# A*mbush family: A* Variations for Ambush Behaviour And Path Diversity Generation

Kelwin Fernández, Glebys González, and Carolina Chang

Grupo de Inteligencia Artificial,
Universidad Simón Bolívar, Venezuela
kelwin@gia.usb.ve, glebys@gia.usb.ve, cchang@ldc.usb.ve
http://www.gia.usb.ve

**Abstract.** In the area of Artificial Intelligence for Videogames, A* is a commonly proposed algorithm for path-finding. Even though this algorithm guaranties optimality, it tends to bring forth similar behaviours in agents that are close to each other. On the other hand, when the agents are sparsely distributed, the algorithm doesnt secure an attack that comes from different places. Having the generation of ambush behaviours and diversity of paths as the goal, A*mbush, P-A*mbush, R-A*mbush and SAR-A*mbush are proposed. They are modification of A* that considers the amount of agents that have a specific node (or edge) in their calculated path, when its computing the cost function of that point in the graph. P-A*mbush, R-A*mbush and SAR-A*mbush are variations of A*mbush which tries to improve the paths generated by the proposed A*mbush algorithm. *abstract* environment.

**Key words:** ambush, pathfinding, A*, group strategies

## 1 Introduction

Generating agents with intelligent looking behaviour has been a constant challenge in the area of Artificial Intelligence for Video games [1]. Among this conducts, the user expects to appreciate agents than can perform tactical movements and group strategies. These tend to be complex in their implementation and usually result in preestablished moves that can be easily identified by the user, after several runs of the game. Having agents execute pathfinding towards a common point, is a problem that frequently appears in this area. The goal point is usually given by a location in the game map (potentially the opponent's position). A well known and used scheme to attack this problem is the generation of the minimal path towards the objective [6, 4]. This path is generated using the game map. When the algorithm is executed, it is very likely for the paths to be confluent. Thus, route diversity and map exploration is avoided.

When the minimal path strategy is applied for chasing the enemy, many escape paths are left open. Therefore, it is of special interest to generate mechanisms of route diversification that can produce ambush behaviours.

The techniques explained in this paper are very adaptable to other various contexts where, leaving the ambush aside, it is important to produce path diversity, with the purpose of avoiding saturation in certain sectors of the underlying graph. Examples of the latter are traffic managers, digital or physical package routing [8], robotics, among many others.

## 2 Formal Problem Definition

Let $G = (V, E)$ be a graph (directed or undirected), where $V$ is the set of nodes and $E$ is the set of edges.

Let $A$ be a set of agents that want to reach a point $t \in V$. Every agent $i \in A$, is located in a node of the graph. Let $pos(i)$ be the position of the agent $i$. Furthermore, a function over $i$ is defined for determining the cost of the displacement of the agents through the graph $\lambda_i : E \longrightarrow \mathbb{R}^{\geq 0}$.

For the sake of avoiding inconsistencies in the knowledge shared between agents, it is considered, without any loss in the problem generality, that $V$ and $E$ are common to all the agents. Nonetheless, the definition of the cost function $\lambda_i$ can change with each agent, allowing the latter have a $\lambda_i$ that is adjusted to fit its restrictions.

Let $path(i)$ having $i \in A$, be the path that the agent $i$ is tacking to reach node $t$.

The degree of ambush towards the node $t$ is defined as:

$$\Phi(t) = \frac{|\{i : path(j) =< pos(j), \ \ldots, \ i, \ t >, j \in A\}|}{\min(|\{< i, t >:< i, t >\in E\}|, |A|)} \tag{1}$$

That is, the proportion of nodes adjacent to $t$, from which an agent in $A$ is reaching the goal, regarding the maximum number of adjacent nodes to $t$. Naturally, $\Phi$'s range is defined as $0 \leq \Phi(t) \leq 1$.

The main objective of the proposed variation is to maximise $\Phi$'s value, having path diversity as a top priority, over getting all agents to walk the obvious optimal route.

