Routing Emergency Evacuees with Cognitive Packet Networks

Huibo Bi, Antoine Desmet and Erol Gelenbe *FIEEE*

Imperial College London
Department of Electrical and Electronic Engineering
Intelligent Systems and Networks Group
{huibo.bi12,a.desmet10,e.gelenbe}@imperial.ac.uk

Abstract. Providing optimal and safe routes to evacuees in emergency situations requires fast and adaptive algorithms. The common approaches are often too slow to converge, too complex, or only focus on one aspect of the problem, e.g. finding the shortest path. This paper presents an adaptation of the Cognitive Packet Network (CPN) concept to emergency evacuation problems. Using Neural Networks, CPN is able to rapidly explore a network and allocate overhead in proportion to the perceived likelihood of finding an optimal path there. CPN is also flexible, as it can operate with any user-defined cost function, such as congestion, path length, safety, or even compound metrics. We compare CPN with optimal algorithms such as Dijkstra's Shortest Path using a discrete-event emergency evacuation simulator. Our experiments show that CPN reaches the performance of optimal path-finding algorithms. The resulting side-effect of such smart or optimal algorithms is in the greater congestion that is encountered along the safer paths; therefore we indicate how the quality of service objective used by CPN can also be used to avoid congestion for further improvements in evacuee exit times.

Keywords: Emergency navigation

1 Introduction

Emergency evacuation of large areas or buildings [7, 17, 26] is effectively a complex and transient transshipment problem, from multiple sources to multiple destinations, where edge capacity varies based on the presence and intensity of hazards. Finding dynamic and real-time solutions to this type of problem requires a highly reactive and adaptive system due to the fast-changing nature of the environment [1–4, 22, 23, 29, 27].

Indeed, not only are the threats which trigger the evacuation themselves dynamic, but in addition to this, the flows of evacuees in the building generally tend to be unstable. For instance, the decision to send evacuees down what appears to be the safest path may lead to a sudden increase in congestion, which could ultimately create a deadly stampede situation. On the other hand, a coarse-grained graph representation of a 3-storey building can contain up to

300 nodes and twice as many edges, making it virtually impossible to resolve and monitor *every* possible egress path in the building.

Owing to the size of the building graphs, using an algorithm such as Dijkstra's each time the conditions change can be computationally expensive. Furthermore, these algorithms focus all efforts on finding the *optimal* solution, instead of considering a collection of safe paths for optimal transshipment. Spanning-Tree algorithms [5] support distributed resolution of routes; however they only focus on one single optimal path and the convergence speed may also hinder performance in scenarios where the environment constantly changes. Opportunistic communications can help users self-determine their evacuation path [13, 18], but also suffers from a "hoarding" effect since most users only have access to the same limited amount of information, which leads them to collectively follow what they perceive as being the shortest path. Unlike techniques mentioned previously which search for one optimal path, algorithms inspired from transshipment theory [21, 20] handle capacity-constrained links and maximize the use of *all* available paths. In particular, time-expanded graphs provide exact solutions to transhipment problems, but become prohibitively complex for large graphs, since the complexity not only depends on the scale of the network, but also the "time horizon" which in turn determines the time-expansion of the graph. Finally, the traffic forecasting solution proposed by Lu et al. [24, 25] also finds routes which maximize the flow of evacuees, but the algorithm cannot not handle dynamic hazards: this would modify the availability and capacity of each edge and disrupt the algorithm's forecasted congestion time-series.

We believe that an ideal routing algorithm for evacuation purposes would not only be decentralized, but also able to self-monitor and constantly make swift adjustments instead of requiring a new convergence process or a full graph search each time the environment changes. This algorithm should be able to optimize the evacuation process not only in terms of shortest evacuation path, but perhaps combine other metrics such as safety, congestion, simplicity of the route or more. Such "self-aware" or nature inspired [11] routing algorithms have been developed for computer network packet routing [9] using the Random Neural Network [8] to construct the routing algorithm. The Cognitive Packet Network (CPN) [10] algorithm is an example of an autonomic communication system [6] which is suggested in this paper as a means to route *evacuees* rather than as a means to forward network packets.

