[https://rosettacode.org/wiki/Knight%27s\_tour#Ant\_colony](https://rosettacode.org/wiki/Knight%27s_tour" \l "Ant_colony) – prints the pattern

<https://github.com/soniakeys/knight/blob/master/knight.go> - prints analysis

<https://www.semanticscholar.org/paper/Ant-Colonies-Discover-Knight's-Tours-Hingston-Kendall/f18e2a4c545375c74248a15681ef77c5a4edc001> - explains ACO

<https://www.mayhematics.com/t/1n.htm> - explains knight tour problem

r,c = square in the chess board of dimensions n \* n

tch[r, c] is a list of squares an ant a has already visited.

Step 1: Start

Step 2: Initialise the chessboard: each T r,c,k = 10 ^ -6 ; where k = edges corresponding to legal moves

Step 3:

For each cycle

Evaporate pheromones: T r,c,k = (1 – p) \* T r,c,k ; 0 < p < 1 and p = evaporation rate

For each starting square

Start an ant: list tch[r, c]

While not finished

Choose next move

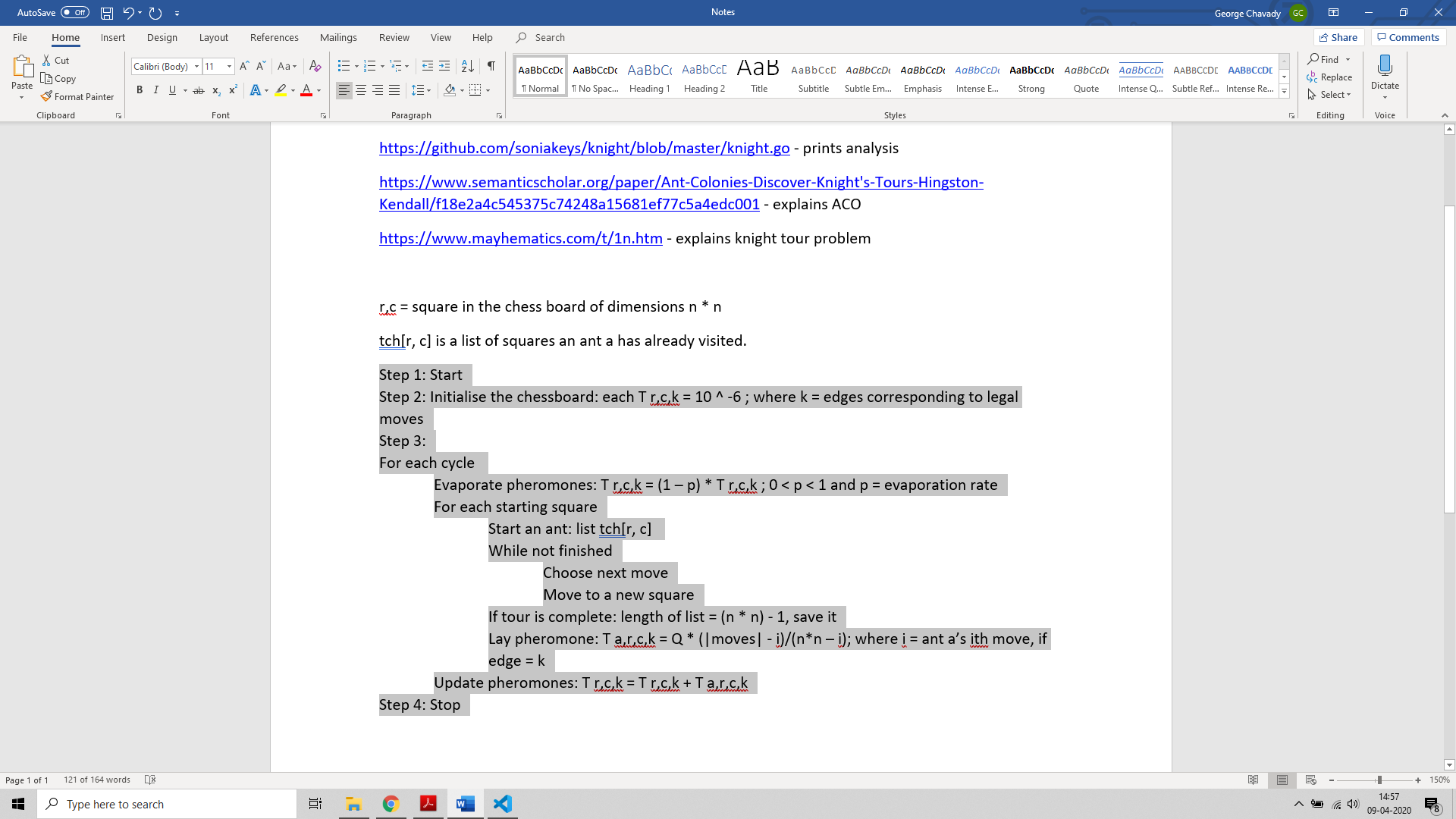
Move to a new square

If tour is complete: length of list = (n \* n) - 1, save it

Lay pheromone: T a,r,c,k = Q \* (|moves| - i)/(n\*n – i); where i = ant a’s ith move, if edge = k

Update pheromones: T r,c,k = T r,c,k + T a,r,c,k

Step 4: Stop



Pros:

* Higher chance of obtaining the shortest path by choosing edges with more pheromone.
* An ACO algorithm for generates about 0.076 tours per attempt when compared to a naive depth first search which yields only 0.000003 tours per attempt.
* Starting of ants from all squares facilitates information sharing between ants. That is, an ant starting on one square can utilise the knowledge gained by ants starting on more remote squares.

Cons:

* Difficult for theoretical analysis.
* Uncertain time to convergence.
* Method is experimental rather than theoretical.