# 1 Introduction

## 1.1 Terminology

Graph G = (V, E), graph define the map (area, zone), and A - agents ($a_i$).
Vertices - all possible positions.
Edges - connection between two positions, there is no more then one edge connecting between two vertices.
Each edge has a time and threat value.
Time defined $t > 0$; Threat defined $0 < p < 1$ where 1 is destruction and 0 is no threats at all.

Agents n agents $(a_0, ..., a_n)$ each agent has start and target point $(a_i, s_i, t_i)$.
No two agents can be in the same vertex at the same time, the destruction of an agent causes its current position (vertex) to be cancelled (The vertex becomes obstacle).
Path - $(s_i, v_i, ..., v_j, t_i)$ represent all the vertices agent traverse between start and target positions.
There is a possibility the agent will stop moving for one step or more, represented as the same vertex number of times, $(..., v_i, v_i, v_i, ...)$.

## 1.2 Problem definition

MAAPF (Multiple Agent Adversarial path finding) is very similar to the original problem of MAPF with one change.
The path between positions are under threat.
Unlike MAPF MAAPF add to each edge an element of probability p.
In the final solution every agent has some probability between 0 to 1 to be dysfunction and become an obstacle and so removing its position (vertex) from the map (graph).
The new algorithm will need to compensate this major uncertainty and provide ways to dynamically overcome that situation.

## 1.3 Input

Graph and agents (G, A).
Target function - The function use to approximate a value from any vertex to the target vertex of specific agent.
$f : (v, t) \rightarrow R$.
The value returned (score) will be use to pick the next edge in every step.

## 1.4 Target

Find P non-conflicts path for each agent from start to target position.
The characteristics of a path is time and survivability.
Both need to be minimal as possible.