A*mbush: una Variación de A* para Generar Emboscadas y Diversidad de Caminos

K. Fernández, G. González, C. Chang

Grupo de Inteligencia Artificial, Departamento de Computación y Tecnología de la Información, Universidad Simón Bolívar, Venezuela

RESUMEN

En el área de Inteligencia Artificial para videojuegos, A es un algoritmo comúnmente propuesto para generar caminos que utilizarán los agentes autónomos. Aunque A* garantiza optimalidad, tiende a producir conductas similares en elementos que estén cercanos entre ellos y no asegura un ataque por diversos flancos cuando los agentes están distribuidos de manera esparcida. Para generar conductas de emboscada y diversidad de caminos, se propone una modificación del algoritmo llamada A*mbush, que considera para el costo de los nodos (o arcos), la cantidad de agentes que tienen calculado ese punto dentro de su camino hacia la meta.*

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 doesn't secure an attack that comes from different places. Having the generation of ambush behaviours and diversity of paths as the goal, A*mbush is proposed. It is a modification of A* that considers the amount of agents that have a specific node (or edge) in their calculated path, when it's computing the cost function of that point in the graph.*

Keywords: A*, estrategias grupales, búsqueda de caminos.

1. Introducción

En el área de Inteligencia Artificial para Videojuegos, generar agentes con conductas que luzcan inteligentes para el usuario, es un reto constante [MF09]. Entre estos comportamientos, el usuario espera apreciar agentes que desarrollen movimientos tácticos y estratégicos grupales, los cuales suelen ser sumamente complejos de implementar. Esto muchas veces genera como resultado acciones prestablecidas que el usuario puede fácilmente identificar después de varias ejecuciones del juego.

Un problema muy tratado es la búsqueda de caminos a un punto común, por parte de grupos de agentes dentro de un juego [MF09]. Este punto suele venir dado por un lugar en el mapa de juego, potencialmente la posición del oponente. El esquema regularmente utilizado es generar caminos de costo mínimo [HNR72] [RN93] hacia este punto, sobre el grafo inducido por el mapa del juego. Es muy probable, que estos caminos confluyan, evitando la diversidad de rutas y exploración del mapa.

Al efectuar una persecución al oponente, la utilización de caminos óptimos como estrategia deja muchos espacios de escapatoria libres, por lo que es de especial interés generar mecanismos de diversificación de rutas que generen situaciones de emboscada.

La técnica expuesta próximamente es adaptable a muchos contextos en los que, si bien no es necesaria una situación de emboscada, es importante generar diversidad de caminos con el fin de no sobresaturar ciertos sectores del grafo subyacente. Ejemplo de estos son controladores de tráfico, enrutamiento de paquetes físicos o digitales [TMSV03], robótica, entre otros.

2. Definición Formal del Problema

Sea $G = (V,E)$ un grafo (dirigido o no), donde V es el conjunto de nodos y E el conjunto de arcos.

Sea A un conjunto de agentes interesados en alcanzar un punto $t \in V$. Se tiene que cada agente $i \in A$, se encuentra en algún nodo del grafo denominado $pos(i)$. Además, se define para i la función de costos de su desplazamiento por el grafo $\lambda_i : E \longrightarrow \mathbb{R}^{\geq 0}$.

Se considera, sin pérdida de generalidad, que V y E son comunes para todos los agentes, con el fin de no generar inconsistencias en la información compartida entre estos. Sin embargo, cada agente puede tener función de costos distintas, adaptadas a sus restricciones.

Sea $path(i)$ con $i \in A$, el camino que está siguiendo el agente i hasta el nodo t.

Se define el grado de emboscada del nodo t como:

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

Es decir, la proporción de nodos adyacentes al nodo t, desde los cuales algún agente está alcanzándolo con respecto al máximo número de nodos adyacentes a t mediante los cuales, los agentes en A podrían alcanzarlo. Por lo tanto, $0 \leq \Phi(t) \leq 1$.

El objetivo de la variación propuesta es maximizar el valor de $\Phi(t)$, dando prioridad a la diversidad de los caminos considerados por el conjunto de agentes sobre su optimalidad.

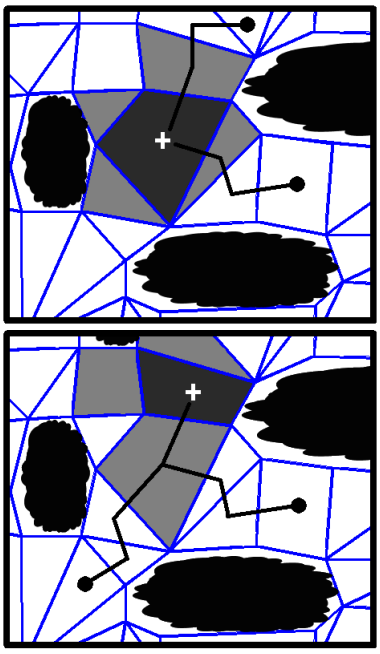

Figura 1: *Ejemplos de emboscadas con valores de $\Phi(t)$ 1 y 0.5 respectivamente*

Véase por ejemplo la figura 1, en ella se puede observar en la imagen superior dos agentes situados en los puntos negros. El objetivo a alcanzar por ambos agentes es la cruz blanca situada en el polígono de color gris oscuro. Los nodos adyacentes al polígono objetivo se encuentran demarcados con color gris claro. Nótese que el número de polígonos adyacentes al destino es de cuatro. De estos cuatro polígonos, sólo dos están siendo considerados en los caminos de los agentes. Por lo tanto, el valor $\Phi(t) = \frac{2}{min(4,2)} = 1$. Podría considerarse que este tipo de emboscada no es óptimo dado que aún quedan dos polígonos libres de escapatoria. Sin embargo, es el mayor grado de emboscada posible dado que existen sólo dos agentes.

