Application of Microsimulation to the Modelling of Epidemics and Terrorist Attacks

Ian Piper and Daniel Keep
School of Computer Science and Software Engineering
University of Wollongong, Australia
E-mail: ian@uow.edu.au

Tony Green and Ivy Zhang
School of Risk and Safety Science
University of New South Wales, Australia
E-mail: a.green@unsw.edu.au

Keywords: modelling, microsimulation, terrorism, epidemiology

Received: August 10, 2009

In this paper, a novel approach to behavioural modelling is presented with reference to biological infection spread in communities. Its potential application to various terrorist-related scenarios is discussed in relation to attack point simulation and interdiction simulation.

Povzetek: Predstavljen je nov način modeliranja bioloških infekcij predvsem v zvezi s terorizmom.

1 Introduction and background

The motivation of this project is to develop a computer model which allows the modelling of human response to a variety of human threat scenarios.

These may range across both natural threats such as disease, fire and flood and unnatural events including accident and deliberate acts such as terrorist attacks.

The tool is being developed in response to a number of requests from third parties, including both govenment and commercial bodies, to address a wide range of roles.

Forensic: can we gain an understanding of events that have already taken place?

Speculative: can we develop a reasonable "what if" model for potential threats?

Decision Support: can we dynamically model and predict the course of an evolving threat?

Training: can we develop realistic models for the training of response personnel?

Evaluative: can we test the validity of assumptions made in other modelling techniques?

Terrorist attacks can take many different forms using a wide range of weapon types and this poses some considerable problems in defining both the risks that arise in modern urban environments as well as optimising interdiction and response strategies. Critical infrastructure is dispersed and provides multiple vulnerabilities and opportunities for attack. As attacks both in London and, more recently, in Mumbai have shown, modern terrorism should be thought of as requiring a 3D spatial vulnerability approach to counter terrorism(7). While both game theory and queuing theory have been used for assessing terrorism events we present an alternative approach to this problem which overcomes their fundamental problem, that of estimating the number that get through to attack.

Microsimulation is a discrete simulation technique which allows for the modelling of the behaviour of single individuals in a complex system (4)(13). It was originally devised for financial and economic modelling (17)(15), but is generally applicable to a wide range of scenarios.

In the current research project, we have created a modular, scalable microsimulation package, called Simulacron, which allows for the rapid creation of microsimulations involving large numbers of people interacting with each other and their environment.

The framework is designed to be scalable and distributable, implementing all interactions in terms of distinct locations and individuals. The state of these locations and individuals is flexible and can be arbitrarily extended. Model behaviour is broken into individual modules which can be combined as needed.

In addition to this simulation framework, we have also developed a number of support tools including a prototype non-linear visualisation package which allows for the creation of complex visualisations by non-programming personnel.

Models are specified using an XML dialect. The creation of very large and complex models is achieved through the use of a preprocessor which allows the instantiation of generalised templates into concrete data sets.

The initial study undertaken was the simulation of an

influenza epidemic (8) at the Royal Naval School (RNS) in Greenwich, London in 1920. It was of particular interest due to the relatively complete information available regarding the outbreak, including the progress of the disease over time and its infection mechanism (6) as well as the behaviour of the population in its "normal" state (14). This scenario was also attractive in that it occurred in an essentially closed community for which we have detailed historical documentation (2).

The model of the RNS outbreak involved the creation of nine dormitories, nine reading rooms, around 40 classrooms and roughly 15 other locations including the hospital. Student behaviours were established in seasonal class groups involving a total of 951 students. In addition, the behaviours of 27 staff were also modelled. Each of these individuals is assigned a unique set of infectious parameters based on statistical distributions shared between all the participants. When the simulation starts, each person is sent to their appropriate location based on their schedule. The movements and interactions, including cross infection, of the individuals is then modelled over time by Simulacron.

Behaviours are a mixture of constrained (students must follow their timetables, sleep in assigned dormitories, etc.) and unconstrained (students move about freely at playtime). As stated previously, the modular nature of the framework allows us to use whatever behaviours are most appropriate for the model at hand.

The underlying infection model used in the simulation is conceptually a combination of the "Susceptible, Infective, Recovered, Susceptible" (11) and "Susceptible, Exposed, Infective, Recovered" (5) models (commonly known as SIRS and SEIR respectively). In addition to the standard states, we added two mechanisms to the model. The first, "hero" time, allows the simulation of the "I'm too busy to get sick," or "It's just a little cold," phenomenon. The second, isolation, allows for individuals to be removed from the cross-infection domain once disease is detected. The isolation mechanic can be conditionally applied only during particular hours of the day. The parameters are effectively selected by a convergence methodology against the historical data. While the full procedure is still being developed, the results in the case study presented agree reasonably well with other influenza studies (9).

2 Methodology

The simulation environment currently consists of a number of programs: Simulacron, the simulator, and DSTP, the template preprocessor. Planned future additions include Jazz, a visualisation engine and Refinery, an automated parameter estimation system.

The following sections examine these in more detail.

2.1 Simulacron

Early on it was decided to divorce the development of the simulation models themselves from the surrounding support code. This led to the creation of a general-purpose microsimulation framework called Simulacron. On to this scaffolding, special-purpose simulation modules can be attached to craft purpose-built simulation environments. The current module set is discussed later in this document in sections 2.2, 2.3 and 2.4.

