**Fase 1:** Identificación del problema

El problema planteado en el contexto del proyecto "Graph Ball Z" se centra en el desarrollo de un juego que involucra la representación y el uso de la teoría de grafos para crear una experiencia de juego desafiante y entretenida. El objetivo principal del juego es guiar al Goku(el personaje principal), con muchas ganas de venganza y de salvar el universo de los poderoso enemigos de otros planetas, a través de una galaxia ficticia compuesta por cinco mapas, en los que se encontraran diversos planetas y para pasar al siguiente mapa, Goku deberá enfrentar un enemigo final quien es el que cuida de sus planetas. El desafío radica en encontrar el camino óptimo para coger desprevenidos a los jefes finales y asi derrotarlos. Este juego plantea diversas necesidades clave que deben abordarse para lograr su implementación exitosa.

Las necesidades planteadas en este contexto son reales y fundamentales para el desarrollo del juego "Graph Ball Z", incluyen la creación de una representación precisa y eficiente de la galaxia en forma de grafo, la implementación de al menos dos algoritmos de grafos (por ejemplo, BFS y Dijkstra) para encontrar rutas óptimas y derrotar el enemigo final de cada mapa, y la construcción de una interfaz gráfica de usuario que permita a los jugadores interactuar con el juego de manera efectiva.

En resumen, el problema identificado se relaciona con la creación de un juego basado en la teoría de grafos, que requiere la implementación de al menos dos algoritmos de grafos, una representación eficiente de la galaxia la cual tiene 5 mapas y una interfaz gráfica de usuario. Estas necesidades se han definido a grandes rasgos y se han diferenciado de las soluciones específicas, evitando prejuicios en la definición del problema. El enfoque se centra en la creación de una experiencia de juego desafiante y entretenida para los jugadores.

**Fase 2:**  Recopilación de información

Con fin de suplir todas las necesidades actuales y las posibles necesidades emergentes del problema en cuestión, la búsqueda empezará por los siguientes conceptos: Grafo, Vértices y Aristas, Grafo Dirigido y No Dirigido, Grafo Ponderado y No Ponderado, Ciclo y Camino, Grado de un Vértice, Conectividad, Grafo Acíclico, Árbol, Árbol de Expansión Mínima (MST), Algoritmos de Búsqueda, Algoritmos de Caminos Más Cortos, Algoritmos de Flujo Máximo, Teorema de Euler y Hamilton, Aplicaciones en la Vida Real.

Para esto tomaremos varías páginas referencia con fin de clarificar la teoría y comprender su implementación real dentro del entorno que estamos tratando:

<https://ccc.inaoep.mx/~villasen/CursoMatDiscretas/Grafos.pdf>

<https://elpixelamplificado.wordpress.com/2020/04/07/las-matematicas-de-los-videojuegos-los-grafos/>

<https://www.freecodecamp.org/espanol/news/algoritmo-de-la-ruta-mas-corta-de-dijkstra-introduccion-grafica/#:~:text=El%20algoritmo%20de%20Dijkstra%20encuentra,los%20dem%C3%A1s%20nodos%20del%20grafo>.

<https://e-archivo.uc3m.es/bitstream/handle/10016/23954/TFG_Alvaro_Parra_De_Miguel_2015.pdf?sequence=1&%3BisAllowed=y>(Sección algoritmos de Búsqueda).

<https://core.ac.uk/download/pdf/421934731.pdf>(Sección Teoría de grafos (2.2))

**Fase 3: Conceptos y definiciones recopiladas:**

1. ***Grafo***: Un grafo es una estructura matemática que consta de un conjunto de nodos (vértices) y un conjunto de aristas (bordes) que conectan estos nodos. Los grafos se utilizan para representar relaciones entre objetos.

2***. Vértices y Aristas***: Los vértices (nodos) son los puntos en un grafo, y las aristas (bordes) son las conexiones entre los vértices. Las aristas pueden ser dirigidas (de un vértice a otro) o no dirigidas.

