Genetic Programming visitation scheduling in lockdown with partial infection model that leverages information from COVID19 testing

Daniel Howard
Director, Howard Science Ltd. UK. Company Fellow and former Capability Leader in AI, QinetiQ Group PLC. Member of SCR Pembroke College, Oxford and ITLAB Visiting Scholar.
dr.daniel.howard@gmail.com

ITLAB-TR-2020-02

June 3, 2020

ITLAB Laboratory Report
Inha University
South Korea
This report\textsuperscript{1,2,3} introduces a computational methodology to minimize infection opportunities for people suffering some degree of lockdown in response to a pandemic. Persons use their mobile phone or computational device to request trips to places of need or of their interest. An artificial intelligence methodology which uses Genetic Programming studies all requests and responds with granted time allocations for such visits that minimize the overall risks of infection, hospitalization and death of people. A number of alternatives for this computation are presented as well as the results of numerical experiments involving over 200 people of various ages. In particular, a model of partial infection is developed and implemented to address the real world situation whereby COVID19 testing indicates risks of infection for members of a taxonomic class - for example, age groups, exploiting such information for the aforementioned purpose.

\textsuperscript{1}Broadfield Capital sponsors this work as entry in the Humies (Human-Competitive Results) 2020 competition.
\textsuperscript{2}The author is immensely grateful to ITLAB and its leader, Professor Phill Kyue Rhee, for facilitating this official technical report publication mechanism.
\textsuperscript{3}This report, its authorship, its origin, as well as the entirety of its contents, are exclusive and original contributions of Daniel Howard PhD of Malvern UK, their sole author and exclusive owner, who not only produced the entirety of its ideas but also coded all of the supporting AI software in Visual C++ that this publication refers to and employs, during the month of May 2020. This work was funded solely by its author at Howard Science Ltd, UK.
1 The problem

In response to a pandemic, the *quaranta giorni* or forty day isolation, a practice originated by the Venetians, a sea faring and trading empire, as the name implies was a forty day isolation measure for incoming ships, to try to curtail the spread of a contagious disease or plague typically from Asia to northern Italy [1]. Modern day lockdowns -* cuarentena* in Spanish speaking countries - are far more draconian measures involving millions of people, that require that all citizens of a country stay mostly confined to the home for a good number of days. Typically a lockdown can last between one and three months before people experience easing measures or partial re-opening of society.

The 2020 lockdowns in response to the COVID19 pandemic have been enforced in different ways. The current and recent Argentine and Spanish lockdowns were enforced strictly, with people allowed out only to purchase food and medicine, often with police enforcing the rules guarding the streets, and citizens needing to make written applications for permission for outings. In some countries in north western Europe and the United States, lockdowns were not as strict. Denmark and the United Kingdom entrusted the population not to infringe lockdown rules and liberal Sweden chose not to lock down in an official sense but instead it practiced a small number of restrictive measures.

The manner of lockdown and how to exit a lockdown are problems that are short of informed solutions. Cast in terms of tradeoff in need of an invention this report is concerned with developing a solution to overcome a tradeoff which may be expressed like so:

**TRADEOFF:** *The more stringent, extensive and longer a lockdown the more effective might be society’s ability to prepare hospital facilities and/or control the spread of the disease but, also, the higher the loss of our personal freedoms; damage to the economy; detriment to personal psychology; negative effects on care of the elderly and vulnerable; higher incidence of social unrest; abusive relations in the home; higher undetected crime; higher incidence of other disease because people are too scared to visit the emergency room or their doctor.*

Inventions overcome trade-offs, e.g., the arch overcome tradeoff: ‘the heavier a mass requiring support the bigger the column’, by allowing more weight supported with thinner columns!, or by inverting the shape of a wing a bridge span is made sturdier in the presence of strong wind with lift force acting downward reversing the trade-off: ‘stronger wind wobblier bridge’. Ideally, we need an innovation can enable the lockdown to be imposed whilst minimising many of its negative and terrible consequences. Indeed, it would be nice if life could continue as normal while in lockdown, a seemingly contradictory statement! the challenge that motivated this work.

Do governments have any idea how to overcome the above tradeoff? in lockdown, when to end it and how to exit it? or how to re-introduce it in a partial way? It seems that they attempt to ensure that those who venture outdoors are fewer in number: Panama [2] used the last number of a citizen’s identity document to offer a two hour window or assigned time slot to leave home for essentials, and, at first, Spain eased the lockdown by asking that certain age groups use certain hours of the day to go out: mothers and children at this time; elderly at this other time, and so on. Although broad measures, they inspire
the work of this report. We note that, in terms of changing a general lockdown to a number of partial ones, limited work exists in the open literature [3].

2 Innovation/Solution

A solution proposed in this report is for citizens in lockdown to enter into smartphone, handset, tablet or computer, a schedule of the places that they wish to visit on that day, or future dates, together with a rough idea of the part of the day they would prefer such outings to take place. The proof of concept work in this report makes use of a data file that tabulates all such requests.

Requests may arrive at different times at a central server that processes them by continuously running and refining a Genetic Programming allocation solution. The software engine therefore, runs on a server and uses the SaaS (software as a service) model of computation and communication with clients. Details of concurrency of requests, response with the solution, and similar issues are not foreseen to be a problem, neither are issues of privacy as the procedure could be greatly anonymized and communications encrypted.

Server compute time is not an issue as Genetic Programming lends itself to parallelism and the types of computations that are revealed in this report are not daunting in nature. There will certainly be a clustering of requests and people by taxonomy of their desires so that some partitions may occur to hundreds or thousands rather than millions of requests. This would further facilitate a fast computation.

For the purposes of the proof of concept that is this report, Figure 1 is part of the data file where 282 citizens of different ages and assumed levels of health in lockdown together with their requests are listed. The health level of these citizens denoted by a number from 1 to 10 is gathered from patient records. Its intent is to capture the strength of the immune system or resistance to the pandemic infection.

