www.sciencemag.org/cgi/content/full/1167053/DC1

Supporting Online Material for

Understanding the Spreading Patterns of Mobile Phone Viruses

Pu Wang, Marta C. González, César A. Hidalgo, Albert-László Barabási*

*To whom correspondence should be addressed. E-mail: barabasi@gmail.com

Published 2 April 2009 on *Science* Express
DOI: 10.1126/science.1167053

This PDF file includes:

Materials and Methods

Figs. S1 to S8

Tables S1 to S3

References

Supporting Online Material

Pu Wang, Marta C. González, César A. Hidalgo, Albert-László Barabási

TABLE OF CONTENTS 1

PART 1. DATA

I. **Introduction** 2

PART 2. METHOD

I. **Introduction** 4

II. **Determining the user characteristics** 4

III. **Simulating mobility patterns** 6

IV. **Validation** 8

PART 3. VIRUS SPREADING

I. **SI model** 12

II. **Simulation of the Bluetooth virus infection process** 13

III. **Simulation of the MMS virus infection process** 14

IV. **Simulation of the hybrid virus infection process** 15

V. **Estimating m_c and m (the real market share)** 15

VI. **Comparison between hybrid virus and Bluetooth/MMS virus** 17

PART 4. CALCULATIONS AND FITTINGS

I. **The latency time preceding a Bluetooth virus outbreak** 19

II. **Theoretical calculation of the giant cluster size** 19

III. **Statistical analysis** 20

REFERENCES 22

PART 1. DATA:

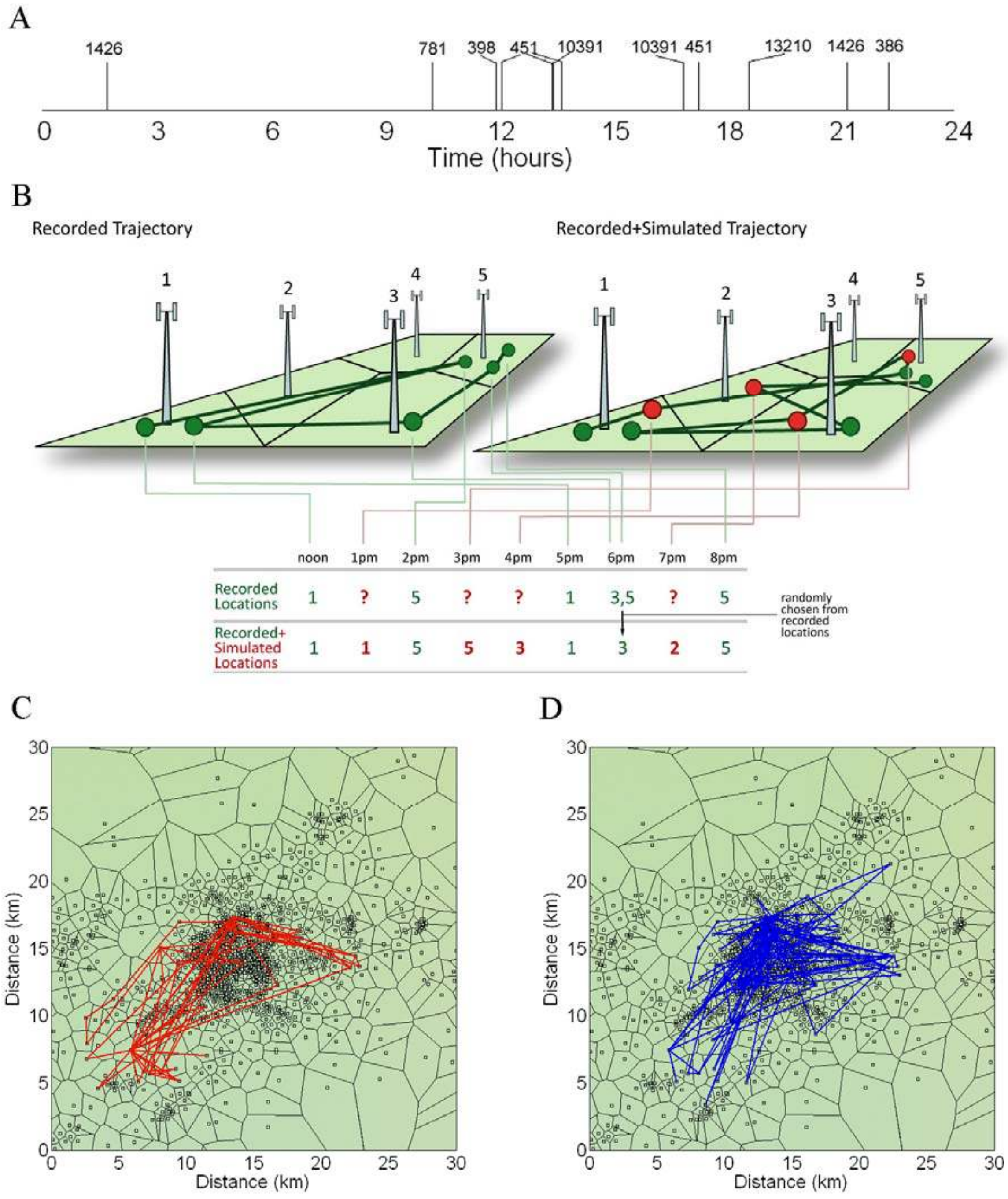
I. *Introduction*

The studied dataset was collected by a mobile phone carrier for billing and operational purposes. It contains the date, time and coordinates of the phone tower, and a record of each phone call and text message sent or received by each of approximately 6.2 million costumers. The dataset summarizes one month of activity. To guarantee anonymity, each user is identified by the data source with a security key (hash code) and any potential personal identifiers were previously removed from the dataset.

In our dataset, there are over 10,000 mobile phone towers, the Voronoi diagram around each tower offering a reasonable estimate of the tower's service area. The Voronoi diagram areas are obtained by Delaunay triangulation (*1*), finding that the distribution of the tower areas follows a power-law with two different scaling regions (Fig. S8A).

The cell phone dataset records the communication pattern of the user base, providing the time and the tower location for each call. Yet, given that the interevent time between consecutive calls is irregular, the sampling is often too sparse (Fig. S1A). To identify the hourly location we developed the following procedure: we discretized time, breaking each day into 24 hour long time intervals (Fig. S1B). If the user calls during a specific hour period, we set user's position to the tower indicated by the dataset. If there are multiple locations recorded in the same hour long period, we choose one of the recorded locations with a probability that in proportional to the frequency of the towers observed in that time period. For example, if there are two calls from two towers between 1 and 2pm, we will randomly choose one for the user's location. If, however, there are two calls from tower one and one call from tower two, then with 0.67 chance we choose tower one. Given the sparseness of the calls, this leaves us with an incomplete location record. Therefore, we developed a methodology to identify the user's likely location building on the results of Ref. (*2*) on human mobility patterns (see Section 2.III and Fig. S1, C and D).