3. ***Grafo Dirigido y No Dirigido***: En un grafo dirigido, las aristas tienen una dirección, es decir, van desde un vértice de origen a un vértice de destino. En un grafo no dirigido, las aristas no tienen una dirección específica.

4. ***Grafo Ponderado y No Ponderado***: Un grafo ponderado asigna un peso o valor a cada arista, lo que puede representar distancias, costos o cualquier otro atributo relevante. En un grafo no ponderado, todas las aristas tienen el mismo valor.

5. ***Ciclo y Camino***: Un ciclo es una secuencia de vértices y aristas que comienza y termina en el mismo vértice. Un camino es una secuencia de vértices y aristas que conecta dos vértices diferentes.

6***. Grado de un Vértice***: El grado de un vértice es el número de aristas que se conectan a ese vértice. En un grafo dirigido, se distingue entre el grado de entrada (aristas que llegan al vértice) y el grado de salida (aristas que salen del vértice).

7. ***Conectividad***: Un grafo se considera conectado si hay un camino entre cualquier par de vértices. Un grafo no conectado se divide en componentes conectados.

8. ***Grafo Acíclico***: Un grafo acíclico es aquel que no contiene ciclos, es decir, no se puede seguir un camino cerrado.

9. ***Árbo***l: Un árbol es un grafo no dirigido, conectado y acíclico. Un árbol es una estructura jerárquica que se utiliza en muchas aplicaciones, incluidas las estructuras de datos.

10. ***Árbol de Expansión Mínima*** (Minimum Spanning Tree - MST): Un MST de un grafo no dirigido y ponderado es un subgrafo que contiene todos los vértices y la menor cantidad de aristas posible para conectar todos los vértices sin formar ciclos.

11. ***Algoritmos de Búsqueda***: Los algoritmos de búsqueda en grafos, como BFS (Breadth-First Search) y DFS (Depth-First Search), se utilizan para explorar y buscar información en un grafo.

12. ***Algoritmos de Caminos Más Cortos***: Algoritmos como Dijkstra y Bellman-Ford se utilizan para encontrar el camino más corto entre dos vértices en un grafo ponderado.

13. ***Algoritmos de Flujo Máximo***: Estos algoritmos se utilizan en problemas de redes para encontrar el flujo máximo a través de una red de nodos y aristas.

14. ***Teorema de Euler y Hamilton***: Estos teoremas se relacionan con circuitos y caminos en grafos y proporcionan propiedades interesantes.

3.) ***Búsqueda de Soluciones Creativas***

Representación grafica :

Mapa 1:

***P1 -- P2***

***| / |***

***P3 -- P4***

***| |***

***P5 -- P6 -- P7***

***| / /***

***P8 -- P9 -- P10***

Mapa 2:

**P1 -- P2 -- P3**

**| / |**

**P4 -- P5**

**| |**

**P6 -- P7**

**| / /**

**P8 -- P9 -- P10**

Mapa 3:

**P1 -- P2 -- P3 -- P4**

**| / | / |**

**P5 -- P6 -- P7**

**| / / /**

**P8 -- P9 -- P10**

En esta representación, los | representan las aristas verticales que conectan los planetas en cada mapa de la galaxia, y los -- representan las aristas horizontales que conectan los planetas en el mismo mapa. Cada planeta está etiquetado como P1, P2, P3, y así sucesivamente. Lo anterior simplemente fue una representación, pero se logra ver un avance en la complejidad de las conexiones del grafo conforme se va avanzando de nivel

***Representación de la Galaxia como un Grafo:***

La galaxia se representa como un grafo G = (V, E), donde V es el conjunto de vértices (P1 a P10) y E es el conjunto de aristas (conexiones). Esto proporciona una estructura matemática para la galaxia.