There is additional information in the figure which, at first glance, appears in the nature of knowing the solution before solving the problem. It is the ‘Immune?’ field. It is a number that if 0 indicates a susceptible individual; 1 an infected individual or 2 an immune who carries a degree of antibody resistance. For the moment let us ignore this field. In the file, the 283 citizens have made approximately 1500 outing requests over three days. On average younger folk have made nine requests while much older citizens perhaps as few as three. For the purpose of the proof of concept they can be only of these types, general time day:

M wishes for the visit to take place during ‘morning’ hours;
P wishes for the visit to take place during ‘afternoon’ post meridium hours;
N wishes for the visit to take place during ‘night’ hours;
A anytime: does not mind at what hour on this day.

Requests on the same day are separated by the colon symbol. The anterior part just discussed is the part of the day while the posterior is the place. In the
Figure 1: A part of the data file used in this work and explained in the text.
proof of concept only six types of establishments are allowed and two of each:

- F may represent a ‘supermarket’ selling food;
- C may represent a sports ‘club’;
- P could represent a ‘park’;
- D may stand for ‘doctor’ a doctor’s surgery centre;
- R may stand for ‘restaurant’;
- S could represent a ‘social’ establishment;

and the number which follows can be 1 or 2 as there are two of each type or a total of 12 establishments in these simulations.

Consider one such request. Person ID 20 wants to go to Doctor’s Surgery 2 at any time of day. The day is divided into eight two hour slots. The solution aims to minimize total risk of COVID19 infection, hospitalization and death by means of a sensible allocation to these slots. It soon communicates back to Person ID 20 an allocation of time for the wish, as shown in Figure 2, that is safest in some global and personal sense. Of course, it sounds strange to visit the doctor at 10 pm!!! but the software of this demonstrator ignores common sense.

Important considerations are the following. World experts in the mathematics of epidemiology at leading institutions complain about the lack of explicit networks in current pandemic models [4]. Arguably, the solution that is contained in this report includes a network in some sense, for it reflects the imminent and future desires of trips by many people. This network information is

---

Figure 2: The solution to the problem also explained in the text.

---

*Personal communication: ‘There are a number of network approaches to modelling. If you are technically minded, then this is one of the better introductions [5]. Any infectious disease model has to include some kind of network within it, although most do not include an explicit network.’
subsumed in the solution but could be extracted and further exploited. This is
because unlike many AI systems that rely on historical data, the system actually
generates its own data as its use and its principal solution are concurrent [6].

Implementation: within the confines of anonymity such patterns of desires,
schedules and non-compliance and the timing of requests and visits can be used
as a tool to study patterns and to further improve safety and people may not
feel the intrusion. Also, there will exist a type of co-evolution between the
solution and the requests that could prove very interesting. For example, non-
compliance perhaps by taxonomic groupings could be investigated and further
solutions discovered to please certain citizens and at the same time keep them
safe.

In the first instance, the solution as described in this proof of concept is a
numerical simulation or optimization which considers all scenarios to select the
least pernicious, or one that heuristically seeks out the safest allocation of times
and places for visitors. This report implements such a search using Genetic
Programming. More sophistication might be added to searching this solution
once patterns of behaviour or infection emerge with use! For example, it could
incorporate the issue of non-compliance, speed of visits, travel times and many
other problem/solution parameters.

2.1 Applications of the solution

The solution finds its application when something is known either of the people
doing the visits or of the places visited. Consider first the former. In the real
world, it is highly unlikely that COVID19 testing will be done by all people all
of the time for if we had such complete and perfect testing we would not need
the solution and would know exactly who needs to stay and who needs to go
out. Moreover, tests are still unreliable and many have poor sensitivity and
specificity.

However, some degree of population testing for COVID19 will take place and
of sufficient importance to enable the construction of taxonomies and probabil-
ities of infection assigned to different groups of people in a given taxonomy. To
use a much more sophisticated version of the proof of concept tool in this way,
an option is to work directly with the probabilities of infection. It requires a
model of partial infections that are carried throughout the simulation. This idea
is pursued in this report and in many of the computations herein as developed
in appendix A.2. Another option is to use more traditional infection models
and to seed many different simulations with fully infected individuals randomly,
carrying out solution of many similar problems that differ in their stochasticity
to obtain average results and their standard deviations (also covered in this
report).

Another important work is to use the tool with information about differ-
ent probabilities of contamination assigned to visitation places. Basic proof of
concept computations are also included.

Finally, something not covered in this report but which may have signifi-
cant merit, would be to retrospectively reverse engineer what has transpired
and how it might have happened differently, with less infection, contamination,
hospitalization, or deaths.

This report and the proof of concept demonstrator it describes is very weak
in terms of specialist mathematics of pandemic infections. However, other than
the basic McCormack 1927 homogeneous SIR models which are useless for our purposes, research is polemic in nature with world experts consistently over-estimating the extent of pandemics infection. It is not an area where there exists overwhelming agreement in modelling.

3 Method

There exist perhaps a number of algorithms of gradient search and heuristic search that might solve the problem. Each is imbued with different advantages and disadvantages but it is not the purpose of this report to present a lengthy comparison between methods but instead to offer some insight into what is chosen and how it works. A variant of Genetic Programming used by this author to solve a number of real-world problems in consultancy in recent years is chosen and its implementation coded in the C++ language. This section broadly describes it.

The particular approach adopted here is similar to a successful GP algorithm by this author that evolves the precise numerical constants of a polynomial in the direct solution of the one dimensional homogenous convection diffusion equation [7]. That publication demonstrates that standard Genetic Programming trees are capable of computing very precise real numbers when and if needed, for example, as the precise coefficients of polynomials. Remarkably accurate numerical constants delivered in a GP determined variable-length vector are evolved and obtained in that and other refereed work [8] [9].

