Reactive and Proactive Congestion Management for Emergency Building Evacuation

Antoine Desmet and Erol Gelenbe
Department of Electrical and Electronic Engineering
Intelligent Systems and Networks Group
Imperial College London, UK SW7 2AZ
http://sa.ee.ic.ac.uk/

***Abstract*—We introduce two congestion metrics to guide emergency evacuations using sensing and local area networks, that use current congestion and congestion forecasts, together with the Cognitive Packet Network (CPN), a routing algorithm which intelligently discovers paths. Using simulations we find that CPN performs well for emergency evacuation with such metrics, and also reveal the dynamics that these schemes can create.**

I. Introduction

In emergency evacuation and management [1], [2], exit signs in built environments can be replaced by smart dynamic networked signs which are controlled by computerized decisions based on current conditions during an emergency, acquired through a sensor network. Such dynamic sings can optimize the flow of evacuees and ultimately improve the evacuation outcome, as evacuees otherwise tend to exit through what they perceive as the shortest path, which often leads to congestion. While congestion is bound to occur when all building occupants rush towards the exits, this may nevertheless be alleviated by directing some evacuees towards less congested safe exits or paths. Thus we present evacuee routing methods that optimize evacuee flows and minimize evacuation times. Most evacuee routing systems focus on finding and guiding evacuees along the shortest safe egress path. For instance, the concept of *artificial potential fields* [3], [4] supports distributed route-finding and can be combined with wireless sensors. Opportunistic communications [5] offer a robust infrastructure-less method to exchange information on the location and intensity of hazards among evacuees, even when network attacks occur [6], while smart search techniques [7], [8] together with hazard propagation prediction [9] can also be used. Recent work [10] confirmed that the shortest-path approach performs poorly in densely-populated areas. As the search is focused on *the* best solution, users are inherently guided towards the same path. This results in widespread congestion along this shortest route – while other less optimal (yet safe) paths will be idle throughout the evacuation.

A simple way to manage congestion is to incorporate it to the routing algorithm's cost metric. Instead of using distance alone, using the path *traversal time* – including queuing time – allows conventional shortest-path algorithms to solve the flow-optimization problem by modeling the building as a network of queues [11], [12]. However, congestion is a routing-sensitive metric [13] which increases with the probability of routing traffic into the path. Given the presence of a time-delayed feedback between congestion and routing decisions, algorithms searching for the route with the lowest cost will perform poorly, since they do not account for capacity constraints and ignore the fact that routing evacuees through the best route mechanically increases its cost. Transhipment theory [14] also offers flow-oriented solutions to optimize the throughput across capacity-constrained networks, using *all* available routes. Transhipment theory is based on the concept of "flow graphs" which, in its simplest form, represents the available flow capacity at each edge; and is generally used to calculate maximum achievable steady-state flow rates between a set of nodes – regardless of the length of the paths used. In contrast, emergency evacuation problems feature a finite number of evacuees, a strong time-minimization constraint, and often dynamic edge capacities. Thus, the steady-state solution offered by flow graphs is of limited relevance for such a dynamic problem. This steady-state limitation can be overcome by performing a time-expansion of the flow graph [15], where dynamic flows can be represented at each time step. However this time expansion comes with a very high memory and processing cost, since the entire graph is essentially replicated for each time-step. In [16], [17], improvements to the time-expanded graph model are proposed in the form of time-aggregated flow graphs, where the flow rate along each edge is a time series. Their Capacity-Constrained Route Planner (CCRP) determines when a flow reaches any given edge of a path, and preemptively reserves capacity on each edge from the expected arrival time to the expected time of departure. While the complexity of CCRP is lower compared to algorithms based on time-expanded flow graphs [18], the CCRP algorithm requires a Dijkstra shortest-path algorithm to be run at each step, which is a very resource-consuming process. Efficient "self-aware" routing algorithms which are nature-inspired [19] such as the Cognitive Packet Network [20], are therefore considered in this paper to route evacuees.