Further information can be obtained from the authors upon request.



	noon	1pm	2pm	3pm	4pm	5pm	6pm	7pm	8pm
Recorded Locations	1	?	5	?	?	1	3,5	?	5
Recorded+ Simulated Locations	1	1	5	5	3	1	3	2	5

Fig. S1. Predicting the movement of mobile phone users. (**A**) Temporal patterns of communication. The numbers listed are the towers' IDs to which the user's phone connected during a call. (**B**) A cartoon showing users' recorded locations and simulated locations. (**C**) A user's trajectory. (**D**) The user's simulated trajectory based on the method described in Section 2.III. Each Voronoi cell approximates a tower's service area.

PART 2. METHOD:

I. *Introduction*

To identify the user locations with an hourly frequency we follow a two step procedure. First, we determine each user's travel characteristics following the results of Ref. (*2*). Second, we reconstruct the position of the user at times when there are no calls such that the fundamental characteristics of each user are preserved. We will then validate our methods by comparing different statistics for the empirical and simulated data, finding a good agreement between these two datasets.

II. *Determining the user characteristics*

To characterize the mobility pattern of the user base we calculate for each user five parameters based on the original data collected by the mobile carrier.

1. Center of mass position (x_{cm}, y_{cm})

Using the recorded locations of a user, we determine its trajectory's center of mass position (x_{cm}, y_{cm}) as:

$$x_{cm} = \sum_{i=1}^{n} x_i/n,\ y_{cm} = \sum_{i=1}^{n} y_i/n \qquad \text{(S1)}$$

where (x_i, y_i) are the x and y coordinates of the tower routing the i^{th} call (or text message) made by the user in question, n is the number of calls (or text message) recorded.

2. Radius of gyration r_g

The radius of gyration of a user measures the range of distances covered regularly by a user:

$$r_g = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((x_i - x_{cm})^2 + (y_i - y_{cm})^2)} \qquad \text{(S2)}$$

3. Most frequent position

We define a user's most frequent position as the location of the mobile phone tower routing most of its calls (or text message).

4. Principal axis θ

To compare the trajectories of different users we calculate the individual reference frame for each user. We do this by finding the set of axes in which the inertia tensor defined by the collection of points visited by each user takes a diagonal form.

The moment of inertia tensor is given by

$$I=\begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix} \quad \text{(S3)}$$

where $I_{xx}=\sum_{i=1}^{n}(y_i-y_{cm})^2$, $I_{yy}=\sum_{i=1}^{n}(x_i-x_{cm})^2$, $I_{xy}=I_{yx}=-\sum_{i=1}^{n}(x_i-x_{cm})(y_i-y_{cm})$ (S4)

We define the x axis of user's intrinsic reference frame as the eigenvector associated with the smaller eigenvalue of the inertia tensor, looking for a reference frame that satisfies: $I_{x'y'}=I_{y'x'}=0$.

This can be achieved by performing the rotation:

$$x'=x\cos(\theta)+y\sin(\theta),\; y'=-x\sin(\theta)+y\cos(\theta) \quad \text{(S5)}$$

such that $I_{x'y'}=I_{y'x'}=0$.

$$\sum_{i=1}^{n}((x_i-x_{cm})\cos(\theta)+(y_i-y_{cm})\sin(\theta))(-(x_i-x_{cm})\sin(\theta)+(y_i-y_{cm})\cos(\theta))=0 \quad \text{(S6)}$$

$$-\sin(\theta)\cos(\theta)I_{xx}-(\cos^2(\theta)-\sin^2(\theta))I_{xy}+\sin(\theta)\cos(\theta)I_{yy}=0 \quad \text{(S7)}$$

This leads us to three possible solutions:

(A). If $\cos(\theta)\neq 0$ and $I_{xy}\neq 0$ then

$$\tan(\theta)=\frac{I_{xx}-I_{yy}\pm\sqrt{(I_{xx}-I_{yy})^2+4I_{xy}{}^2}}{2I_{xy}} \quad \text{(S8)}$$

We select one of the roots to make sure $I_{x'x'}<I_{y'y'}$.

(B). $I_{xy}\neq 0$ and $\cos(\theta)=0$, then from (S7) we must have $\sin(\theta)=0$, hence there is no valid solution in this case.

(C). $I_{xy}=0$, we derive from (S7) θ=0 or θ=π/2. We select 0 or π/2 according to in which of these angles the momentum of inertia is minimum.

Finally, we make a conditional rotation with π to make sure the most frequent position has a positive value on the horizontal axis.

5. σ_x, σ_y

Finally, we define σ_x and σ_y as a way to characterize the vertical and horizontal spread of user's location in its intrinsic reference frame.

$$\sigma_x = \sqrt{\sum_{i=1}^{n} x'^2_i / n}\ ,\ \sigma_y = \sqrt{\sum_{i=1}^{n} y'^2_i / n} \tag{S9}$$

Here (x'_i, y'_i) are the coordinates of positions in user's intrinsic reference frame. We will use these quantities later to help create a universal pool of possible positions to simulate a user's motion.

III. *Simulating mobility patterns*

1. Initial condition

We start our simulation with each user located at its most frequent position.

2. Deciding when users move

We incorporated the fact that the probability that a user "jumps" to a new location depends on the time of the day by estimating the probability that a user changes position directly from the data. We achieve this using a detailed dataset in which the position of each user is recorded every hour. Some services provided by the mobile phone carrier, like pollen and traffic forecasts, rely on the approximate knowledge of the customer's location at all times of the day. For customers that signed up for location dependent services, the date, time and the closest tower coordinates are recorded on a regular basis, independent of their phone usage. We were provided with 6 days of such records for 1,000 anonymized users, among which we selected the group of 206 users whose coordinates were recorded at every hour during 6 day period, resulting in 23,231 recorded positions. For each pair of consecutive hours, we define $N_T(t)$ as the total number of users for which we have location records in both hours. We then define $N_D(t)$ as the number of records in which a displacement is recorded. Hence, the empirically observed displacement probability is given by $P_D(t)=N_D(t)/N_T(t)$ (Fig. S2), which is properly normalized for each hour by $N_T(t)=N_D(t)+N_{ND}(t)$, where $N_{ND}(t)$ is the number of consecutive records where no displacement was recorded.