What is this method? Standard Genetic Programming trees contain functions to produce and output as its result, a variable length vector of real numbers as the GP tree evaluates. Many of these functions are shown in Figure 3. Such functions and terminals have been carefully optimized and experimented with over a decade. They manipulate two pointers to a record and two memory storage variables with both small and large constants seeded at the start of each run manipulated through arithmetic operations. At times, a partial result is written to memory and at other times, the memory variables are picked up or written to as the GP tree is evaluated. Most functions are of arity 2: consider L and R (left and right) branch results as indicated for some functions in the figure. Notice that small changes may result in large sections of the tree having no effect on a calculation as what is below a branch may have discontinued influence. At times a partial result is written to memory, and somewhere else in the tree evaluation an output may become over-written. Two pointers (current position and vector end) are also manipulated and change allowing the output vector to shrink and grow or certain entries to it zeroed.

The result of evaluating a GP tree is a variable length vector of real numbers. These numbers can be of any size and can be positive or negative. A subroutine then operates on these numbers to bound them as positive real numbers in size between 0.0 to 1.0 [7] as illustrated in Figure 4. How is this vector of real numbers used? Imagine there are 1500 visitation requests. The real numbers vector is consulted from left to right as when a child reads words letter by letter. Each number ranges from 0.0 to 1.0 and is processed as follows. Imagine an outing request by a person is AD2 (see person 20 Monday in Figure 1). This person wants to go to Doctor's Surgery 2 at any time of the day. The day is divided, let us say, into eight two hour slots of time. Imagine that number
Figure 3: The functions and constants in the GP tree are proprietary but here are some of them and some of what they do.

Figure 4: A subroutine in the software obtains from the resulting numbers entries which are positive but bounded between 0.0 and 1.
under consideration is 0.2763 (see Figure 4). Then this number would indicate
prescription of the third slot of time since $2/8 < 0.2763 < 3/8$. That allocation
for that visit complete, we move to the next real number in the variable length
vector to assign the time for the next visitation.

When we reach the end of the vector we continue evaluation from the first
element of this vector, thus cycling around this vector until time blocks for all
visitations have been assigned (as in Figure 2). Note that we could possibly
achieve all this with a Genetic Algorithm whose size was exactly that of the
visitations, say 1500 long. However, often an excellent solution can be achieved
with far far smaller vectors.

Practical use of the method can involve a small number of additional strate-
gies. On some problems for a number of initial generations we assign the Dar-
winian fitness as the vector size. This until a desired variable length vector
output size is reached by all members of the population. At that point the
fitness is reset and fitness now reflects how well GP trees solve the problem. Of
course GP can then reduce or increase the size of the resulting vector. The pro-
cedure is often unnecessary but can become important if the solution complexity
is high.

As the method makes use of GP trees and standard GP, then all that we
know about standard Genetic Programming including modularization options
such as ADFs [10], and subtree encapsulation [11] are applicable. Moreover, if
practitioners fear bloat or feel for whatever reason that search is compromised,
the presented method could get the solution started and then work with the
vectors of numbers directly, morphing into a variable length GA algorithm that
exploited mutation. No doubt there are many nuances of this method that could
be criticized by others but the approach works, discovers good solutions, and
appears to be quick on a portable computer, the compiled Visual Studio 2019
executable delivering solutions in just a few seconds. Other methods of AI could
be applied instead of this method or in a complementary fashion.

4 Computed Examples

4.1 Numerical experiments with a partial infection model

A partial infection model is developed specifically for this demonstration in
Appendix A.2. It must be considered that the work of this report is not so-
phisticated in mathematics of epidemic disease but instead it is an engineering
proof of concept study that may incorporate any infection model.

The simulation occupies three days which we call Monday, Tuesday, Wednes-
day, which totals approximately 1500 timed visits by 283 persons to 12 establish-
ments over eight daily two hour slot periods. At the end of each day and at
the end of all three days, the ever changing partial probability of COVID19 for
each person (denoted in this section by the symbol $p$), a number which ranges
from 0.0 to 1.0, see appendix A.2, is consulted, together with the age and gen-
eral health of the subject to determine certain outcomes. The age group of each
subject is one of 20, 30, 40, 50, 60, 70 or 80. The general health of the sub-
ject will presumably be established by a questionnaire or General Practitioner
medical records and, for the purpose of this computational implementation is
encapsulated as a number that ranges from 1.0-10.0. Both the age group and
health state of each person are fed to the computation as input data.

At the end of each Monday, Tuesday and Wednesday, the computation applies the following check to determine whether any individual is likely to self-isolate and participate no longer in visits of subsequent days. This is a determination whether the infected person feels well enough to continue and, if not, then that person will self-isolate and retire until his level of infection is assessed again at the end of computation, to determine his ultimate fate. The rules for that self-isolating decision depend on the partial probability of infection and also on their health state, as described in this table:

<table>
<thead>
<tr>
<th>age group</th>
<th>infected</th>
<th>health</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>p &gt; 0.97</td>
<td>0.95-0.97 0.0-7.0</td>
</tr>
<tr>
<td>30</td>
<td>p &gt; 0.95</td>
<td>0.92-0.95 0.0-7.0</td>
</tr>
<tr>
<td>40</td>
<td>p &gt; 0.92</td>
<td>0.87-0.92 0.0-7.0</td>
</tr>
<tr>
<td>50</td>
<td>p &gt; 0.85</td>
<td>0.80-0.85 0.0-7.0</td>
</tr>
<tr>
<td>60</td>
<td>p &gt; 0.75</td>
<td>0.70-0.75 0.0-7.0</td>
</tr>
<tr>
<td>70 and 80</td>
<td>p &gt; 0.65</td>
<td>0.60-0.65 0.0-7.0</td>
</tr>
</tbody>
</table>

Furthermore, at the end of all three days, the ultimate fates/outcomes for all persons, perhaps after a further month period, are computed and determined from once again from their final partial infection level and original health state. It is done by a set of pre-determined rules.