***Búsqueda de Rutas Óptimas:***

Supongamos que aplicamos el algoritmo BFS (Breadth-First Search) desde la ubicación actual de Goku (por ejemplo, S1) al enemigo final del mapa (por ejemplo, S10) en el segundo mapa. Este algoritmo calcula la ruta más corta considerando el peso de las aristas (representando la dificultad de pasar en presencia de diferentes enemigos no visibles por cada planeta).

***Dinámica del Juego y Aumento de la Dificultad:***

A medida que Goku avanza de un mapa a otro, se pueden agregar más aristas a los vértices y aumentando así el peso de las aristas correspondiente. Esto se puede representar matemáticamente al actualizar los valores de peso en las aristas a medida que Goku avanza.

***Otras alternativas un poco mas avanzadas*** que involucran inteligencia artificial y algoritmos mas avanzados y que darían solución al juego son:

Propuesta tentativa:

***Algoritmo A:\****

En lugar de usar Dijkstra para encontrar la ruta más corta, podrías implementar el algoritmo A\*. A\* es una variante de búsqueda informada que utiliza una heurística para estimar la distancia restante hasta el objetivo. Puede proporcionar rutas más eficientes en términos de tiempo de ejecución que Dijkstra.

***Algoritmo Genético:***

Podríamos explorar la implementación de algoritmos genéticos para que el Goku "aprenda" a recorrer los planetas de una manera más efectiva en el transcurso del juego. Esto introduciría la inteligencia artificial en el juego.

***Algoritmo de Recorrido del Vendedor Viajero (TSP):***

En lugar de buscar una ruta simple hacia el enemigo final, podríamos formular el juego como un problema del TSP, donde Goku debe visitar todos los planetas antes de llegar a su contrincate. Esto haría que el juego sea más complejo y desafiante.

***Algoritmos de Redes Neuronales Artificiales:***

Podríamos implementar redes neuronales artificiales para modelar el comportamiento de los enemigos y ajustar su "inteligencia" durante el juego. Esto permitiría que los enemigos se vuelvan mas astutos en el combate, como por ejemplo que adquieran el poder de quitarle algunas conexiones a los planetas, dificultando asi la llegada de Goku a su posición.

***Algoritmo de Evolución de Comportamiento:***

Implementaríamos un algoritmo de evolución de comportamiento que permita a los enemigos desarrollar estrategias de ataque y de perseguir a Goku, con el fin de defensa. Los enemigos pueden aprender y mejorar su estrategia con el tiempo.

***Variación en las Reglas del Juego:***

Modificaríamos las reglas del juego a medida que Goku r avanza, como agregar elementos de juego, power-ups o obstáculos que cambian dinámicamente. Esto puede agregar variedad y emoción al juego.

**Fase 4: Transición de las Ideas a los Diseños Preliminares**

Las ideas que descartamos son las siguientes:

**Algoritmo Genético:**

Razón para Descarte: Implementar un algoritmo genético para que Goku "aprenda" a recorrer los planetas puede resultar en una complejidad innecesaria para la dinámica del juego. Podría requerir una cantidad significativa de ajustes y pruebas para lograr un equilibrio adecuado.

**Algoritmo de Recorrido del Vendedor Viajero (TSP):**

Razón para Descarte: Formular el juego como un problema del TSP podría complicar excesivamente la jugabilidad. La necesidad de visitar todos los planetas antes de llegar al enemigo final podría restar diversión al juego.

**Algoritmos de Redes Neuronales Artificiales:**

Razón para Descarte: Implementar redes neuronales para modelar el comportamiento de los enemigos podría aumentar considerablemente la complejidad del juego sin necesidad. Además, podría requerir un extenso entrenamiento y ajuste de parámetros.

**Algoritmo de Evolución de Comportamiento:**

Razón para Descarte: Aunque la idea de permitir que los enemigos desarrollen estrategias adaptativas es interesante, puede resultar desafiante de implementar y puede no ser esencial para la experiencia central del juego.