CPN is designed to optimize any measurable Quality of Service (QoS) metric such as delay minimisation [15] and energy optimisation [14, 16] using reinforcement learning [19, 12] with a recurrent Random Neural Network [8]. It has been demonstrated against network disruptions such as "Worm-like" network attacks [28], where parts of the network suddenly become unavailable. Similarities exist between computer network packet routing and evacuee routing, in particular, the objectives: to find the best path(s) across a graph with respect to some measurable metric. Evacuees can be seen as data packets; paths and places within the area can be likened to routers and links which also create delays or become unavailable. Thus, in this paper we present an adaptation of the CPN concept to

the emergency evacuation problem, and evaluate its performance by simulating a building evacuation.

In the following section, we first present the core concept of CPN and then discuss how it may be applied to the emergency evacuation problem. The next section presents the simulation model used to evaluate CPN as an evacuee routing algorithm. Finally, simulation results are analyzed, and conclusions are drawn regarding the effectiveness of an evacuee assistance system based on the CPN routing algorithm.

2 The Cognitive Packet Network

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 slower than the rate at which the network conditions change, thus resulting in a constant lag which hinders performance. 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. 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 payload-carrying Dumb Packets (DP). This routing table stores a fixed amount of optimal paths discovered by SPs and allows a node to instantly switch to an alternate path if ACKs indicate that the best path is no longer optimal. Unlike other algorithms which query each node in the network or perform exhaustive graph searches, CPN uses its RNN to intelligently allocate its routing overhead based on how worthwhile a path appears to be: the best path(s) are regularly monitored, while proportionally less bandwidth is allocated to paths which show less potential. A detailed presentation and overview of CPN performance can be found in [10].

Applying the CPN concept to emergency evacuation problems can only be achieved if two main requirements are fulfilled:

- A graph-based representation of the area is available, where edges represent paths, and nodes represent physical areas in the building. This graph contains information on distances, edge and node capacities, and the presence of any relevant *static* hazards, such as stairs, narrow passages, uneven terrain, etc.
- The area to evacuate is covered by a sensor network which monitors any *dynamic* hazard relevant to the scenario. Typical hazards include: fire, smoke, water, congestion, restricted visibility, etc.

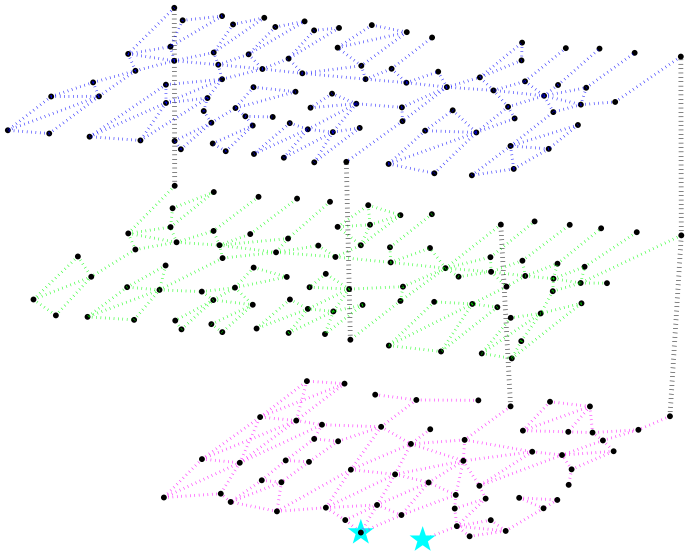

Fig. 1. Graph representation of the building model. The two cyan stars on the ground floor mark the position of the building's exits

In the context of emergency evacuations, the CPN's DPs are effectively the evacuees; while SPs reside in the application's server and "virtually" explore the area, collecting information from the graph and sensors found along the way. This information is then processed into a cost value which is used to compare paths, and CPN ultimately aims at finding the route which exhibits the lowest cost. An advantage of CPN is that its process is independent of the cost function chosen. It can be *any* measurable function, from the shortest and safest path, to functions factoring in the "simplicity" of the path, congestion, or weighed more towards safety, at the cost of increased distance.

3 Simulation Model and Experiment

We use the Distributed Building Evacuatio Simulator (DBES), a Discrete-Event Simulator (DES) to evaluate the effectiveness of CPN for evacuee routing in fire-related emergency evacuations. The area to evacuate is a building, based on the three lower floors of Imperial College London's EEE building. Each floor in this building has a surface area of approximately 1000 m^2. In order to run the CPN algorithm, a coarse graph representation of this building is made (Figure 1).