The outcomes are three. The ‘none’ outcome means that the person was either not infected or infected but presented as asymptomatic, a common finding in young people. In this outcome the person did or did not develop anti-bodies to COVID19, although this last factor is not incorporated in a measure of solution goodness.

The second possible outcome is ‘hospitalized’. This assumes that the person reached hospital and perhaps the ICU of the accident and emergency department, but did not die. The person was required to convalesce in a special facility until one or more COVID19 tests finally released the person back to the community. In this case surely the person developed anti-bodies but that again does not concern this proof of concept.

The third outcome is ‘death’ with or without a period of hospitalization. The rules as to who ends up in hospital, who will recover or who will very regrettably pass away are:

---

5 The table is different for the case of full infection but there was no time to include that table in the report. However, that table is equivalent for the case of fully infected individuals and is thus only concerned with health levels.

6 It is important to understand that persons registered as dead in this proof of concept do not register as hospitalized.
The above table expressed as a computational function returns both the total number of persons who end up hospitalized $N_H$ or sadly deceased $N_D$. The former does not include the latter. The former are people who ended up in hospital but who did not die. That is in the nature of this proof of concept computational study but can easily be changed.

How are such computations driven and why do they qualify as real-world? Although COVID19 testing will never become universal and never become completely accurate we can use indications or some evidence from such noisy numbers to advise policy or to plan the intended visitations to places during a lockdown.

Figure 5 shows the current implementation of entry of these drivers. Imagine testing reveals that more people of age groups A or B have COVID19 than of age groups C or D. This could be due to a great number of reasons. Rather that develop a phenomenological model which is greatly suspect of being a littered with false causalities to explain the differences in findings, we can use such 'sentiments' as inputs, and therefore drivers, to the model. Indeed, much research considers the existence of subgroups by their susceptibility to disease [12].

The figure shows the current implementation whereby the user enters differences in probability of COVID19 attributed to different age groups. These are probability percentages that vary from 0.0 to 1.0. What transpires in the software being that every individual in that age group class is assumed to be infected to that partial level. The visitations then take place according to the direction of the Genetic Programming solution with the probability calculus that is presented in appendix A.2. All allocations to the roughly 1500 visits are made by GP and the simulation and judgements as above are made. Finally a fitness function $F_F$ or measure of solution goodness is constructed which depends on the ultimate outcomes for the people, i.e, the combination of the total number of hospitalized $N_H$ together with the total number of sadly deceased
Figure 5: Prior probability of infection by taxonomic group are entered for the runs. In this implementation taxonomy is assumed to depend on age group.

\[ N_D \text{ weighted by some desirability constant } W_C \text{ criterion:} \]

\[ F_F = -1.0[(1 - W_C)N_H + W_CN_D] \]

Runs of this report use \( W_C = 0.65 \), reflecting desire to reduce fatalities. This is one possible fitness function and it must be noted that there is a connection between the number of people who end up in hospitals and those who eventually sadly die, and that most are elderly. An alternative fitness function is to compute the final average rate of infection. If \( n_{2030} \) is the final average probability of infection of 20 and 30 year olds and \( n_{4060} \) the final average probability of infection of 40, 50 and 60 year olds with \( n_{7080} \) being the final average probability of infection of 70 and 80 year olds then a possible fitness measure is:

\[ F_F = -1.0(W_1n_{2030} + W_2n_{4060} + W_3n_{7080}) \]

\( W_1 > 0; W_2 > 0; W_3 > 0 \) and \( W_1 + W_2 + W_3 = 1 \)

and it would be desirable to ensure \( W_3 > W_2 > W_1 > 0 \) because the mortality of the young from COVID19 is low and that of the elderly is very high. However, it suffices to show by a small worked out example that enjoying a lower average infection rate as worked out by the partial infection rules of section A.2 may not result in lower hospitalization and or death rates. The manually worked out example is shown in Figure 6.

The system can execute a great number of parallel independent runs. As one of these runs transpires, the system stores the most fit solution as it emerges while the parallel independent run transpires. The user can run one or many more subsequent parallel independent runs and the software again stores best solutions as they emerge. When the runs end it then automatically ranks all of these solutions from all runs into a global ranking list ordered by highest fitness attained from which the user can select a run. Thus, the user is then able to select any of these solutions to observe all of its details in the interface and compare them at a glance.

Figure 7 provides a key to interpret a part of the output screen. There is total information about visitations which transpired during the day and also at each establishment visited at each time of that day. Other parts of the
Figure 6: In an example involving seven people changes in the allocation of visits for persons can result in higher death rates or hospitalizations regardless of the final average infection for the population: it can be higher or lower. The standard deviation or a measure of a whiskers diagram incorporated into the fitness would help this situation if the quantitative prediction of hospitalization rates and death rates are suspect/unreliable/problematic.

Figure 7: Part of the output interface giving details of number of participants, partially infected participants and newly infected participants (those who had been fully susceptible prior to the visit) for each establishment visited at each time epoch, and summaries for the day.
interface illustrated in Figure 8 give details about the number of persons who feel unwell and retire from future visits to become self-isolated. What tends to happen in these partially infected computations is that few fall sufficiently ill after Monday but many do so after Tuesday as their level of partial infection increases beyond the aforementioned thresholds. The figure also shows another part of the interface summarizes the maximum fitness level.

Figures 14 to 20 present computed results when the driver is only that 20 year olds have a three percent incidence of COVID19 in the population and all other age groups are assumed to be perfectly disease free. Best results of short runs with thirty generations and with small GP populations are developed to show how Pareto front emergence of non-dominated solution arises involving the two parameters \( N_H \) and \( N_D \).

The final infection rate can be seen to be very high and the number of hospitalizations and number of deaths also. This will be discussed in the Conclusion section.

