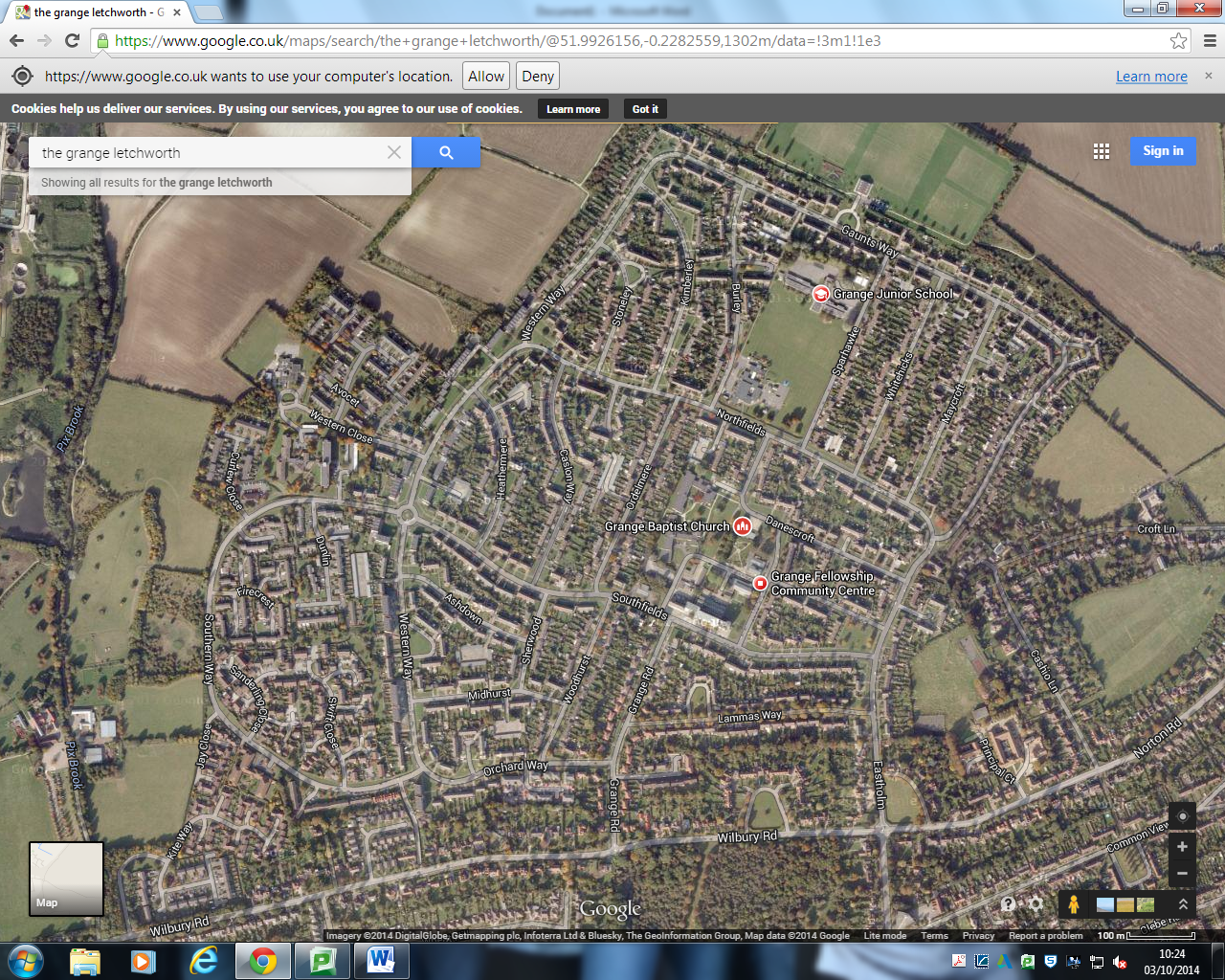
**Police Search**

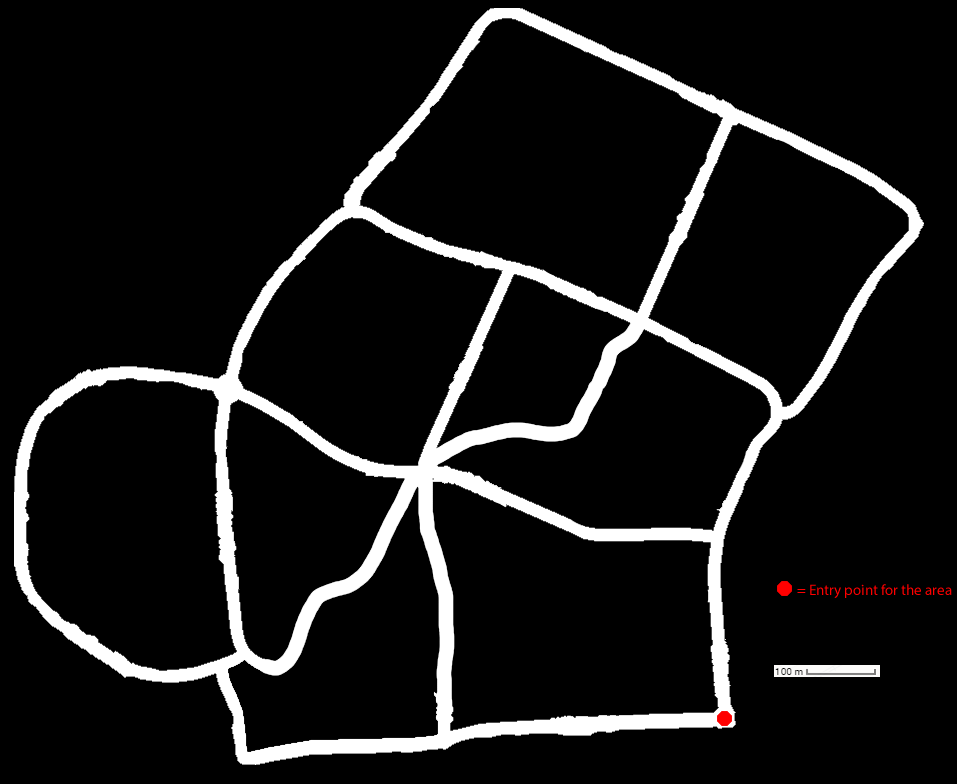
In recent years, the crime rate of the area known in garden city that is Letchworth in Hertfordshire as The Grange, has grown. Police respond often to calls for that particular part of the town and tend to deal with young people who have escaped