Effective length is the cost function and compounds path length with hazard intensity along each edge. For a given path p composed of a collection of edges

V, the cost G is:

$$G_{t,p} = \sum_{V} l(v) \cdot f(v,t) \tag{1}$$

Where *l(v)* is the length of the edge, and *f(v,t)* is the fire intensity on the edge v at time t, so that:

$$f(v,t) = \begin{cases} >> \text{ average edge length} & \text{if the edge is affected by fire,} \\ 1 & \text{otherwise.} \end{cases} \tag{2}$$

This ensures that the effective length of a path exposed to fire will always be greater than any other *safe* path in the building.

Our experiments feature three different scenarios for comparison purposes. The first scenario, referred to as "autonomous", simulates cases where evacuees do not receive any assistance: they simply evacuate through the shortest path, and keep trying a different one if they are blocked by the fire – until they either evacuate or perish. The two other scenarios provide evacuees with individual assistance: each time the fire spreads, Dijkstra's Shortest Path algorithm is executed from each node, and determines the *optimal* path, with respect to *effective length*. The third scenario is akin to the previous one, however the decision engine uses CPN. The results of ten randomized simulations are presented in the following section. The fire outbreak is located on the first floor in a position which is not immediately threatening to any user. Yet as the fire expands, it quickly reaches one of the main evacuation pathways and prompts a major re-routing of all evacuees.

4 Results

Before running full-scale randomized simulations, we experimented with the CPN parameters, and in particular with those related to Smart Packets. At the beginning of each simulation, we allow each node to send SPs to provide their RNN with basic training. We noticed that 5 SPs per node is enough for them to resolve at least one egress path. More than 50% of the nodes are able to find the shortest path after 10 SP/node. These are mostly nodes located along the main paths, which take advantage of the numerous SP ACKs transiting through them; while the path length from "leaf" nodes are generally below 120% of the corresponding shortest path. We also found that the Drift Parameter – an SP's probability to choose the next hop at random over the RNN's advice – has an impact on the quality of the routes found by CPN. The first path found by an SP provides the very first reinforcement learning to the RNN's neurons. This path is generally not optimal: the first SPs effectively travel at random since the RNN is not trained yet. With very low drift rates, subsequent SPs will be virtually unable to explore anything beyond this initial path that has set up the RNN's neurons. As a result, the same sub-optimal path is visited over and over and the same neurons are always reinforced, which results in overtraining. While low drift rates let CPN quickly find an exit path for each node, the routing performance soon ceases to increase and may remain far below optimal. On the other

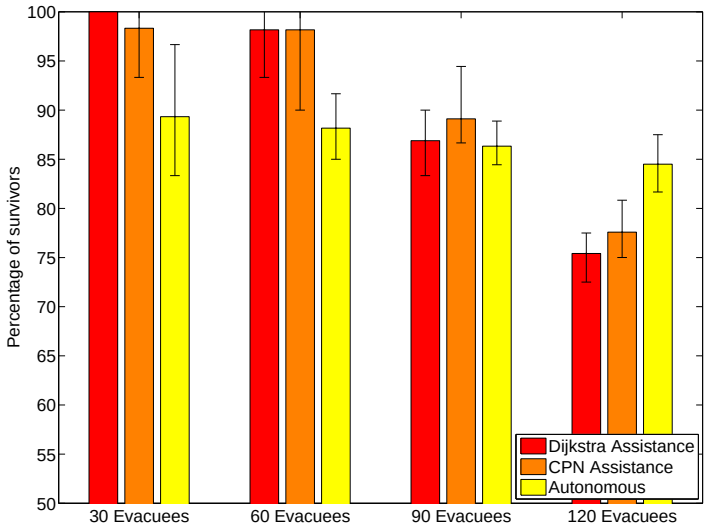

Fig. 2. Percentage of survivors for each scenario. The results are the average of 10 randomized simulation runs, and error bars shows the min/max result in any of the 10 simulation runs.

hand, allowing the SPs to constantly drift leads to very slow set-up times, but all shortest path will eventually be discovered since the SPs effectively perform a random walk.