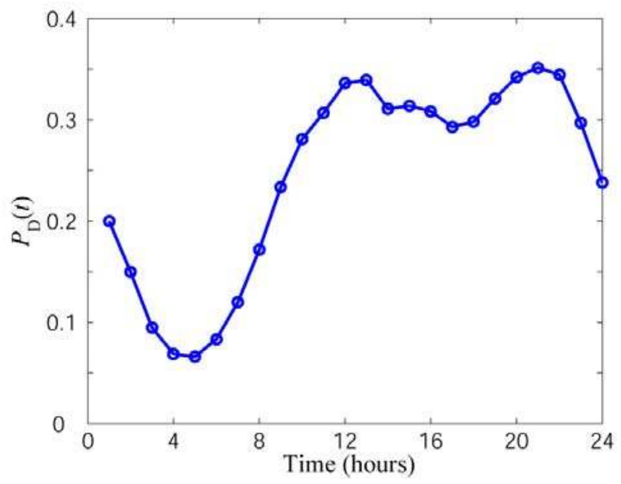


Fig. S2. The moving probability of a user, given by $P_D(t)=N_D(t)/N_T(t)$.

3. Generating motion

To simulate epidemic spreading we need to assign hourly positions to each individual. Because of the limited amount of information available in our dataset, we interpolated the original data for those hours in which location data was not available. Hence, in our epidemic simulation, the sequence of positions describing the motion of a user combines real data and positions simulated according to the algorithm we describe next. For those hours where there is call data available for a given user, we assign the user to the location of the tower routing the call. When more than one call was recorded in a given hour, routed by different towers, we randomly choose one of the recorded positions, as we decided to keep a consistent record in which only one location is assigned to a user in each time period.

For the hours when there is no location data available for a given user, we generate its location according to the following algorithm.

(1) Decide if the user moves

For each user, we decide if it will move or not in a given hour according to the moving probability described above (Fig. S2).

(2) Extract a position seed

Next we extract a position seed for a user. This is done from the pool file generated directly from the empirical data according to the following procedure:

Following the results of Ref. (*2*) we calculated x/σ_x and y/σ_y in user's intrinsic reference frame for 5 million empirical locations. We then stored these values in a file. Pool file was generated before the simulation.

For those hours in which there is no localization information, our simulation begins by randomly choosing a seed from the pool file.

(3) Scaling and rotating user's trajectory

After randomly selecting (a, b) from the (x/σ_x, y/σ_y) pool file we rescale these values as:

$$x' = a\sigma_x, y' = b\sigma_y \tag{S10}$$

We then rotate and re-center these coordinates to move them from the user's intrinsic reference frame to the country's coordinates

$$x = \cos(\theta)x' - \sin(\theta)y' + x_{cm}, y = \sin(\theta)x' + \cos(\theta)y' + y_{cm} \tag{S11}$$

(4) Check if new position is in the country

After re-centering and rotating the coordinates we look whether the newly chosen (x, y) position falls within the country's territory. In the case it dose not, we discard it and extract a new position seed.

(5) Find the nearest tower

After generating a new set of (x, y) position for the user we identify the nearest cell phone tower. If the tower is different from the tower recorded in the user's previous location we assign this to the user as its new position. If it is the same tower we go back to (2) and re-start the process.

IV. *Validation*

In this section we test our methodology by comparing the statistical properties of the simulated trajectories and those observed for the empirical data. In general, we find that our simulation does not introduce considerable biases in the statistical properties of the observed trajectories.

1. Global variables

We begin our analysis by looking at the (x, y) position distribution. As it is expected, we find that there are no significant differences between the real and simulated distributions (Fig. S3, A and B). Next we show the distribution for the scaled variables (x/σ_x, y/σ_y) (Fig. S3, C and D), which again are consistent between the real and simulated data, albeit we note that the simulated data appears to be slightly more isotropic than the empirical one. We continue our validation by looking at the cross-section of the (x/σ_x, y/σ_y) distribution where $y/\sigma_y=0$. Again, we find good agreement between the empirical and real data, albeit we also observe a small tendency for the simulated data to define a slightly more symmetrical distribution than that defined by the real data (Fig. S3E). Finally, we look at the r_g distribution for the real and simulated data, finding a good agreement between both distributions despite a tiny discrepancy at large gyration radius where the empirical distribution tends to be slightly more populated (Fig. S3F). We do not believe that these small deviations will affect our results on the spread of mobile viruses.

2. Local variables

To study the changes introduced by the simulation in the parameters characterizing the user trajectories, we first look at the r_g of the simulated user trajectories as a function of the r_g measured directly from the calling data (Fig. S4A). As expected, the simulation introduces noise into the trajectories which results in a moderate scatter between these variables. Still, both r_g appear to be consistent along the full range. Next we study the principal angle θ of real and simulated trajectories (Fig. S4B) finding in general that the θ angles are either the same or differ by a difference of one or two π. Hence the simulation does not appear to introduce a considerable bias in the direction of a user's main axis of motion. We continue our validation procedure by studying variations in the anisotropy of trajectories introduced by our simulation. We do this by looking at the ratio between σ_y and σ_x for real and simulated trajectories (Fig. S4C), finding that the simulation does introduce some scatter with a small bias to reduce the anisotropy of the most anisotropic trajectories. Finally, we explore the distance between the empirical and simulated center of mass defined by users' trajectories, finding that while in some cases these two quantities differ by as much as the r_g of the user in question, in most cases this distance is about 10% of user's r_g (Fig. S4D).