The framework has been written using the D Programming Language. This choice was made because it provided a number of advantages over more traditional choices such as C, C++ or Java. Since it compiles to native machine code, it has the performance advantages of C and C++. It also directly incorporates automatic memory management similar to that of Java. Distinct from the above, D's advanced template support has allowed for the creation of complex, repetitive code to be automated within the language including serialisation and XML parsing libraries.

We have also utilised a customised version of Don Knuth's literate programming environment (12) modified to process D instead of C.

The framework was specifically designed to support very large simulations, leading to the adoption of a master/slave architecture, in which the processing of the simulation can be arbitrarily divided among one or more worker (slave) processes under the supervision of the master process. Each of these processes may operate in a loosely-coupled environment such as that provided by a cluster or network of machines. This architecture allows for the development of models of far greater size and complexity than could reasonably be supported by a monolithic, single-process design.

To date, it has been possible to operate with a single slave running on the local machine. This has permitted models with tens of thousands of individuals.

The only requirement that the framework imposes upon the model is that it must be expressible using a combination of distinct locations (called Cells) and individuals (called Peeps[1]). All interactions and behaviours of these are specified by the simulation modules. Due to its modular nature, the framework itself places no requirements for any particular state information for locations or individuals; their state may be arbitrarily extended by each module, by attaching fields to cells and peeps, to allow for composition of model components into arbitrarily complex simulations.

Time in Simulacron is managed via discrete "ticks", time intervals chosen to be small enough to capture important details in the model but large enough to render large simulations practical. The master is responsible for coordinating the slave processes via issuing tick commands. This mechanism was chosen in preference to the alternative, queueing theory, approach (with free-running slaves) as this was markedly simpler to implement given that there is a poten-

[1]Note that these are unrelated to the popular American marshmallow confection of the same name.

tially unbounded number of future events for every individual object in the simulation at any given time.

It is also worth noting that using queuing theory would make it far more difficult to distribute processing as any event has the potential to affect the state of any object in the simulation. This means that no computational node can proceed *other than* the node which has the next event, thus making distribution of the work pointless.

In addition, the framework allows for the definition of multiple states, each referred to by name. Each cell and peep is in one specific state at any given time, independent of the state of other objects in the simulation. Cells and peeps may have their current state changed by modules at any time.

Input to the simulation is via an XML dialect. Although not our first choice, XML has proven to be a good fit to our needs; the extensible nature of the framework demanding the ability to represent structured data of arbitrary complexity, a task well suited to XML. However, this has proven to be prohibitive in terms of defining large data sets, a problem which is in part resolved by the template preprocessing program DSTP discussed in section 2.5.

Output from the simulator can be encoded in a number of formats. Initially, XML reports were used due to the availability of third party tools which could read and process these reports including web browsers such as Internet Explorer and Firefox, which acted as rudimentary viewers, and Excel, which allowed for more involved analysis.

However, these reports had an undesirable overhead both in terms of disk space and processing time. This has been resolved by moving to a more compact and more easily processed database format based on the SQLite engine.

2.2 Movement modules

In itself, Simulacron does not prescribe any behaviours for either locations of individuals. One basic requirement for modelling people is movement. To this end, there are two modules which provide for different aspects of simulated motion.

The first of these is the scheduling module. This allows for each peep to have one cyclic schedule defined per state for them. These schedules are sequences of time and action pairs. When the trigger time is reached, the defined action is performed. Actions can include moving the peep to another cell and changing their state (both deterministically and at random).

For example, one could use a cyclic schedule to define a full week's worth of movement; going to work on weekdays and engaging in leisure activities on the weekends. Alternately, the same could be done by defining a one-day "workday" schedule, a number of one-day "weekend" schedules and selecting between them via the state mechanism.

There is also a similar scheduling mechanism designed for "one-off" events. Whilst the cyclic schedules define the time for events relative to the start of the cycle, the "one-off" schedule defines the time for events as an absolute date and time.

The second movement module is the dispersal module which, in contrast to the scheduling module, is applied to cells. Simply put, at each simulated tick, the cell will disperse each peep currently located within it to a cell randomly selected from a defined set. The destination cell may, in turn, also be associated with a dispersal set. This is used to simulate, for example, children moving about in a playground or movement of people within a large office building.

2.3 Infection Module

This, the first specialised module to be developed, supports the modelling of a single infectious process. In its simplest form it allows the injection of one or more infected peeps into a population of susceptibles. The infection can then be spread between peeps within the same cell.

The infection model, as stated earlier, is a modified combination of traditional models. The infection progresses through several states: susceptible, latent infected, asymptomatic infective, "heroic" infective and symptomatic infective; following this, the peep will either recover (and possibly become immune) or die.

Each of these is present in various existing models (such as those in (11) and (5)), with the exception of "heroic" infective. This state was originally introduced to model people's tendency to shrug off or hide sickness. It is implemented as a linearly increasing chance of "detection" starting at 0% and ending at 100%, at which point the peep becomes symptomatic infective.

This also plays into another feature of our model: isolation. In its most basic form, this causes symptomatic infective peeps to be forcibly sent to an isolation cell until they either recover or die. The heroic infective state interacts with this to produce an increasing chance of a peep being noticed and sent to isolation as the infection progresses.

Isolation can be configured to only be in effect during particular times of the day. This was used in our RNS simulation to limit isolation to daylight hours; people are unlikely to be isolated at night.