En el segundo caso de la misma figura, puede apreciarse una instancia en la cual de los tres posibles polígonos adyacentes existe sólo uno siendo considerado en algún camino. Para este caso, la métrica daría como resultado $\Phi(t) = \frac{1}{min(3,2)} = 0.5$.

La importancia de considerar el mínimo entre el número de nodos adyacentes y el número de agentes es no penalizar casos como el primero, donde los caminos responden correctamente al problema planteado.

3. A*

El algoritmo de A* [HNR72] [RN93] [MF09] es una variación del algoritmo de Dijkstra [CLRS09] para cómputos de caminos de costo mínimo.

Consiste en un algoritmo de búsqueda informada [RN93], basado en los siguientes elementos:

- g: Es el costo acumulado desde el nodo inicial a un nodo actual v.
- $\hat{h}$: Estimado del costo desde el nodo actual v a la meta.
- $\hat{f} = g + \hat{h}$: Estimado del costo desde el nodo inicial a la meta, pasando por v.

Para garantizar optimalidad, la heuristica $\hat{h}$ debe ser admisible [HNR72], es decir, no debe estimar costos mayores al óptimo del grafo.

El algoritmo actúa de forma voraz, expandiendo el siguiente nodo no explorado con menor costo estimado $\hat{f}$. Este proceso continúa hasta llegar a la meta.

En la figura 2 puede apreciar un pseudocódigo del algoritmo de A^*.

```
A_star(i, t):
    Para cada v en V:
        v.costo = inf
    pos(i).costo = 0

    nodo ini
    ini.v = pos(i)
    ini.f = 0
    ini.g = 0
    ini.h = h(v,t)
    ini.padre = null

    Mientras !vacia(q):
        nodo n = menor(q)
        extraer_menor(q)

        Si n.v = t:
            retornar reconstruir_camino(n)

        Si n.g > n.v.costo:
            continuar

        Para cada w sucesor de n.v:
            nodo s
            s.v = w
            s.h = h(w,t)
            s.g = n.g + lambda_i(<n.v, w>)
            s.f = s.g + s.h
            s.padre = v

            Si s.g < w.costo:
                agregar(q,s)
        Fin para
    Fin Mientras
```

Figura 2: *Pseudocódigo del Algoritmo A* del agente i con destino t.*

En este caso, la complejidad asintótica en tiempo del algoritmo de A* es de $\mathcal{O}(|V|log(|V|) + |V| * h + |E|)$, con h el costo de cómputo de la función $\hat{h}$, suponiendo que se cuenta con una implementación eficiente de cola de prioridades tal como un Fibonacci heap [CLRS09] (estructura de montículo que permite acceder al mínimo elemento del montículo y

agregar un elemento en tiempo constante; además de eliminar uno en tiempo logarítmico amortizado) y que estamos computando el costo heurístico de cada nodo una sola vez.

4. A*mbush

La variación propuesta al algoritmo A^* para solucionar el problema de generación de emboscadas A*mbush (por la traducción al inglés de emboscada, ambush) consiste en una modificación de la función g, que favorezca la diversidad de caminos, a la cual denominaremos g'.

4.1. Enfoque de Costos Agregados en Nodos

Sea $\Psi(v,i) = 1 + (\#j : j \in A \wedge v \in path(j))$, el número de agentes distintos al agente i, que contienen al nodo v en sus caminos hasta t. Se considera que si un agente no está buscando en dicho momento alcanzar el nodo t, o si aún no ha realizado la búsqueda del camino, este es vacío, por lo que no se consideran en el cómputo de $\Psi(v,i)$; dado que i no ha computado ya su camino hasta t, se considera nulo su camino.

Para el nodo inicial, se considera $g'(pos(i),i) = 0$. Sea $< v,w >$ el siguiente arco a considerar en la expansión del nodo v en una iteración cualquiera del algoritmo, se considera $g'(w,i) = g'(v,i) + \lambda_i(< v,w >) \cdot \Psi(w,i)^2$

Dado que $\Psi(v,i) \geq 1$, el camino obtenido por *A*mbush* es óptimo sobre la nueva definición de g', por lo que las propiedades de A^* se mantienen [HNR72]. Sin embargo, sobre la función original de costos, el camino obtenido no es necesariamente óptimo.

Es importante notar que $\Psi(v,i) = 1$ para todo v no considerado en caminos de otros agentes. Análogamente, $\Psi(v,i) > 1$ en otro caso. Esta condición genera que los agentes consideren la exploración de caminos subóptimos en el grafo original en la medida de que un menor número de agentes se encuentren explorándolos.

En la figura 3 se puede observar el pseudocódigo de esta variante del algoritmo A^*.

4.2. Enfoque de Costos Agregados en Arcos

La variación efectuada en este caso, es similar a la anterior, con la excepción que se define la función Ψ sobre arcos en vez de nodos, es decir, se define:

- $\Psi(< v,w >, i) = 1 + (\#j : j \in A \wedge < v,w > \in path(j))$
- $g'(w,i) = g'(v,i) + \lambda_i(< v,w >) \cdot \Psi(< v,w >, i)^2$

Las mismas propiedades que se cumplen sobre la variación de incremento de costos en nodos, se cumplen para la variación actual.