II. CPN and Congestion-Aware Path Metrics

The Cognitive Packet Network concept aims at solving the problems experienced by large and fast-changing networks, where the convergence time of "overall" routing schemes eventually becomes prohibitive. CPN alleviates this issue by letting each network node send a small flow of "Smart Packets"

(SP) which are dedicated to network condition monitoring and new route discovery. In our implementation, these SPs reside in the application server and behave like *virtual* evacuees, exploring both new and known paths, and gathering network condition information as they progress. Every node in the network hosts a Random Neural Network (RNN) which is used to direct incoming SPs towards their next hop. SPs are also allowed to "drift" away from the RNN's recommendation and explore new paths randomly – in a manner comparable to "ant colony" algorithms. Each time an SP reaches its intended destination, an acknowledgement (ACK) message containing all measurements made by the SP travels back to the source along the original path – with loops removed – and updates all nodes visited with fresh measurements. Every node along the path uses the measurements carried by SP ACKs to adjust their RNN by performing Reinforcement Learning (RL), they also store this information in a table used to source-route the evacuees. This routing table stores a fixed amount of optimal paths discovered by SPs and allows any node to instantly switch to an alternate path if ACKs indicate that the best path is no longer optimal. An overview of CPN performance can be found in [20].

We introduce two congestion-oriented path metrics to use with the CPN routing: (a) *current* congestion values to calculate path traversal times for "reactive" control, and (b) future or "proactive" resource allocation to calculate path traversal times, which allows the routing algorithm to account for future increases in congestion when comparing routes. Both metrics use a graph-based representation, where edges represent paths, and nodes represent physical areas. The graph provides the following information

- Edge distance, so that a transit time T_v can be calculated for each edge v based on the evacuees' average walking speed, and
- Edge's capacity C_v, defined as the number of evacuees which can concurrently travel along an edge.

The reactive path metric is based upon the assumption that the queue levels in the building reach a steady-state value (as defined in the context of queuing networks). Since the queue levels remain stable, the complete path traversal time can be calculated using queueing network theory based on *current* congestion observations. For a given path *p* composed of a collection of edges *V*, the path traversal time *T* is estimated as follows:

$$T_{t,p} = \sum_{V} \frac{(N(v,t)+1)}{C_v} \cdot T_v, \tag{1}$$

where $N(v,t)$ is the number of evacuees queueing in the area at the instant t. The feedback loop between the path metric and path assignment can create path oscillations. These do not concern a given evacuee but distinct evacuees only. The reactive metric requires a sensor system that measures queue length. While vision-based, proximity or presence detection systems can perform this task, the associated deployment costs, and the potential for issues related to robustness and accuracy of the infrastructure are the main drawbacks of this metric.

The proactive metric keeps track of capacity reservations by dividing the evacuation time in N steps of interval $T_{interval}$ and assigning as many "time-bins" $B_n(v)$ to every edge. The time span of a bin $B_n(v).TimeSpan$ is from the instant $B_n(v).startTime = n \times T_{interval}$ up to, but not including, $B_n(v).endTime = (n+1) \times T_{interval}$.

To enforce the edge's maximal flow capacity, time bins have a maximum capacity B_{max_v} (eq.2) which is a "discretization" of the edge's maximal flow ($\frac{1}{T_v} \cdot C_v$) over the time-period $T_{interval}$. Put simply, B_{max_v} is the maximum number of evacuees that can transit through v within $T_{interval}$.

$$B_{max_v} = \frac{T_{interval}}{T_v} \cdot C_v \tag{2}$$

Once an evacuee is assigned a path, the reservation process iterates through every edge along the path, and reserves capacity on the relevant bins. A full bin at the expected time of arrival signifies that other evacuees will already be occupying this edge and that the user will have to wait in line. The algorithm will then search for the earliest bin with spare capacity and reserve a space. The user is not expected to depart the edge earlier than the time associated to the bin he was allocated to (plus his *own* service time). The algorithm is detailed in Algorithm 1. If the algorithm is simply used to assess the traversal time of a path, line 8 (where the flow capacity is reserved) is skipped. The algorithm successively assigns a route to a user, then performs a routing update until every user has a route. No particular order or priority regime is applied when paths are being assigned to evacuees.

III. Simulations

We use the Distributed Building Evacuation Simulator (DBES) [21], a discrete-event simulator to evaluate the effectiveness of both metric and the ability of CPN to route users. At this stage, we assume the evacuees can be individually routed by means of a communication device which displays the path. In later stages of this research project, we intend to adapt the routing algorithm so that path recommendations can be displayed on dynamic exit signs in the building. The routing algorithm's parameters are configured as follows:

Reactive metric: the CPN algorithm allows each node to send a batch of 5 SP every ten seconds throughout the simulation. Evacuees also receive updates every ten seconds, but not all at the same time, as updates are not synchronized across all evacuees.