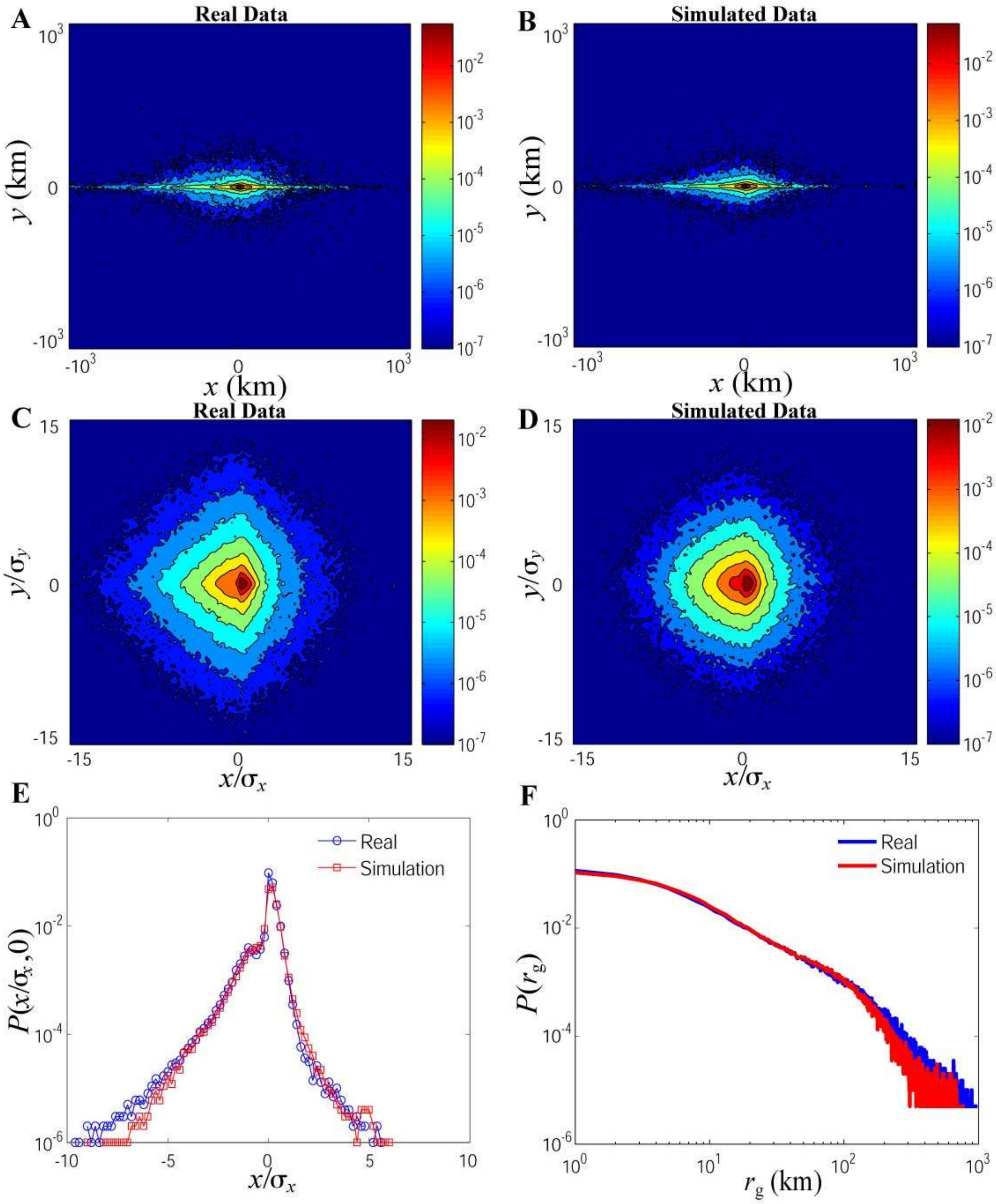


Fig. S3. Comparison of the global parameters for the real and simulated trajectories. (**A**) Distribution of (x/σ_x, y/σ_y) for 10 million recorded locations. (**B**) Simulated distribution of (x/σ_x, y/σ_y) for 10 million recorded locations. (**C**) Same as (A), but for the distribution of (x, y). (**D**) Same as (B), but for the distribution of (x, y). (**E**) The distribution of $P(x/\sigma_x, 0)$. (**F**) Comparison of the real r_g distribution and simulated r_g distribution.