from committing a crime in the main town and now hide in the streets of The Grange. This means police have to patrol all the streets of The Grange as quickly and as efficiently as possible to give the best odds of catching the escaped criminal.

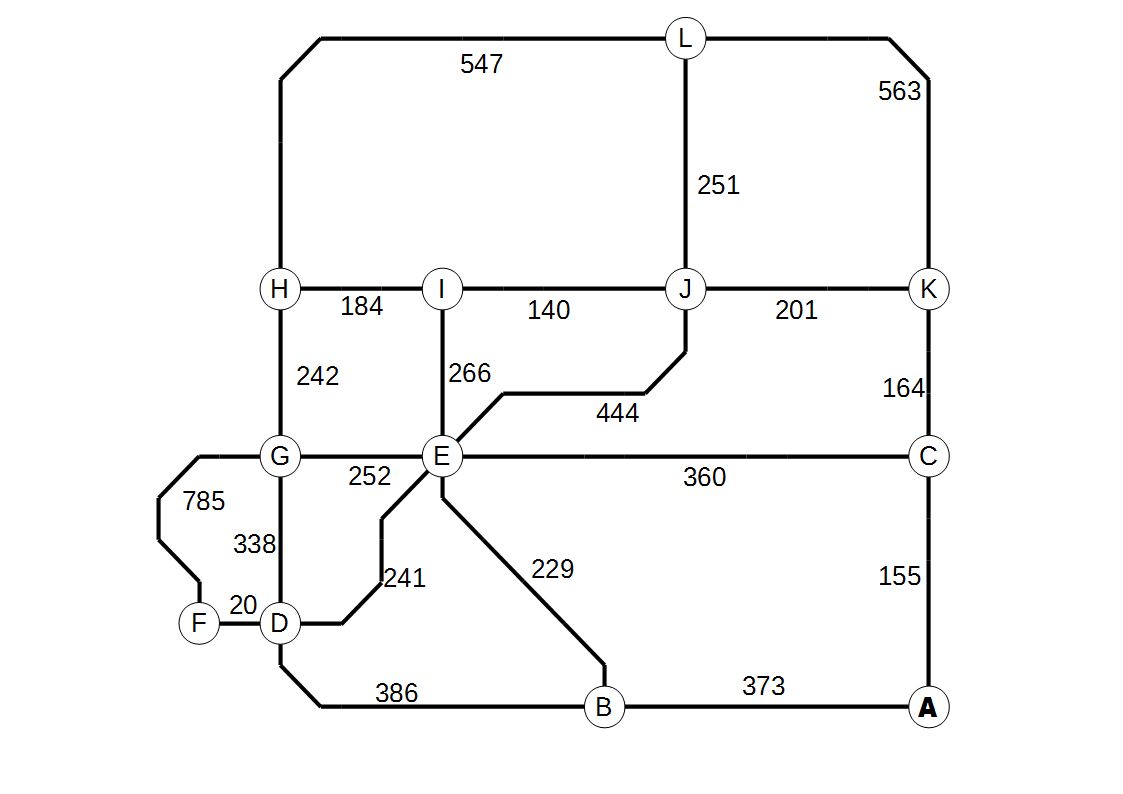
Here is a map of The Grange taken from Google Maps:



The area has many narrow streets and dead-ends that make it difficult for a search patrol to navigate, so they simplify do not search some of the streets. This is remedied by using the police budget efficiently and placed CCTV cameras in areas police find difficult to get to, meaning that perpetrators cannot easily escape.

Here is a monochrome version of the area, with the roads and streets that police would not travel down while searching removed.





I’ve walked down all the streets with a measure wheel and recorded the distance between each relevant point, which is generally a turning. Here is a graph representing my findings. Each node is what I consider a relevant point.

The numbers represent single metres. Each arc is a street and each node is where you can potentially go from one street to another.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** |
| **A** |  | 373 | 155 |  |  |  |  |  |  |  |  |  |
| **B** | 373 |  |  | 386 | 229 |  |  |  |  |  |  |  |
| **C** | 155 |  |  |  | 360 |  |  |  |  |  | 164 |  |
| **D** |  | 386 |  |  | 241 | 20 | 338 |  |  |  |  |  |
| **E** |  | 229 | 360 | 241 |  |  | 252 |  | 266 | 444 |  |  |
| **F** |  |  |  | 20 |  |  | 785 |  |  |  |  |  |
| **G** |  |  | 338 |  | 252 | 785 |  | 242 |  |  |  |  |
| **H** |  |  |  |  |  |  | 242 |  | 184 |  |  | 547 |
| **I** |  |  |  |  | 266 |  |  | 184 |  | 140 |  |  |
| **J** |  |  |  |  | 444 |  |  |  | 140 |  | 201 | 251 |
| **K** |  |  | 164 |  |  |  |  |  |  | 201 |  | 563 |
| **L** |  |  |  |  |  |  |  | 547 |  | 251 | 563 |  |

The total weight of the graph is 6141.

The data I’ve obtained was collected in person by myself, so there may be some unknown errors or erroneous values that have been gained due to human error. When I check my data against what Google Maps has recorded, I see that I am not that far out with my values, being only 15 metres at most out.

Since the police’s duty is to travel down each of the arcs of the above graph, this requires the use of the Chinese Postman algorithm. This algorithm will allow us to figure out what is the most efficient way to traverse the neighbourhood.

To perform the algorithm, one must first identify all the nodes that have an odd amount of arcs connecting it. The nodes B, C, H, I, K and L all happen to have exactly 3 arcs connecting to them.

Then one must ‘pair up’ each of the nodes to another node. When you pair two nodes together, you obtain the smallest possible single direction traversable route from that node to the other. You then add all these pairings together. From all the potential node pairs, you select the lowest value made from this process and add it on to the total weight of the graph to find the length of an optimal ‘Chinese Postman’ route.