4.3. Complejidad

Para ambas variaciones, es posible precomputar la función de incremento de costos Ψ. Si almacenamos dicha función en una estructura de acceso constante, el costo de computar g' es asintóticamente igual al de g, por lo tanto, la única variación en el costo del algoritmo, viene dada por el cómputo inicial de la función Ψ.

```
A_star_mbush(i, t):

    // Inicializacion de costos agregados
    Para cada v en V:
        extra[v] = 1

    // Calculo del costo agregado
    Para cada j en A - {i}:
        Para cada w en path(j):
            extra[w] += 1

    Para cada v en V:
        v.costo = inf
    pos(i).costo = 0

    nodo ini
    ini.v = pos(i)
    ini.f = 0
    ini.g = 0
    ini.h = h(v,t)
    ini.padre = null

    Mientras !vacia(q):
        nodo n = menor(q)
        extraer_menor(q)

        Si n.v = t:
            retornar reconstruir_camino(n)

        Si n.g > n.v.costo:
            continuar

        Para cada w sucesor de n.v:
          nodo s
          s.v = w
          s.h = h(w,t)
          s.g = n.g
             + lambda_i(<n.v, w>)*extra[w]^2
          s.f = s.g + s.h
          s.padre = v

          Si s.g < w.costo:
              agregar(q,s)
        Fin para
    Fin Mientras
```

Figura 3: *Pseudocódigo del Algoritmo A*mbush del agente i con destino t.*

Ambas variaciones, tienen complejidad en tiempo $\mathcal{O}(|V|log(|V|) + |V| * h + |E| + |A| * |V|)$.

En el campo específico de los videjuegos, los grafos de interés, vienen dados por la división en polígonos del mapa [MF09] [CS11] (regularmente con pocos lados), según las regiones transitables de éste o cuadrículas y sus adyacencias [MF09] [CS11]. Dado que los polígonos tienen un número reducido de lados, estos grafos, suelen ser poco densos; es decir, $|E| \in \mathcal{O}(|V|)$, por lo que el tiempo de ejecución de este método, sobre los grafos de interés en el área de videojuegos, viene dado por $\mathcal{O}(|V|(log(|V|) + h + |A|))$. En la figura 4 se puede apreciar un ejemplo claro de la descomposición de un mapa de juego en polígonos. Cada polígono se considera un nodo del espacio de búsqueda. Dos nodos son adyacentes si comparten un lado común.

Adicionalmente, el número de agentes suele no ser lo suficientemente grande para afectar significativamente la com-

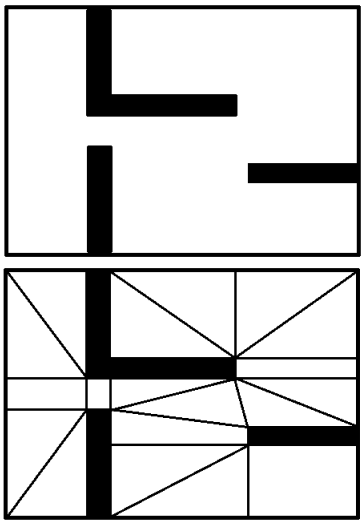

Figura 4: *Arriba: Mapa de juego, los bloques negros son obstáculos. Abajo: Partición del mapa del juego en polígonos convexos.*

plejidad del algoritmo. Casos extremos en los que sea necesaria la utilización de un gran número de agentes, se pueden solucionar con la agrupación de éstos bajo líderes ficticios [MF09], de modo que por cada grupo se realice un solo cálculo del camino. Naturalmente, mientras se agrupen los individuos en conjuntos más grandes, la diversidad de caminos se verá afectada negativamente.

5. Trabajos Relacionados

Un problema similar al aquí expuesto se da con *Space-Time* A^* [Sil06], que plantea una variante de A^* donde se tiene un grafo no dirigido en forma de cuadrícula, con función de costos común para todos los agentes y constante entre cada par de nodos.

El objetivo principal de dicho trabajo es evitar el paso de dos agentes por un mismo nodo en un instante de tiempo dado. La variación planteada, incluye una extensión en el número de dimensiones de A^*: se considera además de la posición de los agentes, el tiempo transcurrido. Dicha variación tiene un costo añadido en tiempo y memoria considerable debido a que se incrementa el tamaño del espacio de búsqueda.

Aunque los algoritmos tienen objetivos diferentes, A^*mbush presenta una clara ventaja sobre (*Space-Time* A^*) en varios aspectos.

En primer lugar, la solución propuesta en este artículo resuelve mediante un algoritmo de menor complejidad de cómputo problemas más generales, debido a que permite grafos dirigidos con costos variables. Asimismo se puede trabajar con agentes que tengan grafos distintos entre sí.

Por otro lado, en el caso de tener agentes con distintas velocidades, *Space-Time* A^* genera un espacio de búsqueda en extremo denso, cuando la velocidad relativa entre los elementos del juego es alta. En cambio, A^*mbush no se ve afectado por la diferencia de velocidades entre los agentes.

Asignando costos homogéneos para cada arco e incremento infinito sobre nodos o arcos ocupados, el problema que resuelve A^*mbush se reduce al problema planteado para *Space-Time* A^*.

6. Experimentos

Dada la naturaleza de los grafos de interés en el área de videojuegos, la relación definida por los arcos es abstracta, es decir, un agente no se encuentra realmente en un arco entre un par de polígonos [MF09]. Es por esto que en los experimentos se considerará la variación de acumulación de costos en nodos y no en arcos.