4.2 Numerical experiments with a normal infection model

The idea of the partial infections approach seems the most sensible approach to real world application. If for example, it is known which persons are infected then they can be self-isolated at home!

However, an alternative to the partial infection approach of the previous section is to assume that a number of persons are infected and run the computations without partial infection. Presumably in real world, this could be performed a number of times with different stochastic selections about who is infected. Again, this would be as alternative to the method of the previous section and still a real world application for the rate of infection for different groups in the taxonomy would guide how many people in the data file from different groups should be infected.

The runs can then be performed numerous times to a level of statistical significance each with a different choice of inbound infected people in the data file as per the informed assumption.

The infection model is again an assumption, and its numbers are produced via a Monte Carlo process. The arbitrary infection model is explained in appendix A.3 and results of computations are presented in Figures 21 to 24.

Note that information in the output screen is slightly different as there are no partially infected individuals but only susceptible, immune and infected. Figure 9 provides a key to interpret a part of the output screen. There is total information about visitations which transpired during the day and also at each
Figure 9: Part of the output interface giving details of number of participants, partially infected participants and newly infected participants (those who had been fully susceptible prior to the visit) for each establishment visited at each time epoch, and summaries for the day.

establishment visited at each time of that day.

Again the number of hospitalized and of dead appears too high and this will be discussed in Conclusions.

4.3 Experiments that avoid contaminated places

It is possible to run the current software marking establishments at different times as contaminated. The computations are trivial if Genetic Programming can schedule visitors elsewhere so that nobody gets infected and become more interesting when this is not completely possible.

5 Conclusions

This work concentrated on developing the software, the problem and the Genetic Programming solution. Although the levels of hospitalizations appear to be high and of death also, it must be considered that the infection models introduced into these computations are pervasive.

For example, one would not expect a probability of 1/4 to get infected in a supermarket because there is a certain social distancing. The chance of receiving a drop of saliva with COVID19 and getting infected is very much lower, and unlike in the manner of these computations and the advice of some, when coupled with the chance of a direct bump of all susceptibles with all infected, the chances are surely much lower.

The partial infection model is developed here for the first time. If there is something similar in the literature then this author does not know of it. It must be tested and developed better. It is possible that the pessimistic approach of multiplying the terms of worst case is not entirely reasonable.

Lastly, how well infections can be avoided will depend on the number of visitations, the number of establishments visited, the number of people and the frequency of visits. In this research Genetic Programming was handed a tough challenge as the number of visitations was in the order of 1500 and the times slots very few per day. Also the number of establishments was low.
In general, it is said that washing one’s hands is far more effective than social distancing. It is probably true that contamination is more important than person to person transmission. Contamination can be easily incorporated into the model.

Although the model is incipient, it is considered something few have considered if any. Most research is involved in exploiting data sources to predict infection levels or explain the disease. This contribution is different in nature because its deployment could generate data and would also need only tendency of disease data. This contribution is markedly different from contact tracing approaches that are reactive. The work described here and its implementation would be proactive but it could also inform and be informed by contact tracing.

A Appendix: Simple infection models for the computations

A.1 When chance encounters underlie infection

When a susceptible S individual meets an infected individual I there is a possibility of transmission of the disease from I to S. This is usually expressed as an infection probability $p_n$ where $n$ is the number of infected individuals present in the location of the encounters between the susceptible individual and $n$ infected individuals. Sophisticated relations can be determined in function of the number of encounters between a susceptible individual and infected individuals that also incorporate time of exposure modelled as a Poisson process. \[8\]

An important characteristic of such relations is that as the number of infected individuals $n$ grows then $p_n$ increases but not linearly, so that for example one can expect $p_2 < 2p_1$. We will assume, for the purpose of illustration in this work, a very simple infection function based on counting contacts between S and I, that increases with the number of infected individuals but which never exceeds one or as $n \to \infty$ then $p_\infty \to 1.0$. The essence of this behaviour can be crudely emulated using a Monte Carlo approach or, as discussed here, using simple probability trees as illustrated in Figure 10. We ignore dependency on time of exposure to assume all visits to places are of similar duration. The Genetic Programming approach that uses such infection functions, however, is general and able to incorporate any candidate infection function.

From this figure, first, consider the case of two individuals only: P1 and P2. Individual P1 is susceptible while P2 is infected. The number of encounters between these two, the red boxes at the P2 level in the figure, or where both individuals share a same location, is four and the total number of possible outcomes is sixteen. Hence, the opportunity for an encounter taking place and therefore for infection, assuming each individual should have an equal presence

---

7 As contrasted with place contamination and contact with things, and the abating of infection by the washing of hands

8 Personal communication: ‘the risk of transmission increases non-linearly with the number of infected and with time. Suppose I go to the garden centre for one hour and there is one other person, and it gives me a rate, $p$, of being infected each hour we are there. So the probability of me being infected at the end of the hour is $1-\exp(-p*1)$, and if we are there 2 hours: $1-\exp(-p*2)$. If there are $n$ people there then the time I spend contacting each person drops, so the risk per person drops. A reasonable assumption is that the risk with $n$ people is $p*n$, where $c < 1$. This gives you a contamination function.’
for each of the four locations and spend the same amount of time at any visited, is $p_1 = 0.25$. Next consider the case where three individuals participate with P1 susceptible and P2 and P3 both infected. Now we recognize opportunities for all three individuals to exist at the same location, and also opportunities for P1 to share a location with either infected P2 or P3. If we count such opportunities we arrive at $p_2 = 28/64 = 0.4375$ and we verify that $p_2 < 2p_1$. For the particular scenario of Figure 10 the coarse emulation leads to a simple relation to obtain the probability of encounter for any $n$ that is: $p_n = 1 - (3/4)^n$. Considering rather large $n$ it tends to one: $p_{20} = 1 - (3/4)^{20} = 0.9968$.