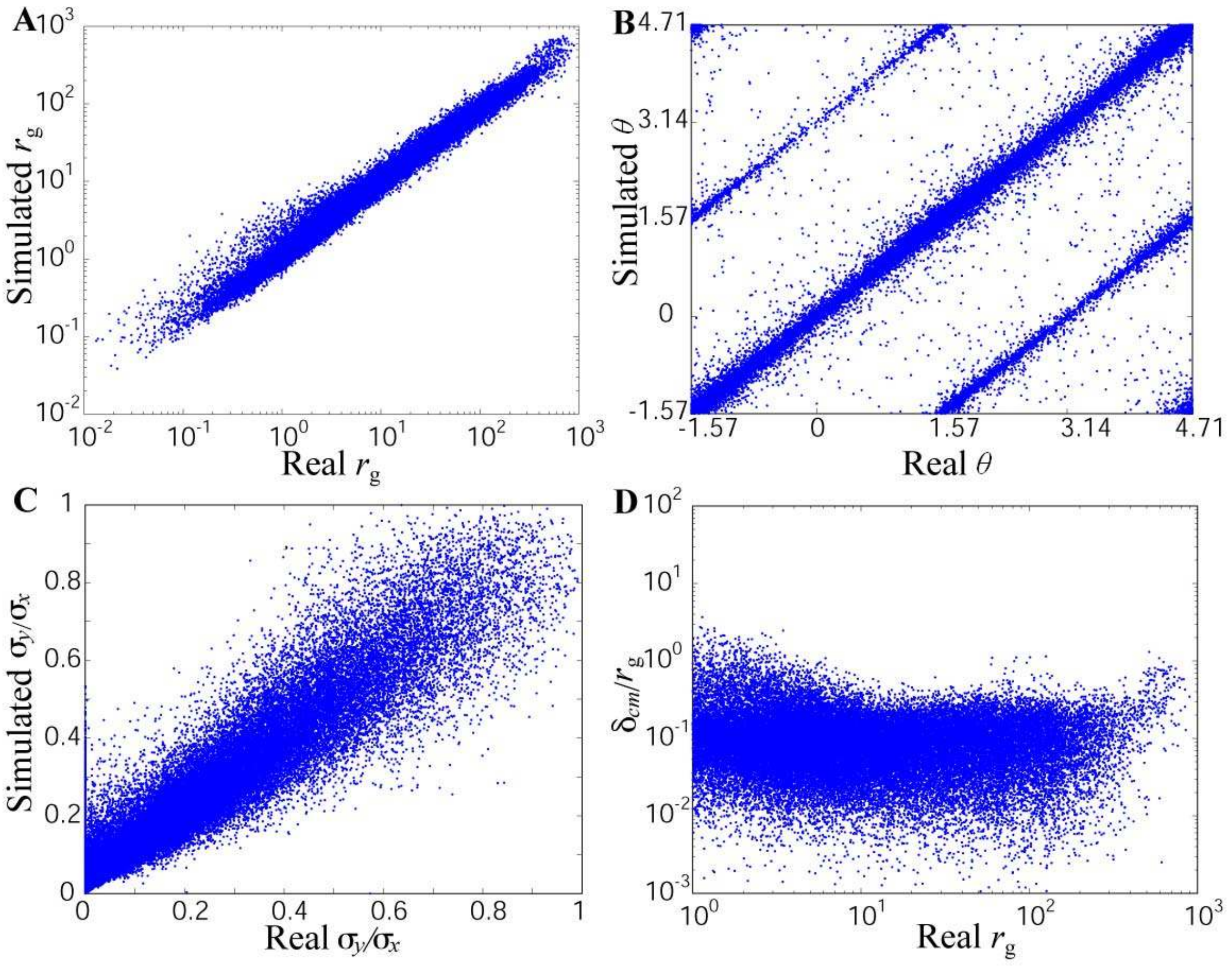


Fig. S4. Comparison of the user specific parameters for 50,000 randomly selected users. (**A**) Real r_g versus simulated r_g. (**B**) Real principal angle versus simulated principal angle for users with $r_g > 0$. (**C**) Real σ_y/σ_x versus simulated σ_y/σ_x for users with $r_g > 0$ and $\sigma_x > 0$. (**D**) The distance between user's real and simulated center of mass positions divided by r_g as a function of user's real r_g for $r_g > 1$km users.

PART 3. VIRUS SPREADING

I. *SI Model*

To simulate the infection within a Voronoi lattice associated with the individual towers we use a compartment model of infection, in which the population is divided into disjoint classes or compartments whose size changes in time. Usually the number of compartments depends on the possible states of an infection. Here we assume that the mobile phones can be in only two possible states: either infected (*I*), when they are transmitting the infection or susceptible (*S*), when they are prone to the infection. In our simulations we are interested in the initial spreading process in the absence of recovery or antiviral software, so we do not consider the possibility that the phones could recover from the infection, a reasonable assumption due to the limited capacity of some handsets for installing antiviral software (*3*), combined with the users' current lack of concern about the threat of mobile phone viruses (*4*, *5*).

In a standard SI model an infected mobile phone can infect a susceptible phone at a rate μ as captured by

$$S + I \xrightarrow{\mu} 2I \tag{S12}$$

Thus the number of infected users (*I*) evolves in time as

$$\frac{dI}{dt} = \beta SI / N \tag{S13}$$

We define the effective infection rate as $\beta=\mu\langle k\rangle$, given by $\mu=1/\tau$, the virus's infection rate, or the inverse of τ (the time that the virus takes to infect a susceptible handset) multiplied by $\langle k\rangle$, which is the average number of contacts each user has. As described in the following sections, we use empirical values of $\langle k\rangle$ that depend on the particular infection process.

Although nowadays many mobile phone viruses require user confirmation to install (*5*), in our simulations we assume that the virus does not need a user confirmation, corresponding to the worst possible spreading scenario. If the virus does need user's confirmation, we need to randomly remove a fraction of users that will not install the virus (will not be infected), which is equivalent with reducing the handset's market share (number of susceptible handsets).

II. *Simulation of the Bluetooth virus infection process*

The empirical dataset offers the user location only with the resolution of the mobile tower's reception area, i.e. we do not know where a user can be found within a given tower's reception Voronoi cell. Therefore, we perform the simulation in two parts, in order to combine the spreading of a virus within each tower area with the users' diffusion between the towers.

Part 1: We run the SI model (S12) for a period of an hour to identify the number of infected users within each tower area. The Bluetooth infection process involves establishing a Bluetooth connection between the infected and susceptible handsets, transferring the virus file via Bluetooth and installing itself on the susceptible handset to start next generation of attack. We use a technically reasonable time of τ=0.5 minute as the time to complete this process. The effective infection rate in (S13) depends on the user's average number of contacts $<k>$. A user's $<k>$ is defined by Bluetooth communication area A and the population density inside a tower's service area:

$$< k >= \rho A = \frac{NA}{A_{\text{tower}}} \tag{S14}$$

We define the Bluetooth communication area as $A=\pi r^2$, where we choose $r=10m$, representing the typical reception range of a Bluetooth device. The population density inside a tower's service area is $\rho=N/A_{\text{tower}}$, where A_{tower} is the tower's service area, approximated by the area of the Voronoi cell surrounding the respective tower and N is the number of users within that area. Because the population density in each tower's service area varies widely (Fig. S8, C and D), the average number of contacts per user also varies widely from tower to tower, making the effective infection rate larger in areas with higher population density.

Each simulation starts with one randomly chosen infected user at time t=0, where we set time t=0 as midnight in the first day of observation and the simulation's unitary time step is defined by the time τ that the Bluetooth virus takes to infect a susceptible phone, i.e. 0.5 minute. Hence, the infection rate μ=1 at each simulation time step. At each step, we calculate how many susceptible users are infected in each tower's service area. For each user, $<k>I/N$ represents the probability that an infected user is within the Bluetooth range (if $<k>$ is larger than 1, we set $<k>$=1, because an infected handset can only establish connection with one handset at each time step), thus the number of susceptible users infected in a time step is:

$$\begin{cases} \mu S < k > I / N, & < k >< 1 \\ \mu S I / N, & < k > \geq 1 \end{cases} \qquad \text{(S15)}$$

Part 2: We use the mobility patterns discussed in Section 2.III to identify the location of the users at each consecutive hour. Note that the number of users in each Voronoi cell changes from hour to hour, consistent with the user mobility patterns. For example, while the business district has many users during business hours, it decreases significantly during the night.

By iterating the procedures described in parts 1 and 2, we can capture the spreading of the virus across the entire mobile population.

III. *Simulation of the MMS virus infection process*

Contrarily to Bluetooth viruses, which spread by spatial proximity, a handset infected by an MMS virus typically sends a copy of itself to each contact (mobile phone number) found in the handset's phone book, i.e. the virus spreads along the links of the underlying social network, and spatial proximity is not required. To simulate the spreading of an MMS virus, we approximate a user's phone book with the list of numbers the user communicated with during one month of observation. We assign the virus to a randomly chosen handset, and it will send MMS to all its identified contacts. Newly infected users proceed to infect their contacts, so the virus spreads by an iterative process. The MMS service is not an instantaneous service, and there is some time delay on receiving a MMS. This delay depends on the mobile phone company's servers and the size of the multimedia files. In our simulation we choose τ=2 minutes as the time required for a MMS virus to be received by another handset and to install itself. We choose each simulation time step as 2 minutes, and set μ=1 and $<k>$=1. That is, once infected a mobile phone can infect sequentially all its $K_{\max}$ contacts. This scenario is more conservative than infecting all $K_{\max}$ contacts in each time step. Here we also include a plot of the degree distribution with the mean degree as 6.08 and variance as 36.68 (Fig. S5).