Figure 2 presents the results of ten randomized experiments. The bars show the percentage of successful evacuees, and the error bars show the single highest and lowest values found in any of the ten iterations. The experiments with lower densities (30 and 60 evacuees) clearly show that CPN reaches the performance level of the Dijkstra's SP algorithm. The results show that approximately 10% of the "autonomous" users die. These evacuees discover that their initial escape path is blocked by the fire, and while they backtrack the fire has time to spread to all exits, effectively trapping them inside the building. On the other hand, evacuees which benefit from the assistance systems receive an early warning that their path is dangerous and re-route early enough to escape the building alive. As the density of evacuees increases (90 – 120 evacuees), the assistance systems progressively lose their advantage, and become clearly detrimental to the evacuees. This is largely due to congestion forming in the building, as can be seen on Table 1. The Dijkstra's algorithm tends to simultaneously send all evacuees through one single optimal path, which creates a synchronized rush and results in the highest levels of congestion. Because CPN is decentralized, each node reacts to the fire and updates its path recommendation at a different time, and this has a tendency to disperse users which in turn slightly reduces the congestion.

Interestingly, the CPN assistance system is able to switch *before* the shortest path becomes affected by fire. This is due to the fact that SPs not only visit the shortest path, but also drift alongside it. Such drifting SPs are exposed to the fire surrounding the shortest path; and since these SPs return with higher cost values, the RL punishes the corresponding neuron. Finally, the scenario where users are left to evacuate on their own exhibits a large "entropy" amongst users, which means that the hallways within the building are less subject to congestion and evacuees can travel faster. Table 1 confirms that congestion plays a critical role – more so than finding the shortest safest path – especially in high density scenarios.

Number of Evacuees	Autonomous Paths	CPN Based Paths	Dijkstra Based Paths
30	3.0	3.5	3.5
60	10.3	12.4	13.0
90	18.6	21	22.3
120	27.8	30.2	31.7

Table 1. Average number of nodes traversed by the evacuees that were seen to have a *non-zero queueing time that indicates some level of congestion*, for different path finding algorithms. We see (from left to right) that as the path finding algorithm gets closer to the optimal, the level of congestion increases since more of the evacuees follow the same best evacuation paths. This illustrates the side effect of using path finding algorithms that find those paths that are safer or easier to traverse during an evacuation.

5 Conclusions

In this paper we have proposed an adaptation of CPN, a Self-Aware computer network routing protocol, to the problem of emergency building evacuation. We have proven that CPN is able to reach performance levels which are on par with those of *optimal* algorithms. Simulations also revealed that using the combined safety and length of an egress path as metric is ineffective in high-density scenarios. In these cases, it appears that reducing congestion and distributing the flow along every available path (even those which constitute a detour) is a more effective strategy. Future research will be directed in this area: we will define a new routing metric that represents a path's travel time, taking congestion into account. While we have proved the effectiveness of CPN in this paper, our next step in this research project will be to assess the complexity of CPN against other algorithms. Of particular interest is the *time-to-live* and quantity of SP required to reach the performance levels of algorithms which perform full graph searches at each step, since these two parameters widely contribute to CPN's complexity.