Proactive metric: CPN allows each node to send one SP each time a path is allocated to an evacuee. The time bins' interval $T_{interval}$ is set to 15 seconds: this corresponds to the time it takes to walk through the longest edges in the building (staircases). This value provides a good time resolution for the main bottleneck, which are also the staircases.

For comparison purposes, we conduct a third set of experiments without evacuee flow optimization, where evacuees

Data: Departure time $T_{departure}$; Path P
Result: Path Traversal Time and Capacity Reservations

$T_{arrival} \leftarrow T_{departure}$;
forall the *Edges* $v \in$ *Path* P **do**
  /* Find earliest free time bin */
  forall the $B_n(v)$ *of edge* v **do**
    Find $\min(n)$ so that:
      $\cdot B_n(v).reservations < B_{max_v}$
      and
      $\cdot B_n(v).TimeSpan$ is on or after $T_{arrival}$
  end
  $B_n(v).reservations \leftarrow B_n(v).reservations + 1$
  if $T_{arrival} < B_n(v).startTime$ **then**
    /* Edge is available at the projected arrival time */
    $T_{arrival} \leftarrow T_{arrival} + T_v$
  else
    /* Evacuee expected to queue until $B_n(v).startTime$ at best */
    $T_{arrival} \leftarrow B_n(v).startTime * T_{arrival} + T_v$
  end
end
return $T_{arrival}$

Algorithm 1: Capacity reservation algorithm

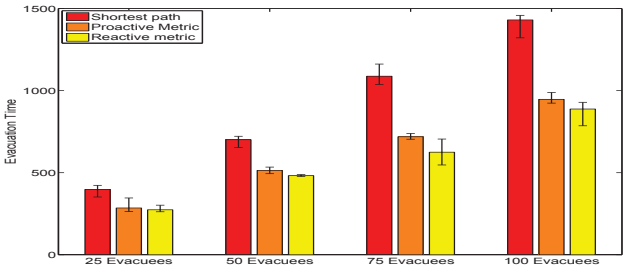


Fig. 1. Total evacuation time, arbitrary time units. The results are the average of 5 randomized simulation runs, and error bars shows the min/max result observed in any of the 5 simulation runs.

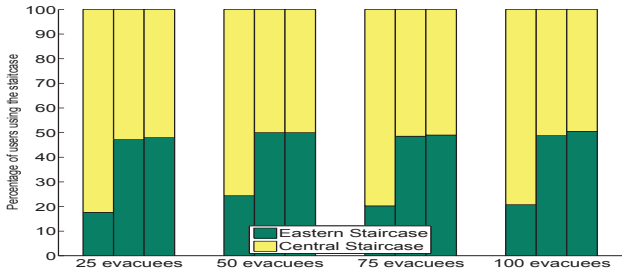


Fig. 2. Distribution of staircases used to reach the ground floor from the first floor. Bar order: [SP Routing | Proactive Metric | Reactive Metric]

exit through the shortest path. The total number of evacuee at the beginning of the simulation is set to {25, 50, 75, 100}. Evacuee's walking speeds are randomized (Gaussian distribution) and we use the distribution's average (1.5m/s) to calculate edge transit times (T_v). The featured graph (Figure 3) represents the three lower floors of Imperial College London's EEE building. Each floor has a surface area of approximately 1000 m^2. The building combines office and classrooms with a large lobby area on the ground floor, where the two exits are located. Overall, the graph consists of approximately 250 nodes and 400 edges. The users' initial locations are also chosen at random at the beginning of each simulation, from a pool of nodes on the middle floor. We purposefully select starting locations which will result in an uneven initial user distribution, to ensure that the flow optimization problem is not trivial. Finally, each configuration is simulated five times to allow for some initial location and evacuee speed variations, and average results are presented.