A.2 Introducing: partially infected people

We now introduce perhaps what may be an unusual idea. The real world objective of this initiative is to leverage off COVID19 testing. Imagine that upon testing for COVID19 the incidence is found to be higher in age groups 20 and 50. It is unlikely that COVID19 testing will be done by all people all of the time. Moreover, tests are still unreliable. Hence, we need to work with partial knowledge. An option to explore might be to work directly with the probable levels of infection that are informed by the testing for different groups of people as organized in some taxonomy: for example, consider the probability of infection to be 5 percent and 8 percent respectively and zero in other age groups. It motivates use of a ‘partial infection’ but why? If we knew who was or was not infected we would let them out or keep them in isolation. However, as we cannot test every single person, it is useful to assign to them uncertainty, a probability level. In such a case, we must consider that any of the people in Figure 10 namely P1, P2 or P3 may be partially infected (and partially susceptible).

The figure shows the encounters between persons, i.e., two people sharing one location (1,2,3,4) at the same moment in time, thus coming into contact with each other. We are especially interested in two encounters: P1-P2 and P1-P3 as we can assume that one person P1 is susceptible while the other two are infected. Note that under permutations assigning I and S there is no need to count encounters P2-P3 as these would represent two infected, and neither of interest are encounters between two susceptibles. The thirty two encounters that determined the probability of infection result in the union that gives twenty

---

Figure 10: Location probability tree involving three individuals P1, P2, and P3 who may occupy one of four possible locations: 1, 2, 3 or 4.
eight encounters in the figure. Each of P1-P2 and P1-P3 result in 16 encounters but 4 encounters are shared. So only 28 out of 64 possible combinations of three locations are of interest. This ratio gives a crude probability of infection \( p_2 = \frac{28}{64} = 1 - \left(\frac{3}{4}\right)^2 = 0.4375 \). We now consider permutations of I and S components of partially infected persons in such basic probability tree count calculations as will become apparent in the below examples\(^\text{11}\).

### A.2.1 An example involving three people

As an example, consider the interactions between three people with partial infection:

- P1 has I = 0.03 and S = 0.97,
- P2 has I = 0.04 and S = 0.96,
- P3 has I = 0.01 and S = 0.99.

To arrive at an infection probability we need to consider contributions from the following six possibilities\(^\text{12}\): ISI, IIS, ISS, SII, SSI, SIS. Here are the numerical contributions from each term:

- ISI = \(\left(0.96 \times 0.03\right)(16 - 4) + (0.96)(0.01)(16 - 4) + (0.96)(0.03)(4)\)/64 = 0.0090,
- IIS = \(\left(0.99 \times 0.03\right)(16 - 4) + (0.99)(0.04)(16 - 4) + (0.99)(0.04)(4)\)/64 = 0.0155,
- ISS = \(\left(0.99\right)(0.03)/(16)/64 = 0.0074, \) subsumed in ISI and with 0.99 in IIS
- SII = \(\left(0.97 \times 0.01\right)(16 - 4) + (0.97)(0.04)(16 - 4) + (0.97)(0.04)(4)\)/64 = 0.0115,
- SSI = \(\left(0.97 \times 0.01\right)/(16)/64 = 0.0024, \) subsumed in SII and with 0.96 in ISI
- SIS = \(\left(0.99 \times 0.04\right)/(16)/64 = 0.0099, \) subsumed in IIS and with 0.97 in SII

Note that the encounters of overlap where all three persons are in the same location give preference to the largest infection, hence the third term which multiplies by four, the overlapping cases (see Figures 10, 11 and 12) The total probability of infection is the maximum of ISI, IIS and SII or 0.0155. Now we compute the new partial infections for our three participants:

- P1 has I = (0.97)(0.0155) + 0.03 = 0.0450 and S = 0.9550,
- P2 has I = (0.96)(0.0155) + 0.04 = 0.0549 and S = 0.9451,
- P3 has I = (0.99)(0.0155) + 0.01 = 0.0253 and S = 0.9747.

### A.2.2 A second example involving three people

Here is a second example, consider the interactions between three people with partial infection:

- P1 has I = 0.95 and S = 0.05,
- P2 has I = 0.98 and S = 0.02,
- P3 has I = 0.01 and S = 0.99.

\(^{11}\)Out of interest the number of non encounters (three different locations for the three persons) are twenty four: 123, 124, 132, 134, 142, 143, 213, 214, 231, 234, 241, 243, 312, 314, 321, 324, 341, 342, 412, 413, 421, 423, 431, 432.

\(^{12}\)there is no need to consider SSS or III
which approaches the situation of one susceptible person, P3, meeting two infected: P1 and P2. Again to arrive at an infection probability we consider contributions from the following six possibilities: ISI, IIS, ISS, SII, SSI, SIS. Here are the numerical contributions from each term:

\[
\begin{align*}
\text{ISI} &= \frac{[(0.02)(0.95)(16) + (0.02)(0.01)(12)]}{64} = 0.0048, \\
\text{IIS} &= \frac{[(0.99)(0.95)(12) + (0.99)(0.98)(16)]}{64} = 0.4189, \\
\text{SII} &= \frac{[(0.05)(0.01)(12) + (0.05)(0.98)(16)]}{64} = 0.0115.
\end{align*}
\]

The total probability of infection is the maximum of ISI, IIS and SII or 0.4189. This number is close to \( p_2 = \frac{28}{64} = 0.4375 \). Now we compute the new partial infections for our three participants:

- P1 has \( I = (0.05)(0.4189) + 0.95 = 0.9709 \) and \( S = 0.0291 \),
- P2 has \( I = (0.02)(0.4189) + 0.98 = 0.9884 \) and \( S = 0.0116 \),
- P3 has \( I = (0.99)(0.4189) + 0.01 = 0.4247 \) and \( S = 0.5753 \).