Cross infection is possible at every simulated tick. For each susceptible peep in a cell, a random number in $[0, 1)$ is generated which is then compared to the following:

$$\frac{P(\text{Infection}) \times \text{Time step}}{\text{Average infectious time}} \times N$$

where $P(\text{Infection})$ is the peep's chance of being infected given continuous exposure over "Average infectious time" (a parameter which is specified globally), "Time step" is the amount of simulated time that has passed since the last tick and N is the number of infectious peeps in the cell.

An infection may also be spread via the environment itself; cells can be "infected" at which point they can transmit the infection to peeps passing through them in the same manner. This could be used, as an example, to represent

an air conditioning system for modelling the spread of Legionella.

It is worth noting that not only is the progression of the infection recorded via a field attached to each peep, but the various parameters that control the infection are as well. This means that every peep in the simulation can have unique infection parameters, which includes the duration of the various stages of the infection, their chance to contract the infection and their chance to recover.

A recent addition was the implementation of "infection masking," a process whereby each peep can be placed into zero or more groups, with membership managed via a bitmap mask. The groups a peep is capable of infecting can then be restricted to a possibly different set of groups. This allows for the representation of complex multi-vector diseases such as avian influenza.

It is interesting to note that there is nothing in either this module or Simulacron itself that limits the module to modelling the spread of an infectious disease among humans. Peeps can be used to represent anything which is capable of spreading or merely contracting an infection: pets, wild animals, birds and even particulate matter or molecules of an airborne pathogen. Similarly, the module can be used to model any process which shares similar viral propagation. This includes things such as memes or even a terrorist recruiter (i.e. infective) subverting dissatisfied (i.e. susceptible) individuals.

2.4 Terrorism module

A recent development, the "TPC" module was created to model simple terrorist scenarios. It allows one to divide all peeps into one of three categories: terrorists, police and civilians; hence the name. Terrorists are modelled as having three important parameters; the first is the "camouflage factor," which represents how adept that individual is at hiding themselves. The higher the factor, the lower their chance of being "detected."

Secondly, terrorists have an attack time; presently, the model only deals with terrorists engaging in suicide attacks. At the appointed time, the terrorist will explode, killing all peeps within the same cell. Finally, terrorists also have a configurable chance of prematurely detonating themselves if they are detected by police.

Police have only one parameter: their perception factor, which represents their capacity for spotting terrorists[2]. Thus, a terrorist's chance of being detected by any given police officer is $F_p(1 - F_c)$ where F_p is the police officer's perception factor and F_c is the terrorists' camouflage factor.[3]

Citizens have no parameters; their only function in the model at present is to serve as cannon-fodder. This is not as needlessly malicious as it may first appear: given a community derived from real-world census data, they can serve to determine when and where an attack is likely to occur. They also provide meaningful data in the event of a premature detonation.

Note that this model does not cover the motions of any of the peeps involved; this is provided by the movement modules.

The precise nature of the terrorist threat may be adjusted to meet the needs of a specific scenario. These may include the following:

Instantaneous lethality: e.g. the detonation of a bomb with a substantial payload.

Delayed lethality: e.g. the release of some chemical or radiological agent which is not instantaneously lethal.

Probabilistic lethality: e.g. the detonation of a low yield or less reliable explosive device.

Infection: e.g. the release of an infectious agent or the terrorist deliberately infecting himself.

2.5 DSTP

Models are specified using a template language, based on XML, which permits the creation of very large data sets via the instantiation of relatively simple macro templates. For example, this allows the specification of a single template for a statistically average individual which can then be instantiated an arbitrary number of times to create a background population into which unique individuals may be "injected" to simulate specific behaviours.

Note that although specifically designed to aid in the creation of Simulacron data sets, DSTP itself is not limited to generating them. It could conceivably be used to output any XML-based format. DSTP also supports self-recursive behaviour where the output of a template may be another template; an integer suffix can be added to filenames which the processor will then decrement when naming the output file.

Like the data set format, the template language is based on XML. It resembles a relatively simple functional programming language with dynamic scope. Its standard constructs are concerned with the definition and substitution/instantiation of variables and templates. It also contains a few primitives for the random sampling of values from normal and uniform distributions; these can be used anywhere a value is expected, allowing an arbitrary mixing of randomly sampled and fixed values.

The language currently lacks constructs such as conditionals, arbitrary looping and arithmetic. Given the complexity of templates already possible with the language, we view this lack as a blessing.[4]

Another key construct is the ability to instantiate a template multiple times with a single instruction. Templates

[2]There is currently no provision for "false positives."

[3]This is normalised to be the probability of detection over a one-minute time span.

[4]The authors take the position that if the template language ever becomes Turing-complete, it would likely be advisable to simply switch to using LISP instead.

can be expanded a specified number of times, for each value in an integral range or for each value in a list of words. It is using this, in concert with the random sampling constructs, that allows for a statistically "generic" person to be defined and then instantiated into a concrete population.[5]

To aid in the construction of more complex data sets, DSTP supports a module system that allows additional constructs to be added without requiring change to the basic processor. These range from very simple automation (such as generating interlinked webs of dispersion cells) to much more complex processes.

One such complex process is the creation of communities: given a set of parameters (such as number of adults, number of children, number of employed, number of houses, etc.) derived from census data, it will attempt to create a community that matches those requirements. This includes automatic creation of families and homes.

Another example is the "office builder" which, given some basic properties such as number of floors and offices, will generate a complete, interlinked office building complete with lifts and street access.