Para cada experimento se estudian tres algoritmos. Una implementación base de $A*$ que sirve de referencia, una implementación de generación de caminos simples pseudo-aleatorios mediante búsqueda en profundidad, como mecanismo eficiente que genere diversidad de caminos y por último, la variación propuesta, A^*mbush.

Para el algoritmo A^* se muestra el valor de emboscada generado con dicho algoritmo (Φ_{A*}). Para aleatorio y A^*mbush, se muestra el incremento porcentual del costo promedio de los caminos generados con cada método (I_{Rand} e I_{A^*mbush} respectivamente) en contraste con los costos mínimos obtenidos por A^* y el grado de emboscada (Φ).

Serán considerados en los experimentos dos instancias de mapas, la primera instancia (ver figura 5) cuenta con 60 polígonos y la segunda (ver figura 6) con 85. Todos los polígonos son convexos para reducir el costo de la verificación de pertenencia de un punto a un polígono [Hai01]. Cada polígono induce un nodo en el grafo ubicado en su centro geométrico [MF09]. Existen otras posibles convenciones para la discretización de los mapas de juego, se utilizan estas convenciones dado que son las más empleadas [MF09]. La función de costos es común para los agentes, se considera el costo proporcional a la distancia euclidiana entre el centro de los polígonos.

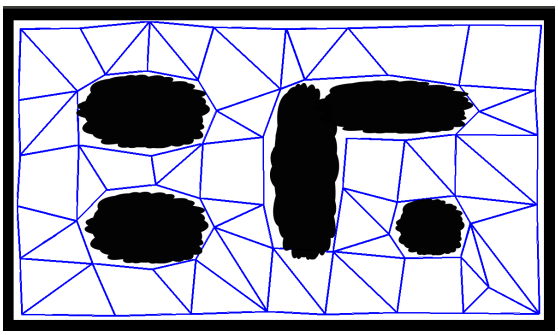

Figura 5: *Mapa 1 poligonalizado (60 polígonos).*

En las secciones 6.1 y 6.2 se presentan experimentos particulares con tres y cinco agentes, respectivamente. Dado que los polígonos generados tienen a lo sumo cuatro lados, es posible garantizar cobertura máxima con cuatro agentes.

En los ejemplos correspondientes a agentes cercanos, se disponen todos los agentes en el mismo polígono inicial. En cambio, en los ejemplos con agentes alejados, se coloca cada uno en polígonos diferentes distanciados. Igualmente, se realizaron experimentos variando el número de agentes,

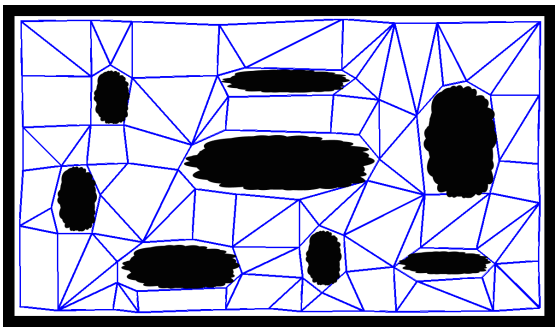

Figura 6: *Mapa 2 poligonalizado (85 polígonos).*

los resultados correspondientes a dichos experimentos se describen en la sección 6.3. Los resultados señalados en negritas indican el máximo valor de Φ encontrado.

6.1. Experimentos con tres agentes

Mapa 1 (60 nodos, 3 agentes)					
E j	Φ_{A^*}	I_{Rand}	Φ_{Rand}	I_{A^*mbush}	Φ_{A^*mbush}
a0	0.50	51.58	**1.00**	8.56	**1.00**
a1	**0.67**	12.51	0.33	0.65	**0.67**
a2	**1.00**	103.41	0.50	0.00	**1.00**
a3	0.67	28.81	0.33	0.90	**1.00**
a4	**1.00**	439.06	0.50	0.00	**1.00**
a5	**1.00**	1.76	**1.00**	4.98	**1.00**
a6	**1.00**	156.84	**1.00**	0.00	**1.00**
a7	**1.00**	206.13	0.50	0.00	**1.00**
a8	**1.00**	147.87	**1.00**	4.51	**1.00**
a9	**1.00**	142.67	0.50	0.00	**1.00**
c0	0.33	284.27	0.33	92.87	**0.67**
c1	0.33	217.75	0.33	57.02	**1.00**
c2	0.50	141.86	0.50	47.29	**1.00**
c3	0.50	0.00	0.50	60.79	**1.00**
c4	0.50	143.77	0.50	22.98	**1.00**
c5	0.33	87.10	0.33	31.73	**0.67**
c6	0.33	139.49	0.33	90.91	**1.00**
c7	0.33	72.66	0.33	23.16	**0.67**
c8	0.50	2.81	0.50	13.64	**1.00**
c9	0.33	0.00	0.33	43.77	**1.00**

Tabla 1: *Resultados de experimentos con tres agentes. Los casos con a_i se corresponden a ejemplos con agentes alejados y los casos c_i a agentes en el mismo polígono inicial.*

En la tablas 1 y 2 se presentan los experimentos con tres agentes para el mapa uno y dos respectivamente. En cada tabla, el primer bloque de experimentos se corresponde a casos con agentes alejados, mientras que el segundo a agentes ubicados inicialmente en el mismo polígono.

En la tabla 1 se puede apreciar que A^*mbush logra ser el mejor de las tres técnicas propuestas, mejorando en la mayoría de los casos la métrica de emboscada. Son de particular interés los casos a_2 y todos los casos comprendidos entre el a_4 y a_9, donde A^* logra el valor máximo de emboscada. En estos casos, A^*mbush incrementa el costo promedio de los caminos generados en a lo sumo 5%, por lo que se muestra útil aún en casos donde la emboscada se da naturalmente.