Here are my calculations performed to find the optimal Postman route for my graph.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Node Pairng #1** | **Weight** | **Node Pairing #2** | **Weight** | **Node Pairing #3** | **Weight** | **Total Pairings Weight** |
| BC | 528 | HI | 184 | KL | 452 | 1164 |
| BC | 528 | HK | 525 | IL | 391 | 1444 |
| BC | 528 | HL | 547 | IK | 341 | 1416 |
| BH | 679 | CI | 505 | KL | 452 | 1636 |
| BH | 679 | CK | 164 | IL | 391 | 1234 |
| BH | 679 | CL | 616 | IK | 341 | 1636 |
| BI | 495 | CH | 689 | KL | 452 | 1636 |
| BI | 495 | CK | 164 | HL | 547 | 1206 |
| BI | 495 | CL | 616 | HK | 525 | 1636 |
| BK | 692 | CI | 505 | HL | 547 | 1744 |
| BK | 692 | CH | 689 | IL | 391 | 1772 |
| BK | 692 | CL | 616 | HI | 184 | 1492 |
| BL | 886 | CI | 505 | HK | 525 | 1916 |
| BL | 886 | CH | 689 | IK | 341 | 1916 |
| BL | 886 | CK | 164 | HI | 184 | 1234 |

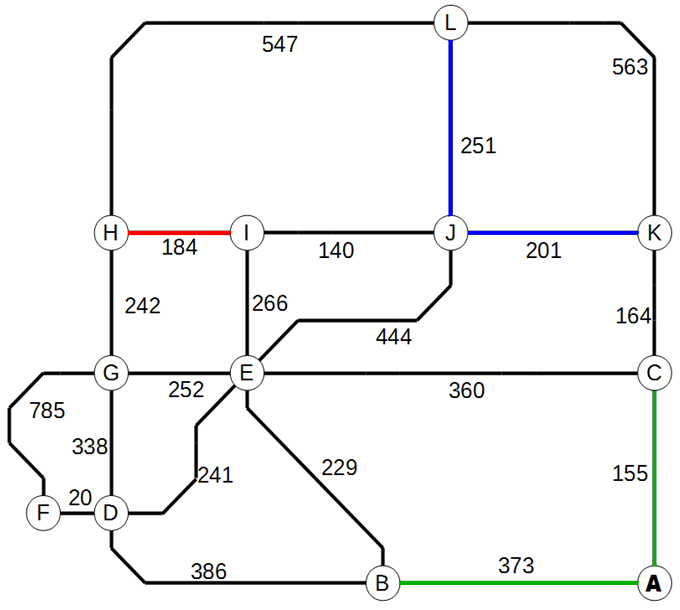
I’ve split the 15 results into 5 groups, based on the first node pairing, so that I can see what is the smallest total pairing is, which is what the black arrows are pointing at.. Here are those from the 5 groups:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Node Pairng #1** | **Weight** | **Node Pairing #2** | **Weight** | **Node Pairing #3** | **Weight** | **Total Pairings Weight** |
| *BC* | *528* | *HI* | *184* | *KL* | *452* | ***1164*** |
| BH | 679 | CK | 164 | IL | 391 | 1234 |
| BI | 495 | CK | 164 | HL | 547 | 1206 |
| BK | 692 | CL | 616 | HI | 184 | 1492 |
| BL | 886 | CK | 164 | HI | 184 | 1234 |

From this table, we can clearly see that the pairings of {BC,HI,KL} have the smallest total weight, with it being 1164. To get the final result for the algorithm, we add that result onto the total weight of the graph (6141).

1164 + 6141 = 7305

So, from this, I have calculated that the length of the optimal Postman route for the police search is 7305 metres.

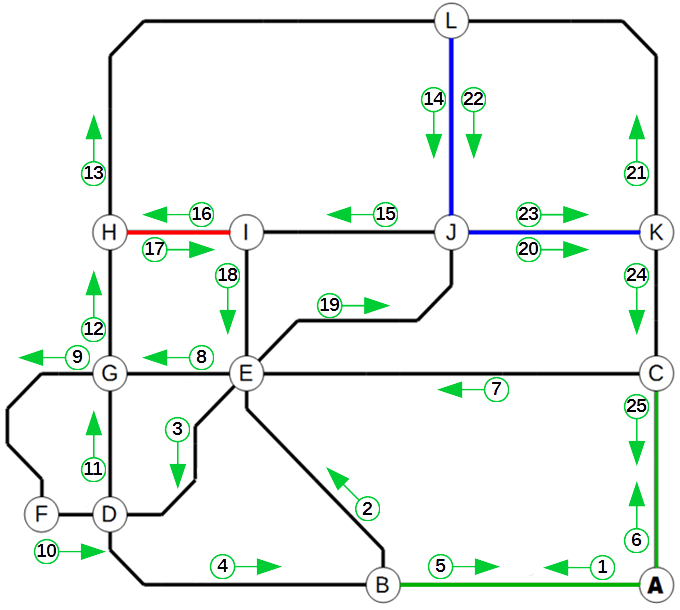
The pairings {BC,HI,KL} take the following paths through nodes to create their respective pairings :