In addition to plug-in modules, common components or even complete templates can be abstracted into libraries via the import mechanism to aid in reuse.

3 Preliminary results

3.1 Infection

The original 1920 outbreak lasted for roughly 25 days. Each simulation was run over 30 virtual days with a time step of five minutes and output every hour. The output represents a snapshot of the entire population, recording the location and infection state for each individual. For infected individuals it also records the source of the infection, and when they became infected. This data allows comparison with the historical number of new cases every day.

Systematic changes to the model parameters in successive simulations allowed investigation of the sensitivity of parameters against the historical data. Furthermore, by varying only the initial random seed, different instances of the same underlying process can be modelled, allowing the determination of statistical parameters, such as mean and standard deviation for such properties as number of deaths.

The infection model is not a standard compartmental SEIRS type model and is more readily suited to backcasting and forecasting methods required for decision support in live situations. Furthermore, because it simulates actions of the individual it can test alternative policies and social controls that are difficult, if not impossible, to test without making some gross assumptions. Because of these attributes, it can be used to test assumptions, made in other techniques, that are not ordinarily testable.

[5]The same can be said of generic locations.

Our technique also allows the investigation of additional properties of the infection process which would be equally difficult to evaluate with traditional statistical modelling. Chief among these is the ability provided to determine "infection chains", a chronological sequence of who infected who, where and when, essential in tracing the exact progression of a disease and identifying critical peeps and cells.

The results, even allowing for the relative simplicity of the model and the inevitable inexactitude of the parameter estimates, showed remarkably close agreement with the historical data, as seen in Figure 1 below, capturing the development, peak and recovery times with surprising accuracy. Contrast this with the result of the more traditional purely statistical model shown in Figure 2.[6]

An unexpected outcome was the presence, among the simulation series, of runs in which, despite all parameters being the same, no epidemic occurred. This suggests that further investigation may be required into the "known" causes of epidemic spread.

3.2 Terrorism

The studies we are currently conducting are more directly of interest to the present audience as they include models related to terrorist activity. Two of these we briefly describe below.

In support of this, a more comprehensive model has been developed involving 14402 individuals, grouped into families, whose overall properties match the statistical census data for Australia, interacting in an environment of 6935 locations including 6000 homes and 935 workplaces, schools, recreational areas, hospitals, etc. The people move within the community according to schedules that emulate employed, part-time employed, unemployed and home workers, primary and secondary school children and infants that again match statistical data for Australian communities.

3.2.1 Attack point simulation—the "living bomb"

The first proposed model addresses the question "What happens if we vary the point of release chosen by a terrorist conducting a biological attack?"

The required model behaviour for this scenario can be achieved, without change to the previously described infection model, by the injection of a "living bomb" represented as a single individual who remains stationary for the duration of the simulation and becomes highly infectious at a predetermined moment in time.

Four locations in a virtual community were used as a preliminary assessment of the impact of location on release from a "living bomb". The four locations were a cinema complex, a club, a large store, and the community hospital.

[6]It could be argued that the statistical model is, in a sense, more accurate than the simulation results as it provides a precise match for the total number of cases. However, the simulation method clearly provides a much more realistic, if less "exact", result.

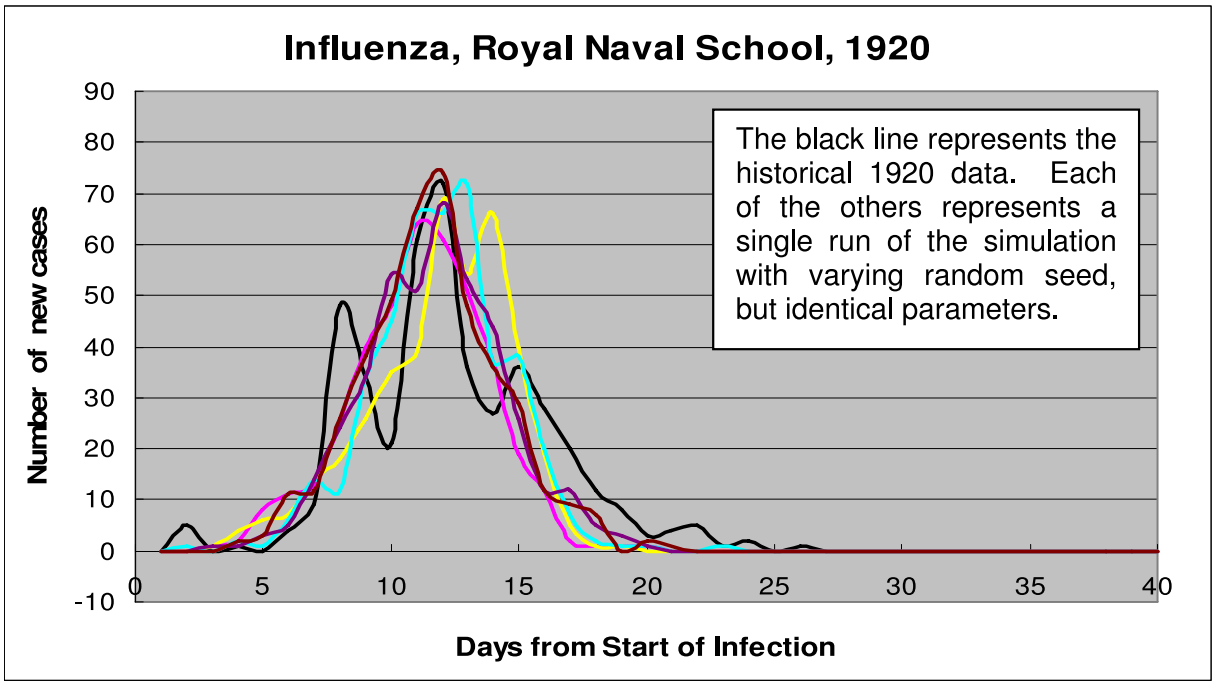

Figure 1: Comparison of simulated runs with historical data

The infection that was used in the simulation was based on smallpox with typical time parameters given by CDC information (3). The probability of infection for each individual was the reproduction rate for an infected person adjusted for the timestep used in the simulation. The initial release emulated a badly constructed device with limited ability to spread infection.