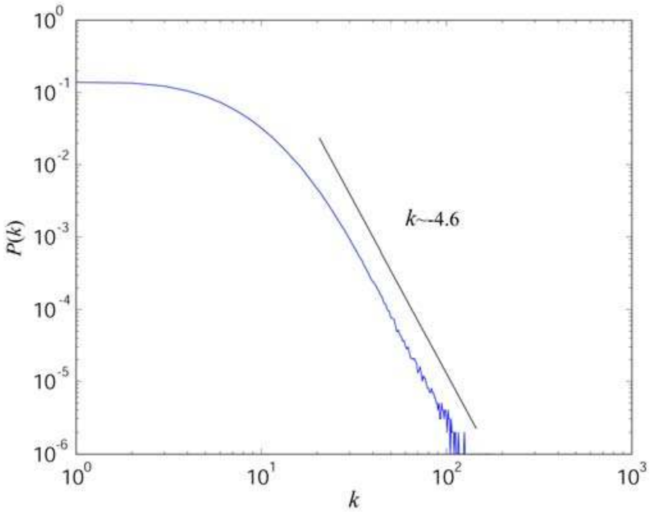


Fig. S5. Degree distribution of the MMS network (Observational period 1 month).

IV. *Simulation of the hybrid virus infection process*

For hybrid viruses we consider the combined communication via Bluetooth and MMS, and study the scenario in which a malware can transmit via both protocols. At each time step, if a handset is susceptible, it can be infected by receiving an MMS virus from an infected contact or by another user within the Bluetooth range, and after it is infected, it will search the susceptible targets nearby and send MMS with the virus to the users in its phonebook.

V. *Estimating m_c and m* (the real market share)

In the manuscript we estimated m_c based on the number of phone numbers each user called during the previous month, finding it to be approximately m_c=0.095, i.e. roughly 10% of the market. This, naturally, represents only an approximation. For example, if we increase the observational period from a few days to three months, m_c decreases to m_c=0.051 (see Fig. S6). The question is, what the real value of m_c, and how does it compare to the market share of the largest OS?

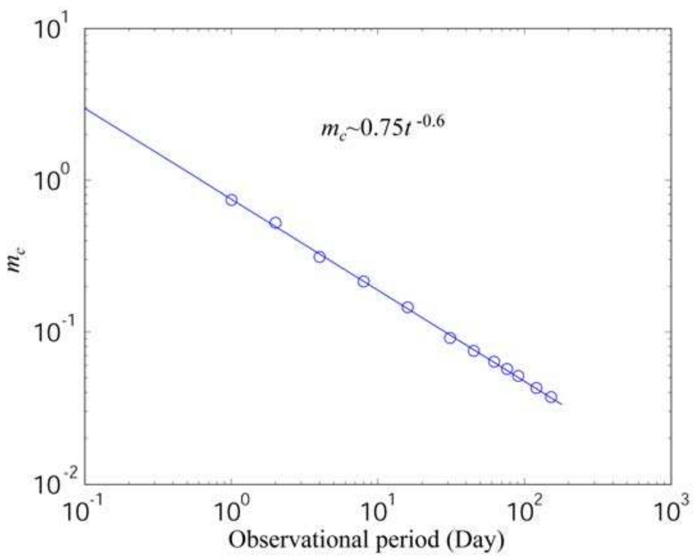


Fig. S6. m_c vs observational period.

The real value of m_c is determined by the number of phone numbers stored in a user's contact list or phone book—which can be much larger or smaller than the numbers the user calls in a one month period. Several factors act to decrease m_c. For example, here we considered only the degree based on calls within the same phone carrier, which has about ¼ market share in the particular country. Considering the full user base could further decrease m_c. Other effects may increase m_c, however, such as the viruses' limited ability to search the phonebook, which would decrease the average degree $<k>$, as it may not successfully find all relevant numbers. Here, without considering the complex components described above, we calculate the m_c by assuming the observational period as 1 year, comparable to a mobile phone's average life time.

Another important question is related to the need to estimate if the market share of any operating system is close to m_c. In this section, we discuss different factors that contribute to the fragmentation of the smart phone market and show how these relate to the percolation threshold determined by the structure of the mobile phone network.

We have identified five factors driving the fragmentation of the smart phone market.

(i): *The fraction of all mobile phones that are smart phones.* It is estimated that 5% of all mobile phones are smart phones (*3*), giving us an upper bound to the total market share that a smart phone Operating System (OS) can reach.

(ii): *Smart phones use a variety of different OS.* Currently the Symbian S60 platform is the most common OS variant in operation (*6*), its market share being larger than 40% among smart phones, which translates into a market share of approximately 2% of all mobile phones.

(iii): *There are several versions of each OS.* The most common OS, the Symbian, has many different versions (OS6.0 to OS9.3) and Cabir (*7*), a virus infecting smart phones with Symbian S60 platform can infect phones running two versions of this OS (OS6.0 and OS7.0, less than 7% of Symbian OS) (*8*). Another virus called Commwarrior was able to infect only phones running Symbian S60 2.x version. New versions of an OS are likely to be protected against the vulnerability exploited by these viruses. Hence, different OS versions can further fragment the market share of phones susceptible to a given virus.

(iv): *Viruses that spread through Bluetooth, can only infect phones that have enabled the Bluetooth discovery option.* This makes phones that are not open for Bluetooth connection discovery unavailable for the spread of these viruses. In principle it is possible to design viruses that can spread to undiscoverable devices, yet current viruses do not posses this feature.

(v): *All known mobile phone viruses require user installation.* While viruses can autonomously copy themselves from one device to another, to finish the infection process, users need to install them. There are several ways in which virus designers trick users to install them, however, many of which having been used in the past.

V. Comparison between hybrid virus and Bluetooth/MMS virus

We also made additional simulation to compare the spatial spreading pattern of a hybrid virus to the spreading pattern defined by Bluetooth and MMS viruses (Fig. S7) confirming that the spatial spreading pattern defined by hybrid viruses represents a combination of the spreading patterns defined by Bluetooth and MMS viruses.

