are robust under uncertainty and that support multimodal evacuations of both vehicle owners and low-mobility populations.

Acknowledgements

We would like to express our deep gratitude to Peter Cinque (New South-Wales State Emergency Services) and Peter Liehn (then at NICTA) for their leadership in the NICTA evacuation project, the foundation of this work. Manuel Cebrian, Caroline Even, Victor Pillac, and Andreas Schutt drove the development of many of the algorithms presented here and have been truly amazing collaborators. Finally, we would like to thank the reviewers for their detailed reading of the paper and their numerous suggestions, which improved the paper significantly.

References

[1] A. Andreas and J.C. Smith, *Decomposition algorithms for the design of a nonsimultaneous capacitated evacuation tree network*, Networks **53** (2009), 91–103.

[2] D. Bish, *Planning for a bus-based evacuation*, OR Spectrum **33** (2011), 629–654.

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

[4] T. Dhamala and I. Adhikari, *On evacuation planning optimization problems from transit-based perspective*, Int. J. Oper. Res. **15** (2018), 29–47.

[5] J. Erdmann, *SUMO's Lane-Changing Model* pp. 105–123, Springer International Publishing Cham 2015.

[6] C. Even, A. Schutt, and P. Van Hentenryck, *A constraint programming approach for non-preemptive evacuation scheduling*, 21th International Conference on the Principles and Practice of Constraint Programming (CP-2015), Lecture Notes in Computer Science, Vol. 9255, 2015, pp. 574–591.

[7] M. Goerigk, B. Gruen, and P. Hessler, *Combining bus evacuation with location decisions: A branch-and-price approach*, Transp. Res. Procedia **2** (2014), 783–791.

[8] M.H. Hasan and P. Van Hentenryck, *Large-Scale Zone-based Evacuation Planning – Part I: Models and Algorithms*, Networks (submitted) (2020).

[9] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, *Recent development and applications of SUMO - Simulation of Urban MObility*, Int. J. Advances Syst. Measurements **5** (December 2012), 128–138.

[10] S. Krauß, Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics, Ph.D. thesis, University of Cologne, 1998.

[11] M. Li, J. Xu, L. Wei, X. Jia, and C. Sun, *Modeling a risk-based dynamic bus schedule problem under no-notice evacuation incorporated with dynamics of disaster, supply, and demand conditions*, J. Advanced Transp. **1** (2019).

[12] A. Maheo, P. Kilby, and P. Van Hentenryck, *Benders decomposition for the design of a hub and shuttle public transit system.*, Transp. Sci. **53** (January–February 2019), 77–88.

[13] V. Pillac, P. Van Hentenryck, and C. Even, *A conflict-based path-generation heuristic for evacuation planning*, Transp. Res. Part B **83** (2016), 136–150.

[14] H. Zheng, *Optimization of bus routing strategies for evacuation*, J. Advanced Transp. **48** (2014), 734–749.