Mapa 2 (85 nodos, 3 agentes)					
E j	Φ_{A^*}	I_{Rand}	Φ_{Rand}	I_{A^*mbush}	Φ_{A^*mbush}
a0	0.67	242.81	0.33	4.95	**1.00**
a1	**0.50**	62.04	**0.50**	0.72	**0.50**
a2	**1.00**	254.91	0.50	0.00	**1.00**
a3	**1.00**	891.31	0.50	5.66	**1.00**
a4	0.50	252.42	**1.00**	11.15	**1.00**
a5	0.50	43.63	0.50	10.08	**1.00**
a6	**1.00**	498.94	**1.00**	0.00	**1.00**
a7	**1.00**	261.09	0.50	10.42	**1.00**
a8	**0.67**	39.64	0.33	0.00	**0.67**
a9	0.50	386.92	**1.00**	9.37	**1.00**
c0	0.50	344.86	0.50	114.95	**1.00**
c1	0.33	22.20	0.33	7.40	**0.67**
c2	0.50	24.70	0.50	58.64	**1.00**
c3	0.50	64.63	0.50	8.48	**1.00**
c4	0.33	303.53	0.33	17.12	**0.67**
c5	0.50	354.85	0.50	90.76	**1.00**
c6	0.33	956.07	0.33	83.64	**0.67**
c7	0.33	141.46	0.33	23.36	**0.67**
c8	0.50	1287.64	0.50	152.28	**1.00**
c9	0.50	253.61	0.50	55.86	**1.00**

Tabla 2: *Resultados de experimentos sobre el Mapa 2 con tres agentes. Los casos con a_i se corresponden a ejemplos con agentes alejados y los casos c_i a agentes en el mismo polígono inicial.*

Igualmente, casos como a_0, donde la versión pseudo-aleatoria alcanza el mismo valor de A^*mbush, se aprecia un incremento en el costo de los caminos considerable en contraste con la variación propuesta. Teniendo para este caso particular un incremento de 51.58% para la versión aleatoria y 8.56% para A^*mbush.

En los casos donde los agentes parten de la misma posición inicial, la mejora de A^*mbush es considerable, logrando en su mayoría maximizar el valor de la métrica de emboscada.

En la figura 7 se puede apreciar cómo tres agentes partiendo desde el mismo punto toman caminos distintos generando una emboscada al punto destino marcado con una cruz. Los caminos mostrados se encuentran en bruto, es decir, se muestran las uniones directas entre los centros de los polígonos, generando movimientos bruscos. Sin embargo, se pueden utilizar algoritmos de suavizado de caminos [MF09] para evitar cambios fuertes en la orientación de los agentes.

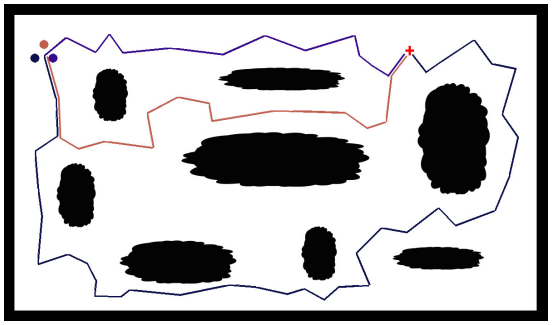

Figura 7: *A*mbush en mapa 2 con tres agentes cercanos*

6.2. Experimentos con cinco agentes

Mapa 1 (60 nodos, 5 agentes)					
Ej	Φ_{A^*}	I_{Rand}	Φ_{Rand}	I_{A^*mbush}	Φ_{A^*mbush}
a0	**1.00**	129.89	**1.00**	5.14	**1.00**
a1	0.67	7.51	0.33	7.02	**1.00**
a2	**1.00**	128.00	0.50	8.57	**1.00**
a3	0.67	18.55	0.33	9.98	**1.00**
a4	**1.00**	289.28	**1.00**	0.00	**1.00**
a5	**1.00**	1.06	**1.00**	21.72	**1.00**
a6	**1.00**	119.92	**1.00**	30.52	**1.00**
a7	**1.00**	181.23	0.50	6.90	**1.00**
a8	**1.00**	147.59	**1.00**	11.64	**1.00**
a9	**1.00**	142.83	0.50	9.33	**1.00**
c0	0.33	284.27	0.33	112.58	**0.67**
c1	0.33	217.75	0.33	37.11	**1.00**
c2	0.50	141.86	0.50	111.15	**1.00**
c3	0.50	0.00	0.50	104.96	**1.00**
c4	0.50	143.77	0.50	49.69	**1.00**
c5	0.33	87.10	0.33	20.66	**0.67**
c6	0.33	139.49	0.33	54.55	**1.00**
c7	0.33	72.66	0.33	48.81	**1.00**
c8	0.50	2.81	0.50	36.49	**1.00**
c9	0.33	0.00	0.33	92.86	**1.00**

Tabla 3: *Resultados de experimentos sobbre el Mapa 1 con cinco agentes. Los casos con a_i se corresponden a ejemplos con agentes alejados y los casos c_i a agentes en el mismo polígono inicial.*

En las tabla 3 y 4 se aprecian experimentos análogos a los expuestos en las tablas 1 y 2, pero con un total de cinco agentes.

A diferencia de los experimentos con tres agentes, en los experimentos con cinco agentes, se logró mejorar en todos los casos posibles la calidad de la emboscada. Es importante destacar que, si bien el valor de incremento del costo de los caminos tiende a aumentar con el número de agentes, A^*mbush logra mejorar cuantitativamente el comportamiento de estos.