Figure 3 shows the cumulative number of people who visited the four locations. Only two of the simulations resulted in infection spread, the club and the cinema complex with one infection each over the release time. The infection occurred about 10 hours after the release. The pathogen in the absence of the human host was assumed to be viable for approximately 24 hours. The time of first infection is shown in Figure 3. It suggests that in any release there is a threshold of exposure required to spread infection; this can be the number of people or the time exposed. The store and the hospital did not have enough people moving through the building to make it likely that someone contracted the disease.

In the two cases of disease spread, the disease continued to spread through the community. The first appearance of symptoms occurred 14 days after exposure in the cinema and 18 days from the club. By 80 days, 2560 and 2016 people were infected from the cinema and club exposures respectively. With this particular disease, the relatively long incubation period does allow time for intervention, isolation and ring vaccination so long as surveillance systems for the disease identify a case quickly. The second infected person in each simulation became symptomatic 26 and 31 days after the primary exposure by which time there were 4 and 5 additional infecteds.

These early results suggest that the location of release will be extremely important to the number of subsequent cases of infection that occur. While more studies are required to elucidate the sensitivity to population moving through a target and dispersal effectiveness at the point of delivery, the result does have implications for assessment of risk, the provision of resources for dealing with an outbreak and the effectiveness of possible control mechanisms.

3.2.2 Interdiction simulation

The second proposed model addresses the question "How effective is a specific interdiction regime?"

This basic model may be varied by changing properties in a logical manner. For example, replacing the instantaneous lethality of the terrorist attack with a probabilistic one (a smaller bomb) or replacing it with a conventional infective state simulating the release of a biological agent.

Because of the flexibility of the program, police behaviours may range from completely random to precisely specified, the latter allowing the investigation and validation of predetermined interdiction strategies such as those derived from game-theoretic modelling (16).

The community used in the above "living bomb" scenario was used with the club as a target location for a terrorist attack. The terrorist was embedded in the community undertaking normal activities arriving at the point of explosion just before the time the explosion is due. Police also moving about the community are attempting to stop the attack. In the nine scenarios tested the three factors that are used in the simulation are shown in Table 1. There was one

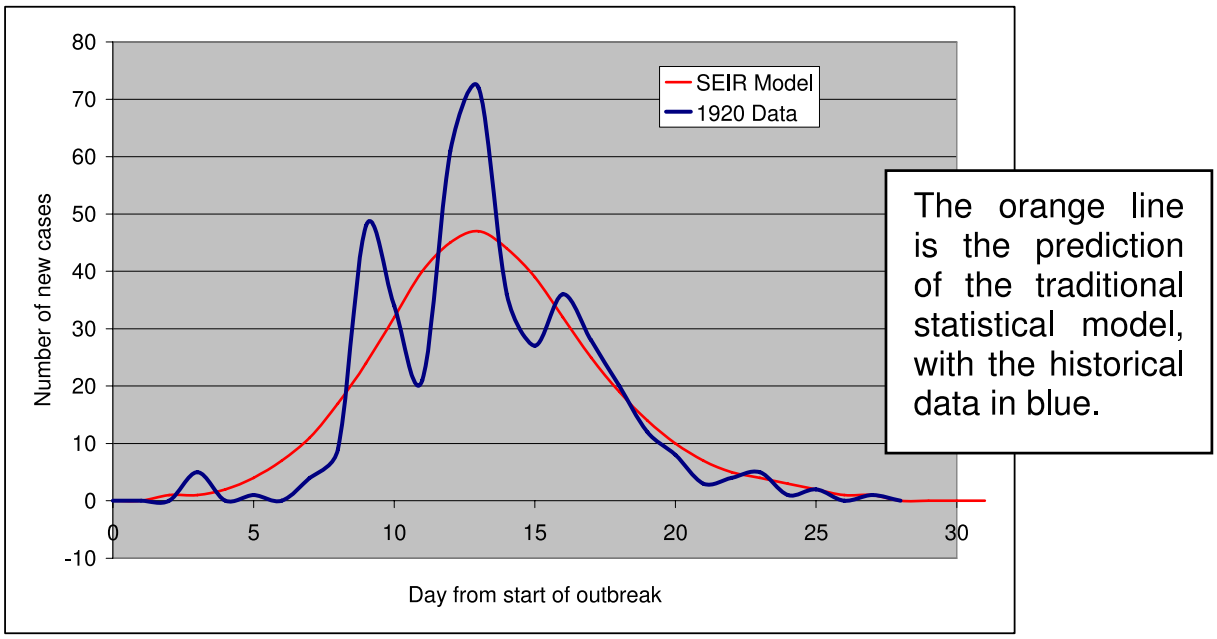

Figure 2: Comparison of SEIR model with historical data

terrorist and ten police in each of these simulations. Figures 4 and 5 show the results of the simulation; the top shows the interception factor as a function of the date and time while the bottom graph shows the location and the number of dead or injured as a function of date and time. The only simulation to get to the target time of 03 Jun 21:23 was run 8. Pre-emptive detonation only occurred in run 0. The other 6 simulations resulted in successful arrest. The decline of the interception factor (top) follows the transition from early to later times in interception eventually resulting in no interdiction and the terrorist reaching the target at the designated time.