Note that in spite of P3 already carrying a small probability of infection, its new infection level \( 0.4247 < 0.4375 \). This is because others were not one hundred percent infected and the result was close to but less than 0.4375.

**A.2.3 A third example involving three people**

Here is a third example, consider the interactions between three people with partial infection:

- P1 has \( I = 0.01 \) and \( S = 0.99 \),
- P2 has \( I = 1.00 \) and \( S = 0.00 \),
- P3 has \( I = 0.01 \) and \( S = 0.99 \).

which approaches the situation of the interaction of the non-infected with a fully infected person. Again to arrive at an infection probability we consider contributions from the following six possibilities: ISI, IIS, ISS, SII, SSI, SIS. Here are the numerical contributions from each term:

\[
\begin{align*}
\text{ISI} &= \frac{[(0.00)(0.01)(16) + (0.00)(0.01)(12)]}{64} = 0.0, \\
\text{IIS} &= \frac{[(0.99)(0.01)(12) + (0.99)(1.00)(16)]}{64} = 0.2493, \\
\text{SII} &= \frac{[(0.99)(1.00)(16) + (0.99)(0.01)(12)]}{64} = 0.2493.
\end{align*}
\]

The total probability of infection is the maximum of ISI, IIS and SII or 0.2493. This number is very close to \( p_1 = \frac{4}{16} = 0.25 \). Now we compute the new partial infections for our three participants:

- P1 has \( I = (0.99)(0.2493) + 0.01 = 0.2568 \) and \( S = 0.7432 \),
- P2 has \( I = (0.00)(0.2493) + 1.00 = 1.0000 \) and \( S = 0.0000 \),
- P3 has \( I = (0.99)(0.2493) + 0.01 = 0.2568 \) and \( S = 0.7432 \).

Notice that P1 and P3 have an infection level \( 0.2568 > 0.2500 \) this is because the people were already one percent infected.
A.2.4 A first example involving four people

An example involving four people is illustrated with the help of Figure 11, consider the interactions between three individuals with partial infection:

- P1 has I = 0.01 and S = 0.99,
- P2 has I = 0.98 and S = 0.02,
- P3 has I = 0.03 and S = 0.97,
- P4 has I = 0.05 and S = 0.95.

Again to arrive at an infection probability we consider contributions from the following: SIII, ISII, IISI, IIIS. Here are the numerical contributions from each term:

\[
SIII = \frac{(0.99)(0.98)(64) + (0.99)(0.03)(36) + (0.99)(0.05)(48)}{256} = 0.2560,
\]
\[
ISII = \frac{(0.02)(0.01)(36) + (0.02)(0.03)(48) + (0.02)(0.05)(64)}{256} = 0.0004,
\]
\[
IISI = \frac{(0.97)(0.01)(36) + (0.97)(0.98)(64) + (0.97)(0.05)(48)}{256} = 0.2481,
\]
\[
IIIS = \frac{(0.95)(0.01)(36) + (0.95)(0.98)(64) + (0.95)(0.03)(48)}{256} = 0.2394.
\]

The total probability of infection is the maximum of SIII, ISII, IISI and IIIS, or 0.2560. This number is very close to \( p_1 = 1 - 3/4 = 1/4 = 0.25 \). It is a higher value because all participants contribute a significant level of infection. Now we compute the new partial infections for our four participants:

- P1 has I = (0.99)(0.2560) + 0.01 = 0.2634 and S = 0.7366,
- P2 has I = (0.02)(0.2560) + 0.98 = 0.9851 and S = 0.0149,
- P3 has I = (0.97)(0.2560) + 0.03 = 0.2783 and S = 0.7217,
- P4 has I = (0.95)(0.2560) + 0.05 = 0.2932 and S = 0.7068.

A.2.5 A second example involving four people

In this example we approach the situation of three infected persons, consider the interactions between three individuals with partial infection:

- P1 has I = 0.01 and S = 0.99,
- P2 has I = 0.98 and S = 0.02,
- P3 has I = 0.97 and S = 0.03,
- P4 has I = 0.99 and S = 0.01.

Once again we consider contributions from the following: SIII, ISII, IISI, IIIS:

\[
SIII = \frac{(0.99)(0.98)(48) + (0.99)(0.97)(36) + (0.99)(0.99)(64)}{256} = 0.5708,
\]
\[
ISII = \frac{(0.02)(0.01)(36) + (0.02)(0.97)(48) + (0.02)(0.99)(64)}{256} = 0.0086,
\]
\[
IISI = \frac{(0.03)(0.01)(36) + (0.03)(0.98)(48) + (0.03)(0.99)(64)}{256} = 0.0519,
\]
\[
IIIS = \frac{(0.01)(0.01)(36) + (0.01)(0.98)(48) + (0.01)(0.97)(64)}{256} = 0.0043.
\]

Note from previous examples that if we simply identify the participant with the highest component of S then that will be the one that indicates the highest contribution. In this case that is P1 and so SIII should be highest, and it is and
is 0.5708. This number is very close to $p_3 = 1 - (3/4)^3 = 0.5781$. It is a higher value because all participants contribute a significant level of infection. Now we compute the new partial infections for our four participants:

P1 has $I = (0.99)(0.5708) + 0.01 = 0.5751$ and $S = 0.4249$,
P2 has $I = (0.02)(0.5708) + 0.98 = 0.9914$ and $S = 0.0086$,
P3 has $I = (0.03)(0.5708) + 0.97 = 0.9871$ and $S = 0.0129$,
P4 has $I = (0.01)(0.5708) + 0.99 = 0.9957$ and $S = 0.0043$.

### A.2.6 Algorithm for modelling partially infected people

For $n + 1$ partially infected people meeting at $s$ possible locations, e.g., $s = 4$, there are $n$ multiplicative constants. In the first three examples with $n = 2$ they are: 16 and 12 which sum to 28. In the last two examples with $n = 3$ they are: 64, 48 and 36 which sum to 148. For $n = 1$ there is only one constant: 4. The constants then get divided by $s^n$ (16, 64, 256,...).