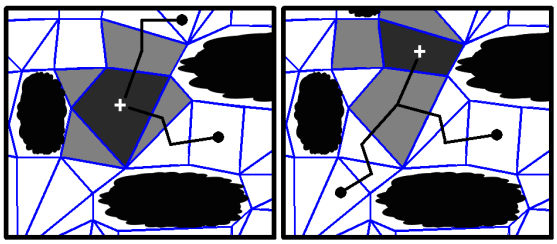

**Fig. 1.** Ambush examples with values of $\Phi(t)$ 1 y 0.5 respectively

See the example shown in figure 1. In the top image there are two agents situated where the black dots are. The objective that both agents want to reach is the white cross located in the dark gray polygon. The adjacent nodes to the goal are painted in light gray. Note that the number of polygons adjacent to the objective is four. Out of those

four, only two are being considered in the agents paths. Therefore, the value of $\Phi(t)$ for this particular case is $\Phi(t) = \frac{2}{min(4,2)} = 1$. It could seem that this ambush result is not optimal, because there are two polygons that are left unreached. However, this is the greatest possible ambush degree, given that only two agents are performing the strategy.

The image at the bottom shows an instance where, out of all the three polygons adjacent to the goal one, only one is being considered in the paths that the agents are tacking. For this case, the metric would give the following result: $\Phi(t) = \frac{1}{min(3,2)} = 0.5$.

With the aim of not penalising cases like the first one, where the ambush was correctly performed, it is important to consider the minimum between the number of agents and the number of nodes adjacent to the objective node.

## 3 Related Works

A similar problem to the one exposed in this paper is given by *Space-Time A** [7]. In this article, a variation of *A** is proposed. It works with an undirected graph, induced from a grid shaped map. The cost function is common for all agents and constant between every node pair.

The main goal of the mentioned paper is to avoid having two different agents in the same node at a given instant of time. For this variation of $A^*$, an expansion in the number of dimensions of *A** is proposed: besides the positions of the agents, the elapsed time is also taken into account. Thus, *Space-Time A** has a considerable increment in the time and memory costs, because the size of the search space has grown.

Other ways of achieving ambush are explained in [9]. This work takes the problem of an agent that has to choose a path across a rectangle, treated as a grid, and a second agent that will have to choose a sub-set of the rectangle trying to block the path of the first one. The approach in this paper is to create a matrix of probabilities that evaluates all the possible combinations of paths and ambush sets, and then choose the best strategy for both agents through the minimax algorithm.

[10] and [11] approach the previous problem with sets of continuous nature. In the first work, discrete versions of the game model are proposed. Afterwards they are used to find optimal solutions that satisfy the original continuous model. The work in [11] proposes various sets of solutions for conditions that were not considered in [10].

Previous works show exponential search spaces, deriving in high computational costs. The $A^*mbush$ family is proposed as an effective method with lower computational order, that can be used in more general conditions.

## 4 A*

The *A** algorithm [6, 4, 3] is a variation of Dijkstra's algorithm [1] that computes paths of minimal cost. It is an informed search algorithm [4], based on the following elements:

- $g$: Represents the accumulated cost from the initial node to the actual node $v$.

- $\hat{h}$: Is an estimate of the cost from the actual node $v$ to the goal.
- $\hat{f} = g + \hat{h}$: An estimate of the cost from the initial node to the goal, having $v$ in the path..

For the sake of securing optimality, the $\hat{h}$ heuristic must be admissible, that is, it shouldn't overestimate any cost against the optimal solution. This algorithm works in a greedy fashion, expanding the next unexplored node with the smallest estimated cost $\hat{f}$ at the moment. This procedure is repeated until the goal is reached.

The estimated running time of $A^*$ is $\mathcal{O}(|V|log(|V|)+|V|*h+|E|)$. Having $h$ as the cost of computing function $\hat{h}$. This cost is derived, assuming an efficient implementation of the priority queue such as a Fibonacci heap [1] and that the heuristic of each node is only computed once.

## 5 A*mbush

In this section we propose *A*mbush*, an $A^*$-based algorithm that solves the ambush generation problem. It consists in a modification of $g$ function, that favours path diversity. We will call this function $g'$.

Let $\Psi(v,i) = 1 + (\#j : j \in A \wedge v \in path(j))$, be the number of agents different from agent $i$, that have the node $v$ in their paths towards $t$ plus one. If an agent is not trying to reach node $t$ at the moment, or it has not performed the search for the node yet, it's path counts as empty. Therefore, the agent is not taken into account when calculating $\Psi(v,i)$'s value.