These two examples show the potential power in this type of modelling as it can allow the investigation not only of complex attacks but also the requirements for resources, the levels of perception or intelligence assistance and the tactics that are needed to optimise these resources if these types of activities are to be prevented.

Our method differs significantly from other methods for assessing interdiction strategies (1)(10) as it is time rather than event driven and based on the detailed modelling of individual behaviour within population groups rather than more abstract constructs.

Because of this, we can "inject" deterministic behaviour patterns for specific individuals into a background population modelled with randomly varying properties.

The advantage as we see it over existing techniques is that it can produce in simulations both the successes and failures together with full information about the paths to success or failure.

4 Conclusion and future work

A number of improvements to the array of available simulation modules are presently being planned. These include a transportation system including both personal and public transport, the addition of multiple simultaneous infections, chokepoints and other rate-limiting devices and a "baggage" system to assist in accurately modelling airports.

A number of new applications of Simulacron are currently in planning, including the simulation of a large transportation facility and the development of a more comprehensive end-to-end terrorism model including the training of agents, the planning, preparation and conduct of attacks and the subsequent response.

From its inception, the simulation system was intended to be part of an integrated risk modelling and assessment software environment. To this end, it is intended to integrate the Simulacron package with a more fully-realised version of the visualisation environment, allowing a bi-directional real-time flow of data between them.

This package is currently called "Jazz" and is based on a non-linear editor allowing a non-programmer to connect various processing components together to produce a visualisation. This visualisation can then be distributed to other parties and either be used interactively or to "play" back a simulation visually. At the moment, Jazz exists as a prototype with development proper intended to begin before the end of the year.

This will allow dynamic monitoring and modification of the simulation process via an intuitive graphical interface. Key to this process is the planned development of a scriptable supervisor process which will moderate and coordi-

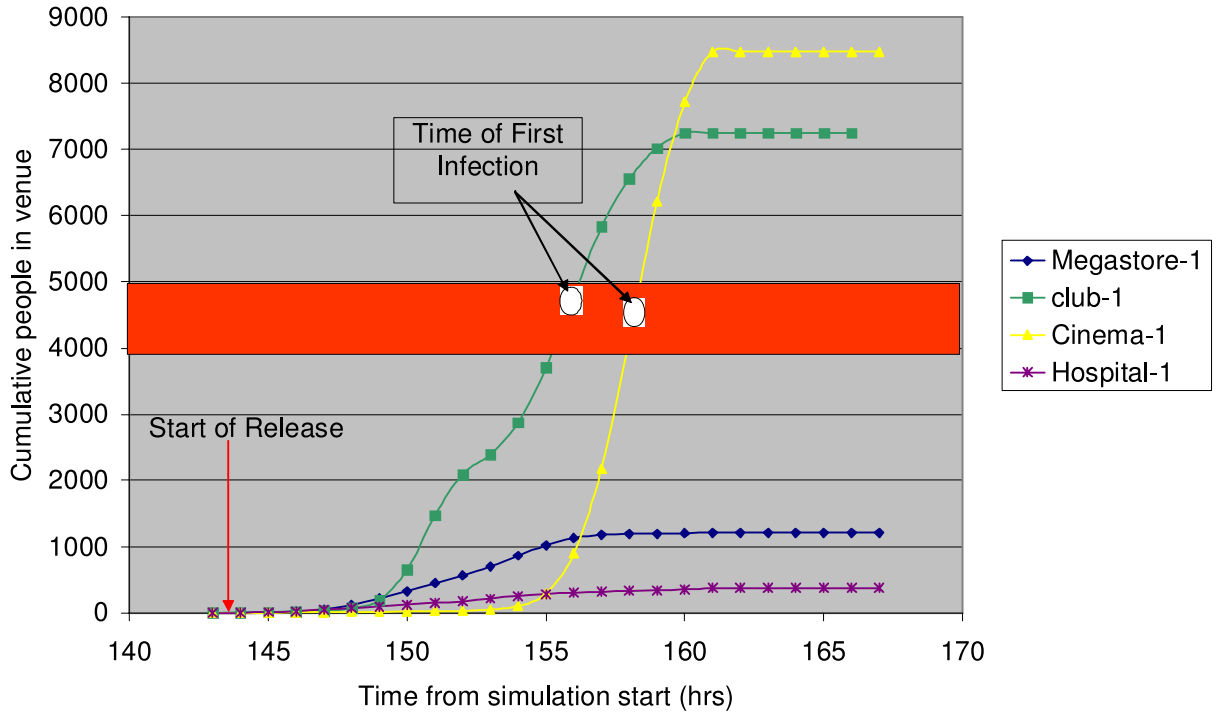

Figure 3: Terrorist smallpox simulation

Run	0	1	2	3	4	5	6	7	8
Camouflage Factor	0.8	0.8	0.8	0.8	0.8	0.8	0.9	0.95	0.99
Perception Factor	0.8	0.6	0.5	0.4	0.2	0.1	0.1	0.1	0.1
Preemptive Factor	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Interception Factor	0.16	0.12	0.1	0.08	0.04	0.02	0.01	0.005	0.001

Table 1: Interdiction Factors

nate this data exchange.