En la figura 8 se puede apreciar una gran diversidad de caminos seleccionados. Nótese que adicionalmente a haber generado una emboscada, los agentes se distribuyeron a lo largo del mapa.

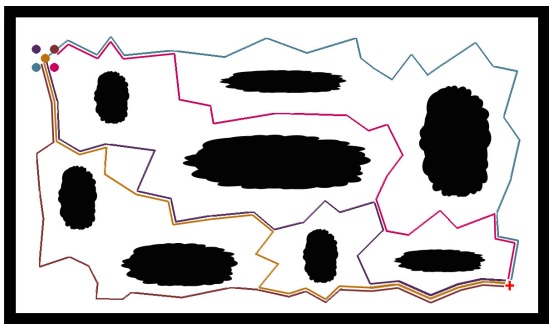

Figura 8: *Ambush en mapa 1 con cinco agentes cercanos*

Mapa 2 (85 nodos, 5 agentes)					
Ej	Φ_{A^*}	I_{Rand}	Φ_{Rand}	I_{A^*mbush}	Φ_{A^*mbush}
a0	**1.00**	223.75	0.33	24.19	**1.00**
a1	**1.00**	122.20	0.50	6.76	**1.00**
a2	**1.00**	284.70	0.50	9.59	**1.00**
a3	**1.00**	562.86	0.50	10.97	**1.00**
a4	**1.00**	185.77	**1.00**	1.08	**1.00**
a5	0.50	26.88	0.50	19.96	**1.00**
a6	**1.00**	386.65	**1.00**	4.84	**1.00**
a7	**1.00**	216.45	0.50	28.72	**1.00**
a8	0.50	37.85	0.25	25.38	**1.00**
a9	0.50	310.39	**1.00**	5.62	**1.00**
c0	0.50	344.86	0.50	68.97	**1.00**
c1	0.33	22.20	0.33	83.28	**1.00**
c2	0.50	24.70	0.50	44.61	**1.00**
c3	0.50	64.63	0.50	6.49	**1.00**
c4	0.25	303.53	0.25	83.31	**0.75**
c5	**1.00**	332.13	**1.00**	55.54	**1.00**
c6	0.33	956.07	0.33	135.02	**1.00**
c7	0.33	141.46	0.33	43.96	**1.00**
c8	0.50	1287.64	0.50	91.37	**1.00**
c9	0.50	253.61	0.50	34.54	**1.00**

Tabla 4: *Resultados de experimentos sobre el Mapa 2 con cinco agentes. Los casos con a_i se corresponden a ejemplos con agentes alejados y los casos c_i a agentes en el mismo polígono inicial.*

6.3. Experimentos Globales

En las tablas 6 y 5 se muestran resultados globales de cada mapa variando el número de agentes. Para cada caso, se generaron 1000 casos aleatorios de distribución de los agentes en los polígonos. Cada una de las distribuciones aleatorias se compara colocando como objetivo cada uno de los polígonos del grafo. Finalmente se promedian los resultados obtenidos para cada métrica.

No se realizó la exploración completa en el espacio de posibilidades dado su rápido crecimiento. De ser necesario disponer k agentes en un grafo con $|V|$ nodos, el número total de combinaciones es de $|V|^{k+1}$, cada una de esas combinaciones debe ejecutar los mecanismos propuestos para cada agente, por lo que la complejidad total sería $\mathcal{O}((|V|log|V|+|E|)\cdot k\cdot|V|^{k+1})$.

Es importante notar de las tablas 6 y 5 como al incrementar el número de agentes, el grado de emboscada de las tres estrategias planteadas aumenta. Sin embargo, A^*bmush logra situarse en todos los casos por encima de las demás estrategias. Logrando a partir de seis agentes para ambos grafos propuestos alcanzar en cada caso el valor óptimo de emboscada, en paralelo con un bajo incremento porcentual en el costo de los caminos de los agentes.

7. Análisis de los Resultados

Se puede observar que A^*mbush genera de manera efectiva incrementar el valor de emboscada, comparado con la estrategia tradicional de búsqueda de caminos A^*. Si bien en ciertos casos no logra una mejora de esta métrica, es notable en la práctica la diversidad de estos caminos. Nótese que la métrica de evaluación sólo considera el segmento final de los caminos obtenidos. Por otro lado, a diferencia de métodos de

Mapa 1 (60 nodos, 3-20 agentes)					
#	Φ_{A*}	I_{Rand}	Φ_{Rand}	$I_{A*mbush}$	$\Phi_{A*mbush}$
3	0.78	406.85	0.78	11.81	**0.95**
4	0.84	403.67	0.84	15.53	**0.99**
5	0.86	402.89	0.87	17.88	**0.99**
6	0.90	405.26	0.89	19.33	**1.00**
7	0.91	406.06	0.91	20.63	**1.00**
8	0.93	405.29	0.92	20.81	**1.00**
9	0.94	403.61	0.93	21.25	**1.00**
10	0.95	404.69	0.94	21.67	**1.00**
11	0.96	405.28	0.94	21.94	**1.00**
12	0.97	405.45	0.95	21.92	**1.00**
13	0.97	405.87	0.95	21.97	**1.00**
14	0.98	405.12	0.95	22.19	**1.00**
15	0.98	403.69	0.96	22.14	**1.00**
16	0.98	403.07	0.96	21.95	**1.00**
17	0.99	405.29	0.96	22.01	**1.00**
18	0.99	405.13	0.96	21.93	**1.00**
19	0.99	403.78	0.96	21.82	**1.00**
20	0.99	403.50	0.97	21.75	**1.00**