It is considered that $g'(pos(i),i) = 0$ for the initial node. Let $< v,w >$ be the next edge that should be expanded from the node $v$, in any of the algorithm's iterations. In these conditions, the expression that determines $g'$'s value is $g'(w,i) = g'(v,i) + \lambda_i(< v,w >) \cdot \Psi(w,i)^2$.

Given that $\Psi(v,i) \geq 1$, the path determined by $A^*mbush$ is optimal under the new definition of $g'$. Hence, the properties of $A^*$ are preserved [6]. Nevertheless the path might not be optimal for the original costs function $g$.

Note that $\Psi(v,i) = 1$ for every node $v$ that is not considered in the path of any agent different to $i$. Similarly, $\Psi(v,i) > 1$ in any other case. This condition allows the agents to consider exploring sub-optimal paths in the original graph. The less agents explore this routes, the greater the chance given to other agents to travel trough them.

### 5.1 Complexity

It is possible to precompute the function $\Psi$ for node cost's increment. If this function is stored in a constant structure with fast access, the cost of calculating $g'$ becomes equal to the cost of computing $g$. Therefore, the only change if the algorithm's cost lays in the initial calculation of function $\Psi$.

The asymptotic complexity of $A^*mbush$ is: $\mathcal{O}(|V|log(|V|) + |V| * h + |E| + |A| * |V|)$.

In the field of video games, the graphs of interest are given by the polygonal division scheme of the map [3, 5] (this polygons tend to have a small amount of sides), according

to the traversable regions and their respective adjacencies [3, 5]. Since the number of sides for the polygons is small, this graphs tend to have little density. In consequence, $|E| \in \mathcal{O}(|V|)$, this means that the execution time of this methods for the graphs of interest in the area of video games, is considered to be $\mathcal{O}(|V|(log(|V|)+h+|A|))$. The amortized cost of the increment function is $\mathcal{O}(|V|)$, therefore, the amortized cost of the A$^*$mbush algorithm equals the A$^*$ cost.

## 6 P-A*mbush: Ambush with Priorities assignment method P

Without a strategy that decides which agent calculates it's path first, routes that are inconvenient could be chosen, because the algorithm doesn't take in account the agent's state in the game. For example, if agent $i$ is the closest one to the goal, it could be beneficial to make $i$ perform the path computing first. This way the agent won't end up tacking a long route to generate ambush, wasting an opportunity to catch the wanted element, while an agent that was further away takes the most direct way.

For this work two a simple P method is proposed based on the real distance between the agents an the goal. This is because the positions of the agents are a very general and intuitive property that defines the advantage that an element could have over another. Other character properties like strength, remaining life, stamina, and any others could be used in this method to generate the desired group strategy while performing *A$^*$mbush*.

Figure 2 shows the path that three agents would take when performing *A$^*$mbush* (left Image) and *P-A$^*$ambush* with real distance (right Image). The circles represent the agents, the numbers inside them represent the order in which each agent computes the path it will take and the black cross represents the goal.

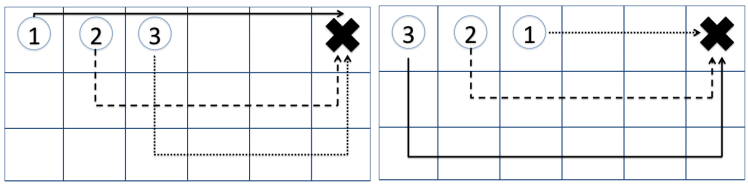


**Fig. 2.** The numbers represent the order in which the agents are calculating the path. Up: Performance of *A$^*$mbush*. Since it does not take the distance to the goal into account, the agent that is closer to the goal, takes a longer path. Down: Performance of *P-A$^*$ambush* using $A^*$'s distance. The agent that can reach the goal first using the path given by $A^*$ has the top priority for calculating the path, the second closest by this distance goes afterwards, and so on.

When using *P-A$^*$ambush*, the behaviours of the agents look more intelligent, since the one that is closer to the goal is taking the most direct path to reach it, instead of the circling around like the one performing *A$^*$mbush* does.