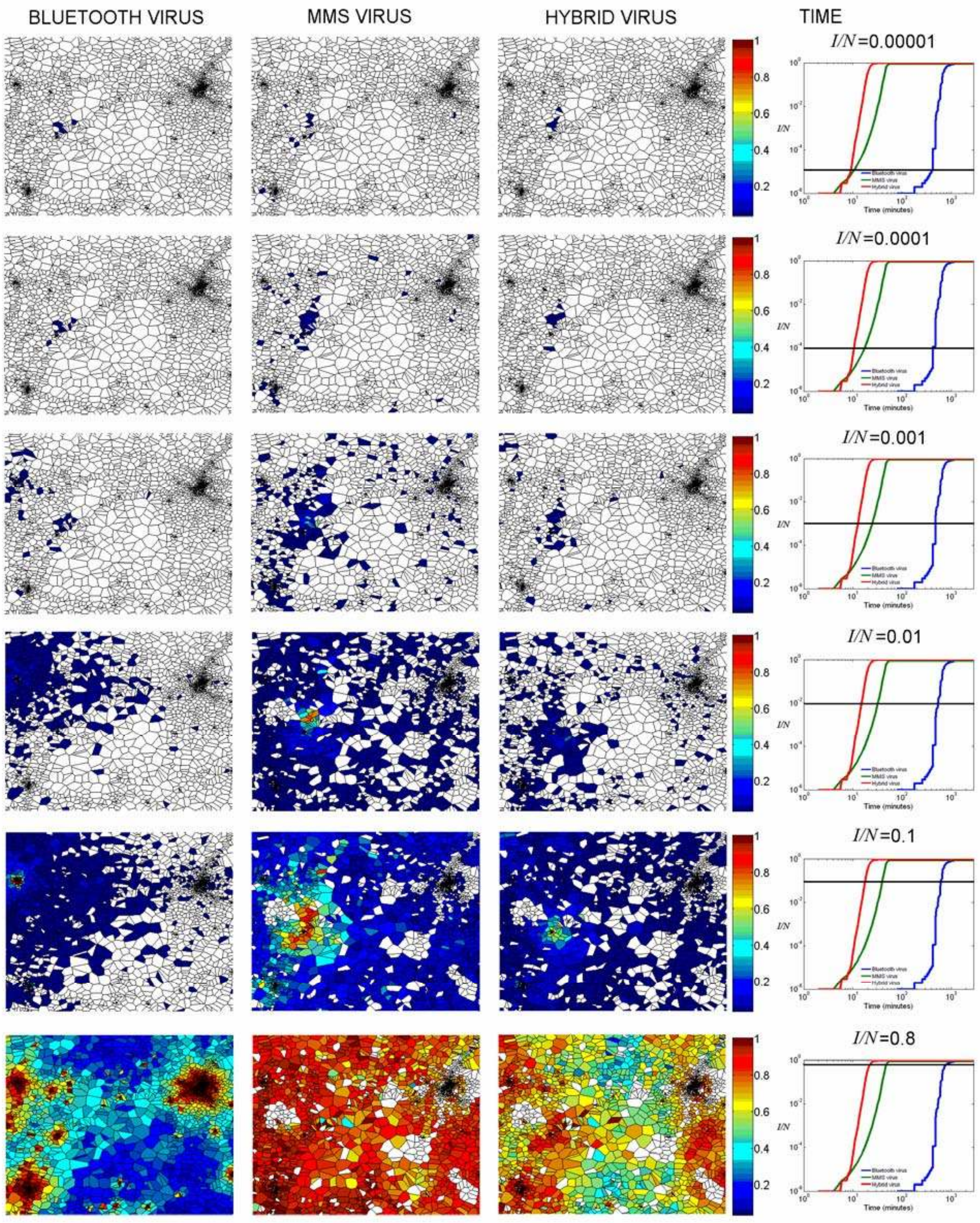


Fig. S7. Spatial spreading pattern defined by Bluetooth, MMS and hybrid viruses.

PART 4. Calculations

I. *The latency time preceding a Bluetooth virus outbreak*

We define the latency time for a Bluetooth virus as the time when the fraction of the infected users is larger than 1%. The latency time depends on the population density (Fig. S8, C and D) where the virus is initially located. The average latency time (samples from 988 towers' service areas) decreases with the population density (Fig. S8B), indicating that Bluetooth virus spreads faster if it starts in a place with large concentration of users. The latency time can be more than two weeks if the Bluetooth virus is released in a very low population density area and it is relative stable when the population density is larger than 1000 persons/km^2 (Fig. S8B). In our simulation for Bluetooth and hybrid virus's spread, to prevent the long latency time caused by virus starting at a low population neighborhood, we randomly select users from tower areas with population density larger than 200 persons/km^2 as the virus sources, which is reasonable in the real case (it is unlikely that a new virus will be released in a remote area).

II. *Theoretical calculation of the giant cluster size*

From Ref. (*9*), $F_0(x) = \sum_{k=0}^{\infty} p_k q_k x^k$, $F_1(x) = \dfrac{\sum_k k p_k q_k x^{k-1}}{\sum_k k p_k}$ (S16)

Where p_k is the probability that a randomly chosen vertex has degree k, and q_k is the probability that a vertex is occupied given that it has degree k, in our case $q_k = m$.

From Ref. (*9*), the giant component size: $S = F_0(1) - F_0(u)$ (S17)

with $u = 1 - F_1(1) + F_1(u)$ (S18)