References

1. Chen, D., Mohan, C.K., Mehrotra, K.G., Varshney, P.K.: Distributed in-network path planning for sensor network navigation in dynamic hazardous environments. Wireless Comm. and Mobile Computing (2010)
2. Chen, P.Y., Chen, W.T., Shen, Y.T.: A distributed area-based guiding navigation protocol for wireless sensor networks. In: IEEE Int'l Conf. Parallel and Distributed Systems. pp. 647–654 (2008)
3. Chen, P.Y., Kao, Z.F., Chen, W.T., Lin, C.H.: A distributed flow-based guiding navigation protocol in wireless sensor networks. In: Int'l Conf. Parallel Processing. p. to appear (2011)
4. Chen, W.T., Chen, P.Y., Wu, C.H., Huang, C.F.: A load-balanced guiding navigation protocol in wireless sensor networks. In: IEEE Global Telecomm. Conf. pp. 1–6 (2008)
5. Dimakis, N., Filippoupolitis, A., Gelenbe, E.: Distributed building evacuation simulator for smart emergency management. The Computer Journal 53(9), 1384–1400 (2010)
6. Dobson, S., Denazis, S., Fernández, A., Gaiti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., Zambonelli, F.: A survey of autonomic communications. ACM Trans. Auton. Adapt. Syst. 1(2), 223–259 (2006)
7. Fischer, C., Gellersen, H.: Location and navigation support for emergency responders: A survey. IEEE Pervasive Computing 9(1), 38–47 (2009)
8. Gelenbe, E.: Learning in the recurrent random neural network. Neural Computation 5(1), 154–164 (1993)
9. Gelenbe, E.: Cognitive packet network. U.S. Patent 6,804,201 (October 11 2004)
10. Gelenbe, E.: Steps towards self-aware networks. Communications of the ACM 52, 66–75 (2 2009)
11. Gelenbe, E.: Natural computation. Computer J. 55(7), 848–851 (2012)
12. Gelenbe, E., Şeref, E., Xu, Z.: Simulation with learning agents. Proceedings of the IEEE 89(2), 148–157 (2001)
13. Gelenbe, E., Görbil, G.: Opportunistic communications for emergency support systems. Procedia Computer Science 5, 39–47 (2011)
14. Gelenbe, E., Lent, R.: Power-aware ad hoc cognitive packet networks. Ad Hoc Networks 2(3), 205–216 (2004)
15. Gelenbe, E., Lent, R., Nunez, A.: Self-aware networks and qos. Proceedings of the IEEE 92(9), 1478–1489 (2004)
16. Gelenbe, E., Morfopoulou, C.: A framework for energy aware routing in packet networks. Accepted for publication in The Computer Journal 54(6), 850–859 (first published online: December 15, 2010)
17. Gelenbe, E., Wu, F.J.: Large scale simulation for human evacuation and rescue. Computers and Mathematics with Applications 64(12), 3869–3880 (December 2012)
18. Gorbil, G., Filippoupolitis, A., Gelenbe, E.: Intelligent navigation systems for building evacuation. Computer and Information Sciences, Lecture Notes in Electrical Engineering (2011 (to appear))
19. Halici, U.: Reinforcement learning with internal expectation for the random neural network. European Journal of Operational Research 126(2), 288–307 (2000)
20. Hamacher, H.W., Tjandra, S.A.: Mathematical modelling of evacuation problems–a state of the art. Pedestrian and evacuation dynamics 2002, 227–266 (2002)

21. Hoppe, B., Tardos, É.: The quickest transshipment problem. In: Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms. pp. 512–521. Society for Industrial and Applied Mathematics (1995)
22. Li, M., Liu, Y., Wang, J., Yang, Z.: Sensor network navigation without locations. In: IEEE INFOCOM. pp. 2419–2427 (2009)
23. Li, Q., Rosa, M.D., Rus, D.: Distributed algorithms for guiding navigation across a sensor network. In: ACM Int'l Conf. Mobile Computing and Networking. pp. 313–325 (2003)
24. Lu, Q., George, B., Shekhar, S.: Capacity constrained routing algorithms for evacuation planning: A summary of results. In: Bauzer Medeiros, C., Egenhofer, M., Bertino, E. (eds.) Advances in Spatial and Temporal Databases, Lecture Notes in Computer Science, vol. 3633, pp. 291–307. Springer Berlin Heidelberg (2005), http://dx.doi.org/10.1007/11535331_17
25. Lu, Q., Huang, Y., Shekhar, S.: Evacuation planning: A capacity constrained routing approach. In: Intelligence and Security Informatics, pp. 111–125. Springer (2003)
26. Malan, D.J., Fulford-Jones, T.R., Nawoj, A., Clavel, A., Shnayder, V., Mainland, G., Welsh, M., Moulton, S.: Sensor networks for emergency response: Challenges and opportunities. IEEE Pervasive Computing 3(4), 16–23 (2004)
27. Pan, M.S., Tsai, C.H., Tseng, Y.C.: Emergency guiding and monitoring applications in indoor 3D environments by wireless sensor networks. Int'l J. Sensor Networks 1(1/2), 2–10 (2006)
28. Sakellari, G., Gelenbe, E.: Demonstrating cognitive packet network resilience to worm attacks. In: Proceedings of the 17th ACM conference on Computer and communications security. pp. 636–638. ACM (2010)
29. Tseng, Y.C., Pan, M.S., Tsai, Y.Y.: Wireless sensor networks for emergency navigation. IEEE Computer 39(7), 55–62 (2006)