Other advantage that *P-A$^*$ambush* displays is that the goal will always be reached in the minimum time possible by one of the agents. Thus, this algorithm shows qualitative improvements over *A$^*$mbush*.

Although the cost of computing *P-A*ambush* with real distance is bigger than computing *A*mbush* for one agent, the amortized cost invested to calculate *P-A*ambush* for each agent equals *A*mbush*'s amortized cost. However, having agents perform *P-A*mbush* with real distance would decrease the frame rate because of the involved constants. Therefore, using the euclidean distance is proposed as a good middle. It will not perform as well as the one using $A^*$'s distance, but also, it's computational cost will not differ from the one in *A*mbush*.

## 7 R-A*mbush: Ambush with fixed radio R

Getting agents to perform *A*mbush* from the starting point can be disadvantageous, since they can take unnecessarily longer routes, when avoiding each others paths. Making the agents perform *A*mbush* when they are closer to the goal could shorten the increment in this cost.

With this reasoning as a base, *R*-A**ambush* is proposed. It is an A*mbush modification that performs $A^*$ until the agent gets inside a fixed radius $R$ around the goal point. Once at this stage, the agent stats performing *A*mbush*. Note that this transition happens only once. Even if the *A*mbush* algorithm makes the agent leave the radio area, it will not start performing $A^*$ again. This prevents a loop that the intelligence could easily fall into. First, the agent, following the path given by A* comes inside the radius R, then *A*mbush* takes it out of the area when trying to go for a unexplored route, subsequently, this behaviour is interrupted and the A* method takes control again, making the agent go inside the radio. Those steps could be repeated infinitely, so once the radio $R$ is crossed, only *A*mbush* behaviour takes place. Figure 3 shows the performance of the *R*-A**ambush* algorithm.

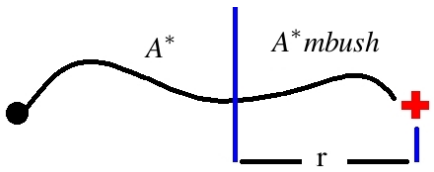


**Fig. 3.** R-A*mbush. The agent located in the black point tries to reach the cross.

The radius $R$ is measured in real distance (A*). It was chosen over the Euclidean, because obstacles in games could make the agents take ambush routes when the path to get to the goal is still very long, and this would antagonize the purpose of reducing the path cost when possible.

## 8 Self Adaptive R-A*mbush: Ambush with adaptive radius R

*R*-A**ambush* provides a solution that is not as general as the one for *A*mbush*. Using the same distance $R$ for the fixed radius in different maps, would produce good results

in some of them and bad ones in others. This means the user would have to manually fix the measure of $R$ by tacking into consideration the size of the map, the path cost function, or the number of obstacles. In addition, given that the cost function can be different for each agent, the radius should be established according to the $\lambda$ measure of every agent.

Thus, *SAR*-A$^*$*ambush* is proposed as a variation of *R*-A$^*$*ambush* that solves this generality problem. Initially, This method makes each agent calculate the path with $A^*$ . Then, a set of points from that route is chosen. This will be the set of the possible radius $R$. The algorithm will select the minimum radius that generates the greatest $\Phi$, starting from the smallest $R$. This is based on the idea of starting to use sub-optimal paths as late as possible. Note that $A^*$ is the specific case where $R = 0$ and $A^*$*mbush* is the one where $R$ is greater than or equal to the real distance between the agent and the goal.

If points that are too close to the goal are inside the set of radius, in particular, $R = 0$, all the other agents will choose the optimal path, once the maximum ambush is reached. This could make the distribution of the agents around the goal very unbalanced towards the nodes that are closest to the agents stating points.