Tabla 5: *Resultados de experimentos con número variable de agentes para el Mapa 1.*

Mapa 2 (85 nodos, 3-20 agentes)					
#	Φ_{A*}	I_{Rand}	Φ_{Rand}	$I_{A*mbush}$	$\Phi_{A*mbush}$
3	0.77	407.06	0.78	12.08	**0.95**
4	0.83	409.85	0.84	15.64	**0.99**
5	0.87	403.54	0.87	17.67	**0.99**
6	0.90	403.63	0.89	19.32	**1.00**
7	0.92	404.63	0.91	19.86	**1.00**
8	0.94	403.69	0.92	20.66	**1.00**
9	0.94	404.42	0.93	21.56	**1.00**
10	0.95	403.30	0.94	21.54	**1.00**
11	0.96	402.53	0.94	21.95	**1.00**
12	0.97	404.34	0.95	21.85	**1.00**
13	0.97	405.51	0.95	21.97	**1.00**
14	0.98	403.64	0.95	22.09	**1.00**
15	0.98	404.98	0.96	22.19	**1.00**
16	0.98	403.45	0.96	21.94	**1.00**
17	0.98	404.12	0.96	21.95	**1.00**
18	0.99	404.21	0.96	21.94	**1.00**
19	0.99	404.70	0.96	21.98	**1.00**
20	0.99	404.67	0.96	21.90	**1.00**

Tabla 6: *Resultados de experimentos con número variable de agentes para el Mapa 2.*

generación de caminos diversos como el pseudo-aleatorio, el incremento en el costo de los caminos obtenidos se mantiene en rangos razonables.

En los experimentos desarrollados, A^*mbush no varía significativamente con respecto al número de polígonos. En cambio, se puede apreciar cómo la posición inicial de los agentes, puede afectar el desenvolvimiento del algoritmo. A medida que los agentes se encuentren más dispersos en el mapa, la mejora de A^*mbush con respecto a A^* no es significativa, debido a que es menos probable que existan nodos o arcos comunes en los caminos de los agentes.

En cambio, cuando los agentes se encuentran concentrados en regiones pequeñas del mapa, A^*mbush genera una mejora significativa en el valor de Φ.

Es importante destacar que el incremento en el costo de los caminos obtenidos por A^*mbush no crece indefinidamente, como se puede ver en los experimentos globales, dado que cuando los agentes abarcan todos los arcos del grafo, los incrementos por nodo o por arco tienden a ser homogéneos.

Una gran ventaja de A^*mbush es que, como favorece a una rápida dispersión de los agentes a lo largo del mapa, si el objetivo a alcanzar se encuentra en movimiento, se genera un mayor número de posiciones de emboscada.

En la figura 9 se puede contrastar el resultado de los tres mecanismos planteados para un caso particular. Nótese que en el caso de A^* los tres agentes llegaron al destino desde el mismo sector utilizando sus caminos más cortos. Para el aleatorio, el comportamiento fue similar al caso de A^*, sin embargo tomaron todos un camino significativamente más largo. En cambio, A^*mbush logró que uno de los agentes tomara un camino distinto, logrando alcanzar al objetivo desde los dos flancos posibles.

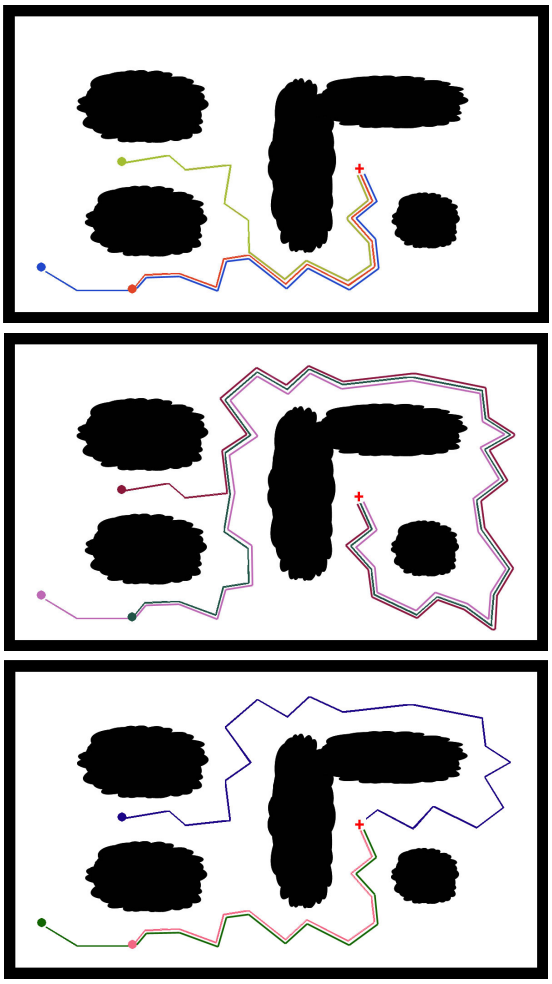

Figura 9: *Caminos generados por tres agentes utilizando respectivamente A^*, pseudo-aleatorio y A^*mbush (de arriba hacia abajo).*

8. Conclusiones

Con un aumento en el costo de cómputo poco significativo, A^*mbush genera una evidente mejora en la conducta de emboscada. También diversifica exitosamente los caminos que seleccionan los agentes.