BC = B 🡨🡪 A 🡨🡪 C

HI = H 🡨🡪 I

KL = K 🡨🡪 J 🡨🡪 L

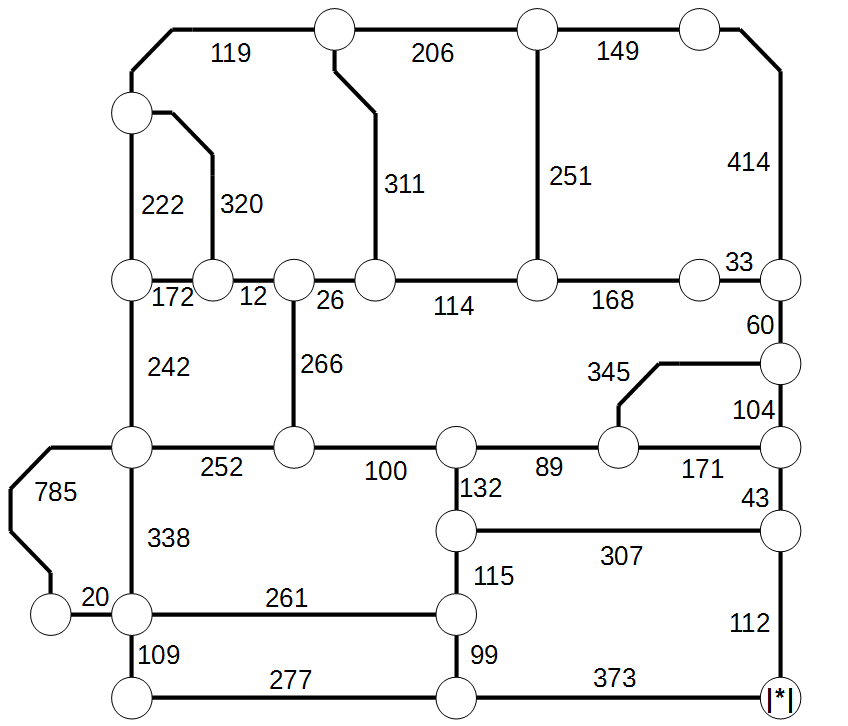
This means these arcs are required to be traversed twice when performing the optimal route.



Here is a graphical representation of what the optimal route could be. By starting at node **A** (the patrol entrance), then going to node B, and so following the numbered directions, you would perform an optimal route.

In text form, this optimal route is:

**A** 🡪 B 🡪 E 🡪 D 🡪 B 🡪 **A** 🡪 C 🡪 E 🡪 G 🡪F 🡪 D 🡪 G 🡪 H 🡪 L 🡪 J 🡪 I 🡪 H 🡪 I 🡪 E 🡪 J 🡪 K 🡪L 🡪 J 🡪 K 🡪 C 🡪 **A**



As I mentioned before, search patrols already have the ordeal of locating an escapee on the streets, let alone navigating them. As a result, they can’t fully search every single street. Here is a diagram showing what would be a optimistically complex graph of the same area. The version you see above contains the streets that the police actually do search, and simplified down arcs. This graph is what the police would prefer to actually follow, but its far too complex and requires too many “Arc to Node transitions”, where every time the patrol reaches a new node, they need to slow down and turn, depending on their route. This kind of action doesn’t happen enough on the graph on the above page to be noticeable, but on a graph like this, it happens far too much and would create a negative impact on the patrol.

This graph also contains too many “odd” nodes to be able to calculate what the optimal Chinese Postman route. Since you pair up every odd node in one instance, that would mean you would have so many potential pairings, the computation required would be astronomical.