IV. Discussion and Conclusions

Figure 1 summarizes the total evacuation times of each configuration. A detailed analysis of the simulation results reveals that the largest bottlenecks occur at the staircases – especially those leading to the ground floor – due to their low capacity and longer transit times. The routing algorithm's ability to optimally distribute the flow of evacuees among the staircases (based on their maximum flow and transit time) largely determines how long the evacuation will take. As expected, the shortest path algorithm performs poorly. Because most users are on the southern or western area of the second floor, the central staircase is part of most of the evacuees' shortest egress path, and therefore overused by the SP algorithm (Fig. 4(a)). On the other hand, Figure 2 shows that the Eastern staircase is used only by 20-25% of evacuees. The simulations where CPN is used with congestion-management metrics reduce the building evacuation time by approximately 30%. Figure 2 confirms that both algorithms evenly distribute evacuees through the building's bottlenecks. We conclude that CPN was able to carefully monitor the busiest paths and closely adjust the flow rates to optimize the flows. Figures 5.(b,c) also show that the Smart Packets performed a thorough network exploration, since they discovered a non-trivial path going up though the top-floor to bypass the congestion on the middle floor.

Both metrics achieve comparable evacuation times, with a slight advantage to the reactive metric owing to its high update frequency and its ability to correct in real-time errors which arise from variations in individual walking speeds. However, the evacuation process and dynamics shows significant differences. Since the proactive metric allows source-routing, evacuees follow the same path until they reach an exit. A close inspection of Figure 4(b) reveals that the edges visited form a tree, where evacuees start from their initial location (leaf nodes), and follow the local branches towards one of the two staircases (trunks) and eventually reach the exit (tree's root node). Overall, it appears to be a well-ordered evacuation

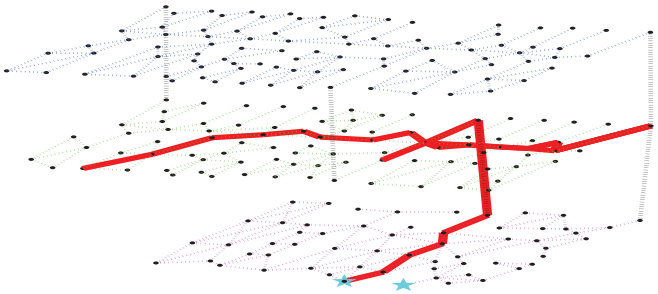


Fig. 3. Trace of one of the evacuees' egress path (highlighted in Red) routed using the reactive metric. The user starts from the path endpoint closest to the left of the illustration. The trace shows that the user has backtracked several times between the two staircases, as a result of the routing algorithm's oscillations, before eventually exiting through the central staircase.

where evacuees join the right queue and wait for their turn to clear the bottleneck. On the other hand, the same Figure for the reactive metric (4(c)) shows a very large number of edges visited "upstream" of the staircases, and in particular in the area located between the top of both staircases. Figure 3 shows a sample trace of the typical motion of reactive-routed evacuees. These oscillations in the routing decisions should therefore be mitigated as in [20] to achieve stable routing decisions.

References

[1] A. Filippoupolitis and E. Gelenbe, "A distributed decision support system for building evacuation," in *Human System Interactions, 2009. HSI'09. 2nd Conference on*, 2009, pp. 323–330.

[2] E. Gelenbe and F.-J. Wu, "Large scale simulation for human evacuation and rescue," *Computers and Mathematics with Applications*, vol. 64, no. 12, pp. 3869–3880, December 2012.

[3] Q. Li, M. D. Rosa, and D. Rus, "Distributed algorithms for guiding navigation across a sensor network," in *ACM Int'l Conf. Mobile Computing and Networking*, 2003, pp. 313–325.

[4] A. Filippoupolitis and E. Gelenbe, "An emergency response system for intelligent buildings," in *Sustainability in Energy and Buildings*, 2011.

[5] E. Gelenbe and G. Görbil, "Opportunistic communications for emergency support systems," *Procedia Computer Science*, vol. 5, pp. 39–47, 2011.

[6] E. Gelenbe and G. Loukas, "A self-aware approach to denial of service defence," *Computer Networks*, vol. 51, no. 5, pp. 1299–1314, 2007.

[7] E. Gelenbe and Y. Cao, "Autonomous search for mines," vol. 108, no. 2, pp. 319–333, 1998.

[8] E. Gelenbe and T. Koçak, "Area-based results for mine detection," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 38, no. 1, pp. 12–24, 2000.