Figures 10 and 11 illustrate that the sum of these constants divided by $s^n$ gives the infection probability for the pure case of encounter of a fully susceptible individual with $n$ infected, e.g., $p_2 = 28/64 = 0.4375 = 1 - (3/4)^2$ and $p_3 = 148/256 = 0.5781 = 1 - (3/4)^3$. With $n = 1$ then $p_1 = 1/4 = 0.25 = 1 - (3/4)$ and the only consideration in such a case are cases SI and IS for the P1-P2 encounter. A formula to obtain the $n$ multiplicative constants $k_j$ where $j = 1, ..., n$ is:

$$k_j = s^{n+1-j}(s-1)^{j-1} ... j = 1, ..., n.$$  

This can be understood pictorially at the top (right) of Figure 12. When division by $s^n$ is considered the multiplicative ratios $g_j$ in the above examples such as 16/64, 48/256 are given by:

$$g_j = (1/s)((s-1)/s)^{j-1} ... j = 1, ..., n.$$  

These factors can be seen to be in descending order. If $n_p$ people participate in an encounter but the number of partially infected people $n$ is less than $n_p$ then we assume one fully susceptible individual with $I=1$ and $S=0$ will meet with the $n$ partially infected individuals and thus $S = 1$ is used in the formula. The calculation follows this ten step procedure (pseudo-code):

1. pre-compute a large number of factors $g_j$ at the start of the run.
2. identify all participants $n_p$ that are assigned to this visit $v$;
3. if all $n_p$ are fully infected then exit the procedure;
4. if none are partially or fully infected exit the procedure;
5. identify the $n$ subset of infected or partially infected out of the $n_p$;
6. if $n < n_p$ include an $S_v = 1, I_v = 0$ participant, use $S_{max} = 1$, and increment $n$;
7. otherwise identify $S_{max}$ and that participant with the highest value of $S_v$;
8. sort the infected in descending order by their component of infection $I_v$;
9. prepare the products ($S_{max} I_v g_j$) and sum them to get the infection level;
10. use it to update infection levels of all $n_p$ participants: $I_{v+1} = p S_v + I_v$. 

Figure 11: Location probability tree involving four individuals P1, P2, P3 and P4 who may occupy one of four possible locations: 1, 2, 3 or 4.

A.3 Non-standard infection function for other testing

An alternative presumed infection function is conceived inspired in the idea of an ‘infection pressure’ and of an infection resistance. It is imagined as follows.

A Monte Carlo simulation generates probabilities of infection as shown in Figure 13.

As an alternative to a Poisson distribution a Monte Carlo process carries out 100,000 evaluations of 40 trials for one of twenty infected persons. So for each person it aims to discover where is the person on 40 occasions. The occasions fit into 7200 seconds of time (two hours). The random number indicating the location is chosen such as to reflect that some areas of the store or establishment are more popular than others (e.g., browsing a magazine rather than simply walking through the area). The person spends either 2 seconds or 30 seconds or 102 seconds in the selected area.
Figure 12: Understanding the formula for the multiplicative constants pictorially.
int nTries = 40; int nINFECTED = 20;
for (k = 0; k < nTries; k++)
{
    for (iInfected = 0; iInfected < nINFECTED; iInfected++)
    {
        iInfectedLoc = GPg_random(7200);
        if (iInfectedLoc < 600)
        {
            kr = 0; j = iInfectedLoc - kr; nTrial_I[iInfected][k] = j / 2;
        }
        else if (iInfectedLoc < 2100)
        {
            kr = 600; j = iInfectedLoc - kr; nTrial_I[iInfected][k] = 300 + j / 30;
        }
        else
        {
            kr = 2100; j = iInfectedLoc - kr; nTrial_I[iInfected][k] = 350 + j / 102;
        }
    }
}

The same is done for one susceptible person. Once these forty trials are computed for the one susceptible and the twenty infected persons, the procedure checks to see whether a susceptible and one or more infected shared an area in those 40 opportunities (40 sampling).

for (kI = 0; kI < nTries; kI++)
{
    indS = nTrial_S[kI];
    for (ik1 = 0; k1 < nINFECTED; k1++)
    {
        if (indS == nTrial_I[k1][kI])
        {
            for (int k2 = k1; k2 < nINFECTED; k2++)
                bT[k2] = true;
        }
    }
}

// risk of infection of susceptible walking into this place
for (iI = 0; iI < nINFECTED; iI++)
{
    if (bT[iI])
        nInfection[iI]++;
}

At the end of the procedure the nInfection array values are divided by the number 100,000 or Monte Carlo trials resulting in the infection probabilities of Figure 13. If the susceptible encounters more than 20 infected then a probability of infection equal to 1 is assumed. Note that a lower or higher rate of infection could be obtained by changing the sampling from 40 to a smaller or larger number.
Figure 13: Probabilities of infection $p_n$ for various $n$ generated by a Monte Carlo process.

References

[1] cdc.gov/quarantine/historyquarantine.html
[3] https://www.medrxiv.org/content/10.1101/2020.03.29.20046011v2
Figure 14: With $s = 4$ for partial infection model.
Figure 15: With $s = 4$ for partial infection model.
Figure 16: With $s = 4$ for partial infection model.
Figure 17: With $s = 4$ for partial infection model.
Figure 18: With $s = 4$ for partial infection model.
Figure 19: With $s = 4$ for partial infection model.
Figure 20: With $s = 4$ for partial infection model.
Figure 21: Run which reads infected, immune and susceptibles from data.
Figure 22: Run which reads infected, immune and susceptibles from data.
Figure 23: Run which reads infected, immune and susceptibles from data.
Figure 24: Run which reads infected, immune and susceptibles from data.