Figure 4 Shows a situation where the agents distribution is not desirable, but the maximum ambush rate is obtained. The agents colored in black performed the method *SAR*-A$^*$*ambush* first. Then, the agents in gray chose the optimal path because the ambush measure cannot be improved anymore, creating an unbalanced distribution around the goal. This can be solved by taking into account a measure of distribution when choosing the radius. Thus, let

$$\Delta = 1 - \frac{(\Sigma i | i \in V \wedge i \in pred(t) \wedge num(i) > \left\lceil \frac{n}{|pred(t)|} \right\rceil : num(i) - \left\lceil \frac{n}{|pred(t)|} \right\rceil}{n - \left\lfloor \frac{n}{|pred(t)|} \right\rfloor} \tag{2}$$

be a metric related with the rate of agents that need to reach the goal from another node to have them equally distributed. The value of the metric is 1 when the agents are perfectly distributed and it tends to zero when the agents are unbalanced. In the formula, $n$ represents the number of agents that already calculated the path. $pred(t)$ is the set of nodes adjacent to the goal node. Finally $num(i)$ is the number of agents that reached the target from the node $i$. The numerator of the main fraction counts the number of the agents that need to be swapped and the denominator is the number of swaps in the worst case scenario (all agents reaching the goal from one node). The only purpose of the subtraction is to use the lexicographical order in the objective function of the strategy. For example, for Figure 4 $\Delta = 0.66$. This means that equal distribution can be reached by making one of the agents that are on the node at the left of the goal, ambush it from above.

The algorithm would search for the maximum ambush rate and in case of a tie it would go for the path that achieves the best distribution. The $\Delta$ value alone is not enough to compare two ambush strategies.

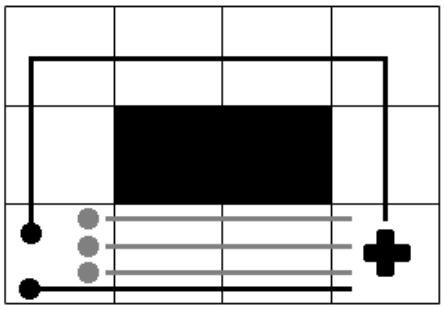

**Fig. 4.** Unbalanced ambush.

### 8.1 Complexity

This variation of *A*mbush* is more inefficient than the original version. If $T(G, A)$ is the complexity in time of computing *A*mbush* for a graph $G = (V, E)$ with $|A|$ agents and the number of candidates to be selected as radius points is $k$, the complexity of the *SAR*-A**ambush* is $\mathcal{O}(k \cdot T(G, A))$.

Since $k \in \mathcal{O}(V)$ (selecting every possible node as radius) the worst case of the algorithm is $\mathcal{O}(|V| \cdot T(G, A))$. Alternative candidate sets can be constructed from taking nodes that are separated by distances that are incremented exponentially, for example, selecting the nodes in positions that are powers of two in the $A^*$ path $(1, 2, 4, 8, \ldots)$. This variation leads to an increment of only $\mathcal{O}(log(|V|))$. In the following experiments the radius set contains every node in the $A^*$ path.

## 9 Results and Analysis

For each experiment five algorithms are studied. A basic implementation of $A^*$ that serves as reference, the three proposed variations $A^*mbush$, $P$-$A^*mbush$ and $SAR$-$A^*mbush$ and finally a Randomized Spanning Tree $RST$. The last one is an algorithm that generates random trees that cover all the graph. It can be interpreted as a change in the agent's cost function by assigning the value of infinite to several edges in the map. The $RST$ is different for each agent, with the goal of generating paths that are variate and stochastic.

For the experiments, two different map instances were considered as they are and as a base for bigger maps. The first one (see Figure 5) is composed by 60 polygons, the second one (see Figure 5), has 85. All the polygons are convex, in order to reduce the verification costs when checking which polygon a point belongs to [12]. Each node in the graph is induced by a polygon in the map, located in it's geometric centroid [3].

The cost function is common to all agents. The cost is proportional to the euclidean distance between the polygon's centers.

Experiments with variable agent numbers were performed. The results are described in section 9.1.