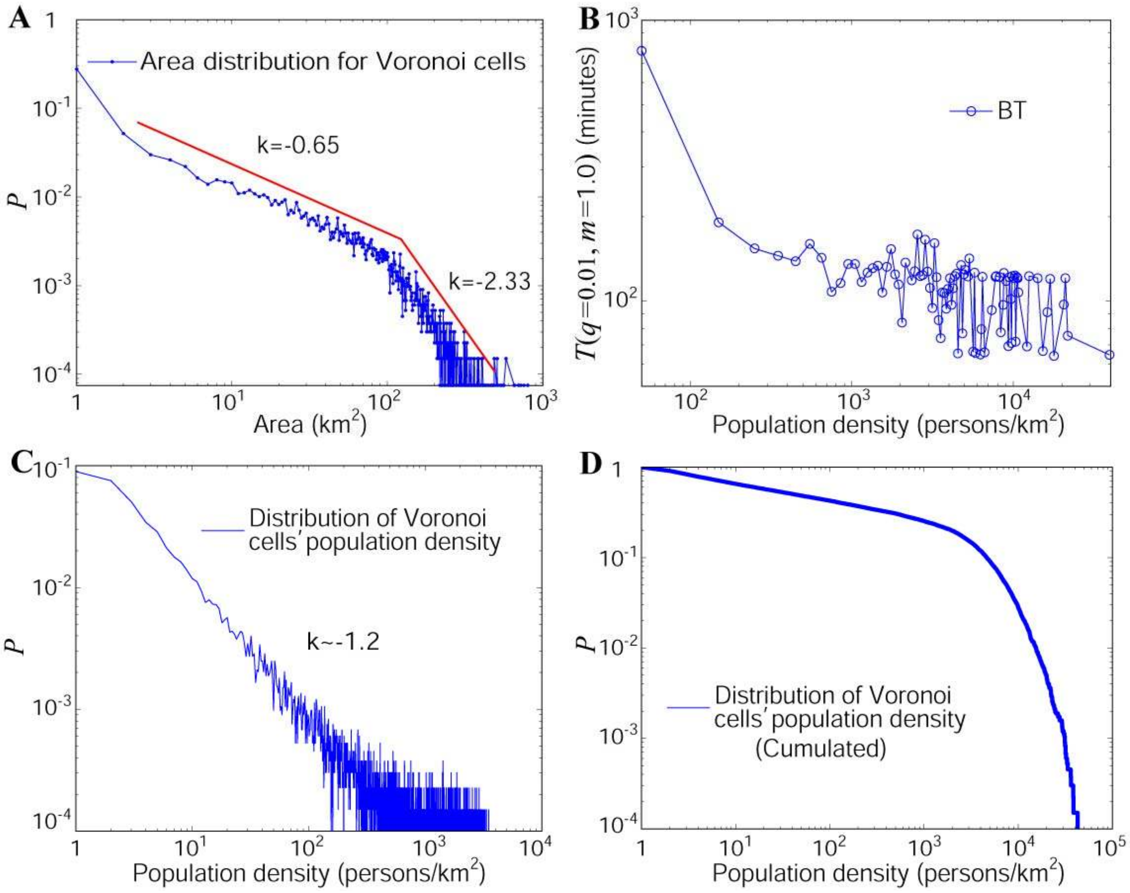


Fig. S8. The distribution of towers' service areas and Bluetooth virus's latency time. (**A**) The distribution of towers' service areas. (**B**) The average latency time (when *I/N*>1%) as a function of population density where the virus is initially located. (**C**) The distribution of population density in each tower's service area. (**D**) Cumulated population density distribution.

III. <u>*Statistical analysis*</u>

The purpose of this section is to support our findings with rigorous goodness-of-fit analysis. Three figures in the original paper involved fitting process: Fig. 2D, E, and F. For each set of data we present parametric models that give the best fit of the data with 95% confidence interval. Using standard curve fitting analysis, we evaluate goodness-of-fit statistics for parametric models calculating the sum of squares due to error (SSE), the R-square and the root mean squared error (RMSE). In each case we report the best fit, given by the minimum SSE. The parameters selected for each fit and the goodness-of-fit statistics are shown in Tables S1, S2 and S3.

Note, however, that the error bars provided by the statistical methods are unreasonably small as they represent only fitting errors, and not the potential systematic errors in the data. Therefore in the manuscript we report the exponents only up to two digits.

Fit evaluation for Fig. 2D

BT	Model	*a*	*b*	SSE	R-square	RMSE
$T_{0.15}$	$\log_{10}(T) = a\log_{10}(m) + b$	-0.5953 ± 0.0364	2.304 ± 0.039	0.001217	0.9944	0.01233
$T_{0.65}$	$\log_{10}(T) = a\log_{10}(m) + b$	-0.633 ± 0.0452	2.581 ± 0.048	0.0008072	0.9924	0.01004

Table S1. Fitting parameters for Fig. 2D.

Fit evaluation for Fig. 2F

L	Model	*a*	*b*	SSE	R-square	RMSE
L_{ave}	$\log_{10}(L) = a\log_{10}(m - m^*) + b$	-0.1887 ± 0.017	1.13 ± 0.022	6.973e-005	0.984	0.002641
$L_{\max}$	$\log_{10}(L) = a\log_{10}(m - m^*) + b$	-0.1951 ± 0.0205	1.457 ± 0.025	6.939e-005	0.9782	0.002634

Table S2. Fitting parameters for Fig. 2F.

Fit evaluation for Fig. 2E

MMS	Model	*a*	*b*	SSE	R-square	RMSE
$T_{0.05}$	$T = a(m - 0.0945)^b$	45.66 ± 4.92	-0.1956 ± 0.0159	5908	0.9499	14.03
$T_{0.15}$	$T = a(m - 0.1280)^b$	44.46 ± 1.93	-0.1699 ± 0.0059	623.5	0.9933	4.719
$T_{0.30}$	$T = a(m - 0.1777)^b$	46.26 ± 3.29	-0.1424 ± 0.0109	1337	0.9641	7.313

Table S3. Fitting parameters for Fig. 2E.

The statistical analysis indicates a systematic dependence of the exponent controlling the divergence of $T(q,m)$ near the critical point on the value of q. Taken on face value, this suggests that the longest minimal path length alone does not control the divergence near the critical point. However, this conclusion would be premature based on the current data—one would probably require a detailed analysis, focusing on model systems, to reach a definitive conclusion.

REFERENCES

1. Fu, T., Yin, X. & Zhang, Y. Voronoi Algorithm Model and the Realization of Its Program. *Computer Simulation* 23, 89-91 (2006).

2. González, M.C., Hidalgo, C.A. & Barabási, A.-L. Understanding Individual Human Mobility Patterns. *Nature* 435, 779-782 (2008).

3. Cheng, J., Wong, S., Yang, H. & Lu, S. SmartSiren: Virus Detection and Alert for Smartphones. *Proc. 5th ACM Int. Conf. Mob. Syst. Appl. Serv.* (ACM, New York, 2007)

4. Hypponen M. Maleware goes mobile. *Scientific American* Nov. 70-77 (2006).

5. Shevchenko. A. An overview of mobile device security. www.viruslist.com (2005).

6. http://www.s60.com/life/thisiss60/S60forbusiness/keyfigures

7. http://vil.nai.com/vil/content/v_130678.htm

8. http://stats.getjar.com/statistics/world/platform_symbian

9. Callaway, D.S., Newman, M.E.J., Strogatz, S.H. & Watts, D.J. Network Robustness and Fragility: Percolation on Random Graphs. *Phys. Rev. Lett.* 85, 5468 (2000).