With this framework, it should be possible to integrate further state of the art simulation packages dealing with such matters as the effects of fire, explosions, etc. This would be achieved by the representation, within Simulacron, of externally simulated events via such mechanisms as a "survivability index" for a set of locations and the dynamic modification of behaviours to represent the response to death, injury and damage to locations. The addition of further Simulacron modules would allow the modelling of emergency response to such events.

One of the key aims of the development of this package was that it be capable of running at better than real-time. This will allow the coupling of non-simulated events to permit the use of the simulation environment to predict reactions and potentially to investigate alternate response strategies in a live system.

Another use is as a forensic tool to analyse past events and to investigate the likely result of alternative intervention strategies.

Work is currently being done on a prototype of the "Refinery" program which will automate the derivation of simulation parameters by iteratively refining them such that the output more closely matches historical or expected results.

Interest in this project has already been expressed by a number of groups within Australia, each of which has seen a different potential use. These range from examination of policy effectiveness in public health, through training scenarios (spot the terrorist) in law enforcement to its use as a decision support environment by emergency response organisations.

Acknowledgement

This work presented in this paper has been supported by the National Security Science and Technology Unit of Prime Minister and Cabinet, Australian Federal Government together with Australian Federal and State Agencies under the NSST CBRN Grant scheme.

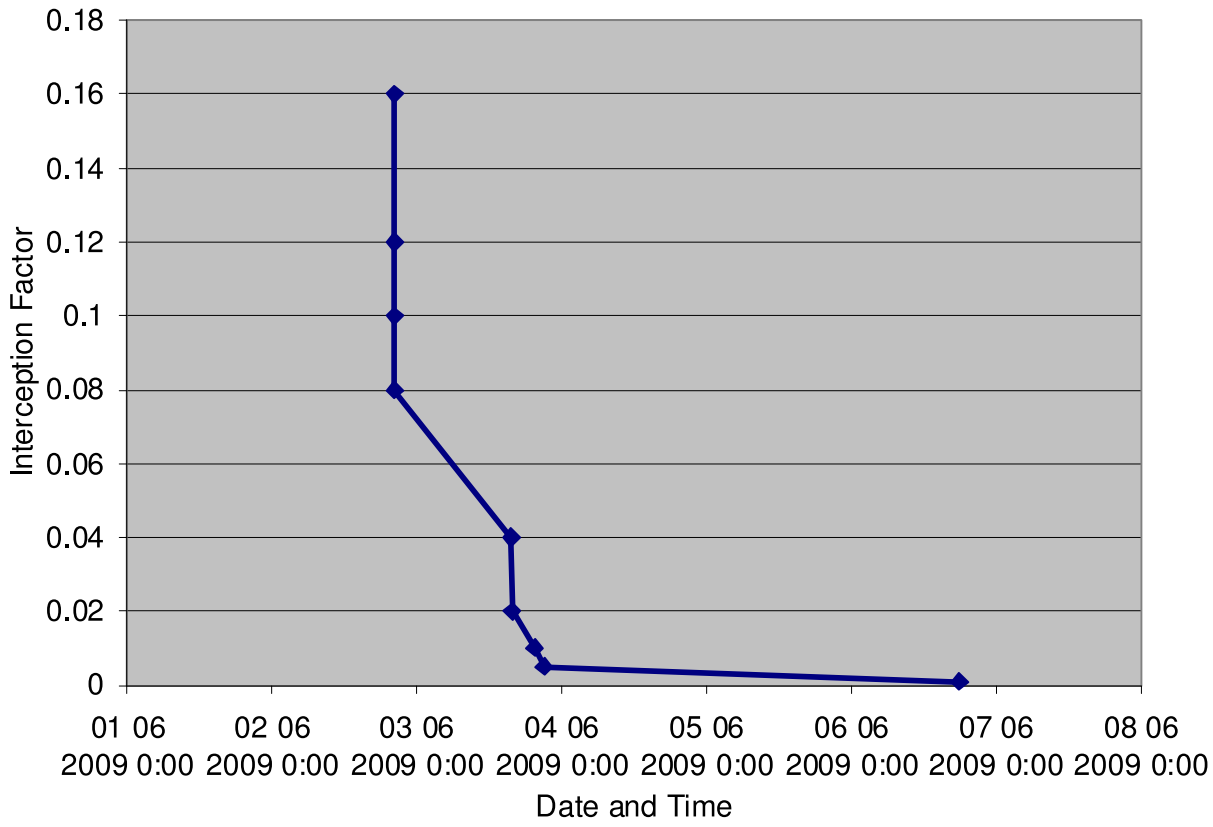

Figure 4: Interception factor

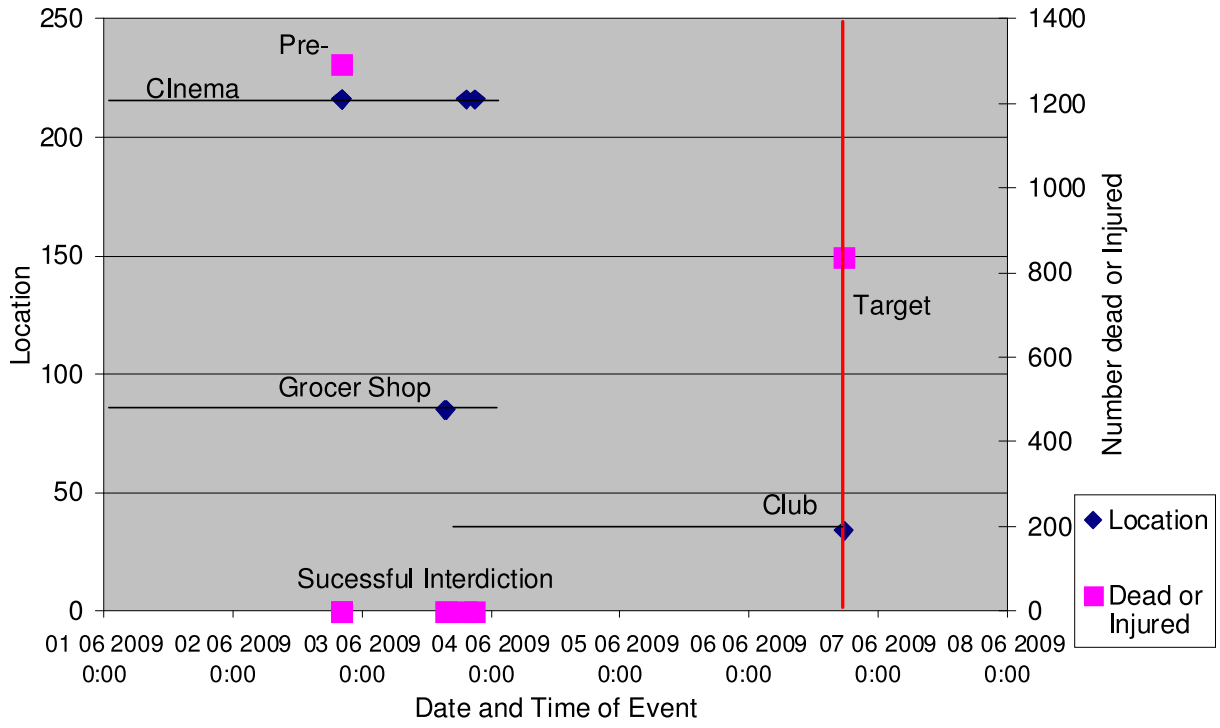

Figure 5: Event location (cell number) and number of dead or injured by interception time

References

[1] Atkinson MP and Wein LM (2008) Spatial Queuing Analysis of an Interdiction System to Protect Cities from a Nuclear Terrorist Attack *Operations Research, 56* pp. 247–254.

[2] Bold J (2000) *Greenwich, An architectural history of the Royal Hospital for seamen and the QueenŠs House* Yale University Press.