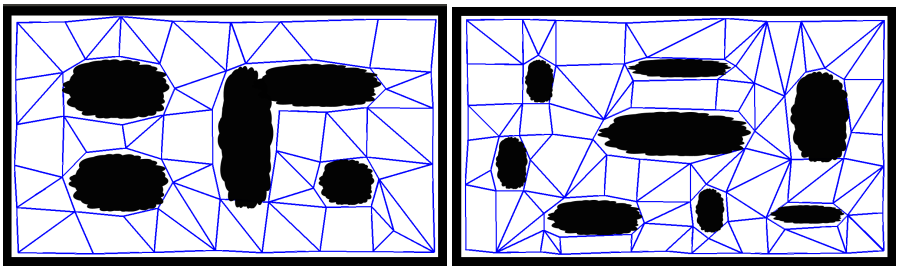

**Fig. 5.** Left: Polygonized map 1 (60 polygons). Right: Polygonized map 2 (85 polygons).

The ambush degree ($\Phi$) towards the goal node is shown for the following algorithms: $A^*$, $RST$, $A^*mbush$, $P$-$A^*mbush$ using real ($A^*$) distance and $SAR$-$A^*mbush$. Besides the ambush degree ($\Phi$), the incremental cost mean is shown. This measure displays the mean of the percentage's increment in the cost of the paths generated by $RST$, $A^*mbush$, $P$-$A^*mbush$ and $SAR$-$A^*mbush$ when compared to the minimal costs obtained with $A^*$.

### 9.1 Global experiments

The results in this section show the top achieved $\Phi$ value in bold font.

**Table 1.** Ambush rate ($\Phi$) - 2-100 agents

| | **Map 1 (60 nodes)** | | | | | **Map 2 (85 nodes)** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| # | $A^*$ | $RST$ | $A^*mbush$ | $P$ | $SAR$ | $A^*$ | $RST$ | $A^*mbush$ | $P$ | $SAR$ |
| 2 | 0.72 | 0.75 | 0.87 | **0.89** | 0.87 | 0.72 | 0.75 | 0.88 | **0.91** | 0.89 |
| 3 | 0.79 | 0.83 | 0.95 | **0.96** | 0.95 | 0.76 | 0.82 | 0.95 | **0.96** | **0.96** |
| 4 | 0.85 | 0.90 | 0.98 | **0.99** | 0.98 | 0.83 | 0.88 | 0.98 | **0.99** | 0.98 |
| 5 | 0.89 | 0.93 | **0.99** | **0.99** | **0.99** | 0.87 | 0.92 | 0.99 | 0.99 | **1.00** |
| 6 | 0.91 | 0.96 | **0.99** | **0.99** | **0.99** | 0.90 | 0.95 | **1.00** | 0.99 | **1.00** |
| 8 | 0.95 | 0.98 | **0.99** | **0.99** | **0.99** | 0.93 | 0.97 | **1.00** | **1.00** | **1.00** |
| 10 | 0.96 | 0.99 | **1.00** | **1.00** | **1.00** | 0.95 | 0.99 | **1.00** | **1.00** | **1.00** |
| 15 | 0.98 | 0.99 | **1.00** | **1.00** | **1.00** | 0.98 | **1.00** | **1.00** | **1.00** | **1.00** |
| 20 | 0.99 | **1.00** | **1.00** | **1.00** | **1.00** | 0.99 | **1.00** | **1.00** | **1.00** | **1.00** |
| 50 | 0.99 | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** |
| 75 | 0.99 | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** |
| 100 | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** |

Tables 1 and 2 show global results for each map, with a variate number of agents. For each case, 1000 random instances of agent's distributions over the polygons in the map were generated. All the cases are compared by locating the goal in each node (polygon) of the graph. Finally, the mean is taken from this results, and it is shown in the tables.

From table 1 it is noticeable that when the number of agent increases, the general $\Phi$ value becomes higher. Also, from more than 8 agents, $A^*mbush$, $P$-$A^*mbush$ and $SAR$-$A^*mbush$report the maximum $\Phi$.