[9] A. Cherniak and V. Zadorozhny, "Towards adaptive sensor data management for distributed fire evacuation infrastructure," in *Int'l Conf. Mobile Data Management*, 2010, pp. 151–156.

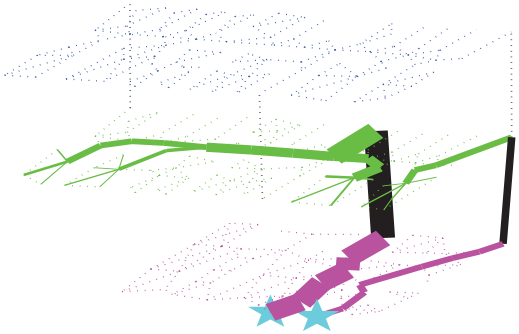
[10] H. Bi, A. Desmet, and E. Gelenbe, "Self-aware networks for emergency evacuation routing," 2013, (To appear).

[11] E. Gelenbe, "Probabilistic models of computer systems. part II: Diffusion approximations, waiting times and batch arrivals," *Acta Informatica*, vol. 12, pp. 285–303, 1979.

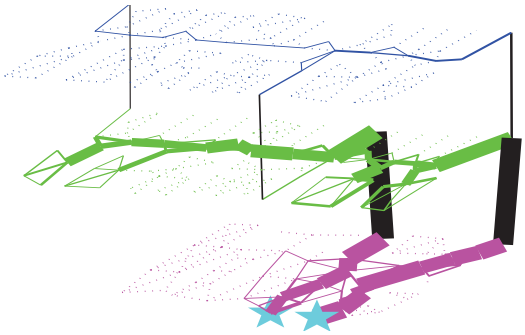
[12] A. Desmet and E. Gelenbe, "Graph and analytical models for emergency evacuation," *Future Internet*, vol. 5, no. 1, pp. 46–55, 2013. [Online]. Available: http://www.mdpi.com/1999-5903/5/1/46

[13] E. Gelenbe, "Sensible decisions based on qos," *Computational management science*, vol. 1, no. 1, pp. 1–14, 2003.

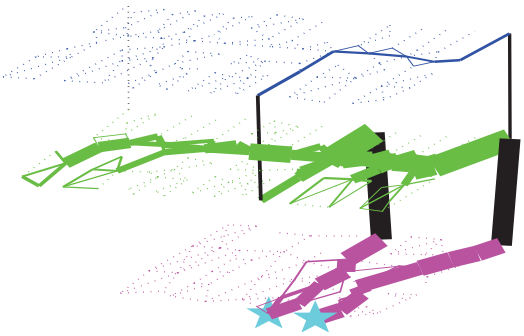
[14] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Courier Dover Publications, 1998.



(a) Shortest-path metric routing



(b) Proactive congestion metric routing



(c) Reactive congestion metric routing

Fig. 4. Edges visited during evacuation featuring 100 building occupants. Line thickness is proportional to the number of visits. To highlight backtracking, we count each successive visit to the same edge by an evacuee.

[15] B. Hoppe and É. Tardos, "The quickest transshipment problem," in *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1995, pp. 512–521.

[16] B. George, S. Shekhar, and S. Kim, "Spatio-temporal network databases and routing algorithms," *Transactions on Knowledge and Data Engineering (TKDE), IEEE,(Submitted in 2008, Also Tech. Report 08-039, Computer Sc., Univ. of Minnesota)*, 2008.

[17] Q. Lu, Y. Huang, and S. Shekhar, "Evacuation planning: A capacity constrained routing approach," in *Intelligence and Security Informatics*. Springer, 2003, pp. 111–125.

[18] Q. Lu, B. George, and S. Shekhar, "Capacity constrained routing algorithms for evacuation planning: A summary of results," in *Advances in spatial and temporal databases*. Springer, 2005, pp. 291–307.

[19] E. Gelenbe, "Natural computation," *Computer J.*, vol. 55, no. 7, pp. 848–851, 2012.

[20] ——, "Steps towards self-aware networks," *Communications of the ACM*, vol. 52, pp. 66–75, 2 2009.

[21] N. Dimakis, A. Filippoupolitis, and E. Gelenbe, "Distributed building evacuation simulator for smart emergency management," *The Computer Journal*, vol. 53, no. 9, pp. 1384–1400, 2010.