[3] CDC (2004) *"Smallpox Overview." Fact Sheet: Smallpox* www.cdc.gov/smallpox.

[4] Connor RJ, Boer R, Prorok PC, and Weed DL (2000) Investigation of Design and Bias Issues in Case-Control Studies of Cancer Screening Using Microsimulation *Am. J. Epidemiol. 151* pp. 991–998.

[5] Daley DJ and Gani J (1999) *Epidemic Modelling: An Introduction.* Cambridge University Press.

[6] Dudley S (1926) The Spread of "Droplet infection" in semi-isolated communities *Medical Research Council Special Report* His Majesty's Stationery Office.

[7] Flaherty C (2008) 3D Tactics and Information Deception *Journal of Information Warfare.* pp. 49–58.

[8] Grist RN (1979) Droplet Infection in Semi-isolated communities, Pandemic Influenza 1918. *British Medical Journal.* pp. 1632–1633.

[9] Halloran EM (2001) *Epidemiologic Methods for the Study of Infectious Diseases* Oxford University Press.

[10] M.G. Hazen, R. Burton, R. Klingbeil, K. Sullivan, M. Fewell, I. Grivell, C. Phips and P. Marland Modelling the Effects of NetCentric Maritime Warfare (NCMW) in Maritime Interdiction Operations (MIO). 8th International Command and Control Research and Technology Symposium, National Defence University, Washington DC, 17-19 June 2003

[11] Kermack WO and McKendrick AG (1927) A Contribution to the Mathematical Theory of Epidemics *Proceedings of the Royal Society of London 115* pp. 700–721.

[12] Knuth DE (1992) *Literate Programming* University of Chicago Press.

[13] Merz J (1991) Microsimulation - a survey of principles, developments and applications *International Journal of Forecasting vol. 7, no. 1* pp. 77–104.

[14] The Cradle of the Navy, National Maritime Museum; 2007. (http://www.nmm.ac.uk/server/show /conWebDoc.6490/viewPage/1).

[15] Orcutt G (1957) A New Type of Socio-Economic Syste. *The Review of Economics and Statistics vol. 39, no. 2* pp. 116–123.

[16] Pita J, Jain M, Janusz Marecki, Fernando Ordóñez, Christopher Portway, Tambe M, Western C, Paruchuri P and Kraus S (2008) Deployed ARMOR Protection: The Application of a Game Theoretic Model for Security at the Los Angeles International Airport *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)- Industry and Applications Track* pp. 12–16.

[17] Weinstein MC (2006) Recent Developments in Decision—Analytic Modelling for Economic Evaluation *Pharmacoeconomics* pp. 1043–1053.