**Table 2.** Mean of the Incremental Distance - 2-100 agents

| | **Map 1 (60 nodes)** | | | | **Map 2 (85 nodes)** | | | |
|---|---|---|---|---|---|---|---|---|
| # | $RST$ | $A^*mbush$ | $P$ | $SAR$ | $RST$ | $A^*mbush$ | $P$ | $SAR$ |
| 2 | 64.54 | 9.00 | 8.54 | **8.40** | 86.07 | 7.05 | **6.06** | 7.22 |
| 3 | 64.67 | 12.98 | 13.18 | **9.78** | 84.93 | 12.40 | 11.63 | **8.91** |
| 4 | 64.99 | 15.13 | 16.54 | **11.94** | 84.24 | 15.50 | 15.97 | **11.21** |
| 5 | 66.13 | 16.35 | 19.02 | **12.12** | 83.96 | 17.80 | 19.08 | **11.49** |
| 6 | 64.55 | 16.87 | 20.58 | **13.01** | 85.22 | 18.97 | 21.94 | **12.56** |
| 8 | 65.53 | 17.62 | 23.94 | **13.45** | 86.81 | 20.89 | 25.96 | **13.42** |
| 10 | 64.93 | 17.45 | 26.21 | **13.35** | 85.88 | 21.42 | 28.86 | **13.81** |
| 15 | 64.24 | 17.51 | 30.15 | **13.27** | 84.91 | 22.15 | 34.43 | **14.04** |
| 20 | 65.22 | 16.88 | 32.11 | **12.74** | 85.22 | 22.00 | 37.22 | **13.95** |
| 50 | 65.13 | 14.19 | 38.42 | **10.76** | 85.06 | 19.02 | 45.00 | **11.84** |
| 75 | 65.03 | 12.86 | 40.42 | **9.88** | 84.77 | 17.48 | 47.31 | **11.09** |
| 100 | 65.27 | 12.08 | 41.88 | **9.39** | 85.26 | 16.47 | 48.88 | **10.39** |

Table 2 shows the mean of the incremental distance, compared against the cost of the path generated by $A^*$ . This results bring to the light the strong disadvantage that arises when using $RST$. Since this method only focuses on generating random routes, it has an mean increment of 85.52% with standard deviation of 0.62 for map one, and a mean increment of 64.73% with standard deviation of 0.55 for map two. This values don't look desirable when compared to the ambush family.

Because $A^*mbush$-based methods make the agents take optimal and and sub-optimal paths, the mean of the incremental distance is lower. Also, the search performed by the ambush based algorithms is informed, so data from the map is being used, while with $RST$ only random paths towards the goal are being pursued.

$SAR$-$A^*mbush$ shows the lowest increment with a mean of 11.5% and 11.66% for maps one and two respectively, with standard deviation of 1.78 and 2.1. This result is expected, since this method makes the agents perform $A^*$ for the larger amount of steps possible, while still securing the highest ambush rate that is reachable.

Results also show that the $P$-$A^*mbush$ technique has a greater incremental distance than $A^*mbush$ for maps 1 and 2 specifically, with a mean of 22.35% and a standard deviation 10.40 for map one, while for map two, the mean is 20,98% and the standard deviation 7.86. Nevertheless, section 9.2 will show that this is not true when applying the algorithms to graphs with larger number of nodes.

Although $P$-$A^*mbush$ increases the cost of the path of the agents that have less priority, this method increases the ambush degree, improving the behavior that $A^*mbush$ provided, which is the goal of this research.

Additionally, both $SAR$-$A^*mbush$ and $P$-$A^*mbush$ assure a quicker attack and a more intelligent looking behavior.

### 9.2 Experiments with multi-scale graphs

**Table 3.** Ambush Rate and Mean of the Incremental Distance with Multiscale Graphs

| | **Ambush Rate** | | | | | **Incremental Distance** | | | |
|---|---|---|---|---|---|---|---|---|---|
| $Nodes$ | $A^*$ | $RST$ | $A^*mbush$ | $P$ | $SAR$ | $RST$ | $A^*mbush$ | $P$ | $SAR$ |
| 85 | 0.88 | 0.97 | **1.00** | **1.00** | **1.00** | 83.05 | 20.12 | 22.30 | **13.99** |
| 170 | 0.92 | 0.97 | **1.00** | 0.99 | **1.00** | 90.06 | 20.34 | 17.89 | **12.22** |
| 425 | 0.78 | 0.94 | 0.97 | **0.98** | **0.98** | 101.17 | 23.06 | 17.07 | **11.44** |
| 850 | 0.80 | 0.94 | 0.96 | **0.97** | **0.97** | 130.80 | 16.86 | 11.66 | **6.13** |
| 1700 | 0.76 | 0.94 | **1.00** | 0.97 | **1.00** | 119.47 | 9.28 | **5.35** | 5.96 |

For this experiments, the graphs were automatically generated by concatenations of the map 2 with it's vertical and horizontal flips. Experiments ran under the same conditions than the global experiments, but with a fixed number of six agents.