Como se puede apreciar en los resultados, en el peor de los casos el algoritmo propuesto no cambia el grado de emboscada, es decir, aún en la situación menos favorable, A^*mbush nunca genera un peor resultado que el obtenido con A^*.

Al diversificarse los caminos, los agentes naturalmente se distribuyen con mayor uniformidad en el mapa. Esto puede ser ventajoso en el caso de un cambio de estado a una nueva conducta, en el que sea conveniente que los agentes no estén conglomerados en un punto. Por ejemplo, si se cambia de conducta de persecución a huída, A^*mbush no sólo permite un comportamiento de emboscada, sino que mantiene a los agentes más alejados los unos de los otros, dejándolos en una posición más adecuada para escapar.

A^*mbush genera comportamientos de emboscada inteligentes para situaciones en las que múltiples agentes necesitan alcanzar un objetivo común. En contraste con estrategias preestablecidas, regularmente implementadas para situaciones específicas de cada juego, que derivan en situaciones repetitivas, A^*mbush logra fomentar situaciones variadas de emboscada dentro del comportamiento táctico grupal de los agentes.

8.1. Aplicación a un Videojuego

A^*mbush ha sido implementado en el juego FatBoy para la materia "Tópicos en Inteligencia Artificial: Juegos" de la Universidad Simón Bolívar. El juego puede ser consultado en http://www.gia.usb.ve/wikis/iajuegos/Kelwin%20Fernández.

En http://www.gia.usb.ve/∼kelwin/research/sctc2012/ pueden encontrarse casos de interés adicionales.

9. Otras Aplicaciones

La diversificación de caminos generada por A^*mbush no se limita a su aplicación al área de videojuegos.

En el caso de las redes, A^*mbush puede utilizarse para generar menor congestión en el envío de paquetes, generando una mayor diversidad en el uso de la red [TMSV03]. Los enrutadores pueden sincronizar cada cierto tiempo su información con los demás, para poder tomar en cuenta la cantidad de paquetes en cada punto. De esta forma es posible generar un uso más uniforme de los canales de una red.

A^*mbush también consigue una natural aplicación en la planificación del tráfico de vehículos. Utilizando la segunda versión propuesta (incremento de costos en arcos), se puede generar una planificación favorezca el uso de rutas más largas y poco transitadas, para aligerar el embotellamiento general.

En el desarrollo de algoritmos heurísticos de optimización, tales como GRASP, Colonia de Hormigas, entre otros [GP10], es posible generar una mayor diversificación en el espacio de búsqueda utilizando variaciones de A^*mbush adaptadas al contexto del problema.

Finalmente, la variación expuesta en el presente trabajo no se limita al algoritmo particular de A^*. La idea central de A^*mbush es el cómputo de una nueva función de costos que penalice los sectores del grafo más utilizados, por lo que el mismo enfoque puede ser implementado en otros algoritmos de cómputo de caminos en grafos con costos, tales como IDA^*, *Fringe Search*, *Branch and Bound*, entre otros.

10. Trabajos Futuros

Debido a la naturaleza del algoritmo, es posible que un agente que se encuentra cerca del objetivo, deba desviarse dado que otros agentes han tomado arcos de interés para éste previamente. Se explorarán variaciones que incluyan prioridades a los agentes para reducir el incremento porcentual del costo de los caminos obtenidos. También se explorará la posibilidad de incorporar selectivamente el algoritmo de A*mbush, siguiendo criterios tales como la cercanía del agente al objetivo, con el fin de minimizar el incremento promedio de los caminos.

Igualmente, se trabajará incluyendo capacidad máxima a los nodos y arcos. Nótese que esta variación responde al problema propuesto en [Sil06] cuando cada arco tiene capacidad unitaria.

Adicionalmente, se implementará una heurística que le permita a los agentes sólo utilizar A^*mbush cuando estén dentro de cierto radio cercano al objetivo. Logrando de esta manera reducir el costo, tanto de cálculo como el de los caminos. Esta heurística también puede adaptarse para generar distintas dificultades de juego.

Referencias

[CLRS09] CORMEN T., LEISERSON C., RIVEST R., STEIN C.: *Introduction to Algorithms*, 3rd ed. The MIT Press, 2009. 2

[CS11] CUI X., SHI H.: Direction oriented pathfinding in video games. *International Journal of Artificial Intelligence & Applications 2* (2011). 3

[GP10] GENDREAU M., POTVIN J.: *Handbook of Metaheuristics*, 2nd ed., vol. 146. International Series in Operations Research & Management Science, Springer, 2010. 8

[Hai01] HAINES E.: Point in Polygon Strategies. http://http://erich.realtimerendering.com/ptinpoly/, 2001. [Online; accedida 26-Noviembre-2011]. 4

[HNR72] HART P., NILSSON N., RAPHAEL B.: Correction to "a formal basis for the heuristic determination of minimum cost paths". *SIGART Newsletter 37* (1972), 28–29. 1, 2, 3

[MF09] MILLINGTON I., FUNGE J.: *Artificial Intelligence for Games*, 2nd ed. Morgan Kaufmann Publishers, 2009. 1, 2, 3, 4, 5

[RN93] RUSSELL S., NORVIG P.: *Artificial Intelligence: A Modern Approach*, 2nd ed. Prentice-Hall, Englewood Cliffs, NJ, 1993. 1, 2

[Sil06] SILVER D.: Cooperative pathfinding. *AI Game Programming Wisdom 3* (2006), 99–111. 4, 8

[TMSV03] TEIXEIRA R., MARZULLO K., SAVAGE S., VOELKER G.: In search of path diversity in isp networks. *IMC* (October 2003), 27–29. 1, 8