Table 3 shows that the group of $A^*mbush$ based methods have better ambush rates than $A^*$ and $RST$. They also show smaller increment that the random method. This means that the desired performance for the proposed algorithms does not decrease when the number of nodes increases. In contrast, $A^*$'s ambush rates become worse when the size of the graph is bigger.

Moreover, table 3 shows that, as the maps become larger two important results emerge. First, he incremental distance becomes smaller for the $A^*mbush$ family. Also, $SAR$-$A^*mbush$ exhibits better ambush rates than the other ones.

## 10 Conclusions

With a raise in the computational cost that has little significance, $A^*mbush$ generates an evident upgrade in the ambush behavior. Likewise, $P$-$A^*mbush$ shows a superior behavior over the traditional $A^*mbush$, because it successfully increases the path diversity.

As it can be appreciated in the results, even in the worst case scenario, the $A^*mbush$ family never produces a result that is worse than the one made by $A^*$. On the same line, $P$-$A^*mbush$ does not show in any case a downgrade when compared with $A^*mbush$.

$SAR$-$A^*mbush$ always shows the smallest incremental distance. And the best ambush rates when the maps are large. Although, due it's computational cost, the use of $P$-$A^*mbush$ could be more desirable when the processing resources are limited.

When paths are diverse, the agents performing any of the $A^*mbush$ methods become more naturally distributed over the map. Not having all the elements rounded up in a small space can be advantageous when agents change from one state of behavior to another. For example, if the conduct changes from pursue to flee, the ambush based algorithms not only allow a better behavior, but also, they maintain the agents further away from each other, which is a better disposition when it comes to strategies of scape.

In contrast with the pre-established conducts that are generally implemented for acting up in specific situations or moments of each game, the proposed algorithms enhance the diversity of ambush situations that can be produced as a tactical move from the agents.

$P$-$A^*mbush$ generates an upgrade to the way the intelligence in the agent's behavior is perceived and $SAR$-$A^*mbush$ delivers an even better looking one. This way, not only the paths become more variate, but the conduct of the agents looks better. It makes the elements in the game act in a way that looks more natural and perform an ambush strategy that is more effective.

For more results and experiments visit http://www.gia.usb.ve/ kelwin/ambush/.

## References

1. T. Cormen, C. Leiserson, R. Rivest and C. Stein, *Introduction to Algorithms, 3rd ed*. The MIT Press, 2009.
2. M. Gendreau and J. Potvin, *Handbook of Metaheuristics, 2nd ed*. Springer, 2010.
3. I. Millington and J. Funge, *Artificial Intelligence for Games, 2nd ed*. Morgan Kaufmann Publishers, 2009.
4. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach, 2nd ed*. Prentice-Hall, Englewood Cliffs, NJ,, 1993.
5. X. Cui and H. Shii, "Direction oriented pathfinding in video games," *International Journal of Artificial Intelligence & applications 2*, 2011.
6. P. Hart, N. Nilsson and B. Raphael, "Correction to a formal basis for the heuristic determination of minimum cost paths," *SIGART Newsletter 37*, pp. 28–29, 1972.
7. D. Silver, "Cooperative pathfinding," *AI Game Programming Wisdom 3*, pp. 99–111, 2006.
8. R. Teixeira, K. Marzullo, S. Savaje and G. Voelker, "In search of path diversity in isp networks," *IMC*, pp. 27–29, 2003.
9. W. Ruckle, R. Fennell, P. Holmes and C. Fennemore, "Ambushing Random Walks I: Finite Models," *Operations Research, vol 24*, No. 2, 1976.
10. W. Ruckle, "Ambushing Random Walks II: Continuous Models," *Operations Research, vol 29*, No. 1, 1981.
11. I. Woodward, "Discretization of the Continuous Ambush Game," *Naval Research Logistics, Vol. 50*, 2003.
12. E. Haines, "Point in Polygon Strategies," *http://erich.realtimerendering.com/ ptinpoly/*, 2001.