**Diseño Preliminar para las Ideas No Descartadas en "Graph Ball Z"**

**Representación de la Galaxia como un Grafo:**

Diseño Preliminar:

Utilizar una estructura de datos de grafo para representar la galaxia, donde cada vértice es un planeta y las aristas representan las conexiones entre ellos.

Asignar pesos a las aristas para reflejar la dificultad de pasar entre los planetas, considerando la presencia de enemigos no visibles en cada planeta.

Implementar una interfaz gráfica para visualizar la galaxia y las rutas tomadas por Goku.

**Búsqueda de Rutas Óptimas ():**

Diseño Preliminar:

Aplicar el algoritmo ejemplo: BFS para encontrar la ruta más corta desde la posición actual de Goku hasta el enemigo final en cada mapa.

Considerar el peso de las aristas en la búsqueda para reflejar la dificultad y los obstáculos presentes en el camino.

Visualizar la ruta óptima en la interfaz gráfica del juego.

**Dinámica del Juego y Aumento de la Dificultad:**

Diseño Preliminar:

A medida que Goku avanza de un mapa a otro, agregar aristas adicionales a los vértices y aumentar el peso de las aristas correspondientes.

La dificultad aumentará proporcionalmente al número de mapas superados, introduciendo nuevos desafíos y obstáculos en el camino de Goku.

Ajustar dinámicamente la interfaz gráfica para reflejar la evolución de la galaxia.

**Algoritmo A:\***

Diseño Preliminar:

Evaluar la implementación del algoritmo A\* como alternativa al BFS para encontrar rutas más eficientes.

Utilizar heurísticas para estimar la distancia restante hasta el objetivo y mejorar la velocidad de búsqueda.

Proporcionar al jugador la opción de elegir entre BFS y A\* según sus preferencias.

**Variación en las Reglas del Juego:**

Diseño Preliminar:

Introducir elementos dinámicos en el juego, como power-ups que podrían facilitar temporalmente la navegación de Goku o obstáculos que complican su movimiento.

Establecer reglas específicas para cada mapa, manteniendo el juego fresco y desafiante a medida que progresa.

**Fase 5: Evaluación y Selección de la Mejor Solución para "Graph Ball Z"**

Criterios de Evaluación:

1.Complejidad Técnica:

* Valor Numérico: 4
* Justificación: Se busca una solución técnica que sea desafiante pero aún así manejable en términos de desarrollo y mantenimiento. Una complejidad moderada permite una implementación efectiva sin sacrificar la jugabilidad.

2.Experiencia del Usuario:

* Valor Numérico: 5
* Justificación: La satisfacción del jugador es crucial. Se valora una interfaz intuitiva, fácil de entender y atractiva visualmente. La experiencia del usuario debe ser priorizada para garantizar que el juego sea atractivo y accesible.

3.Dinámica del Juego:

* Valor Numérico: 5
* Justificación: La evolución dinámica del juego es esencial. A medida que Goku avanza, la dificultad y los obstáculos deben aumentar de manera proporcionada, manteniendo el interés del jugador y ofreciendo un desafío continuo.

4.Eficiencia en la Búsqueda de Rutas:

* Valor Numérico: 4
* Justificación: La eficiencia de los algoritmos de búsqueda es vital. Se valora la capacidad de encontrar rutas óptimas en un tiempo razonable. Tanto BFS como A\* deben cumplir con estándares de rendimiento.

5.Variabilidad en las Reglas del Juego:

* Valor Numérico: 3
* Justificación: Se busca una variabilidad que aporte emoción, pero sin caer en la complejidad excesiva. Debe haber suficientes elementos dinámicos para mantener la frescura del juego, pero sin confundir al jugador.

6.Inteligencia Artificial (Opcional):

* Valor Numérico:
* Justificación: La implementación de características de inteligencia artificial, como aprendizaje o evolución de comportamiento, es opcional. Se considera deseable, pero no esencial para la experiencia principal del juego.

Proceso de Evaluación:

* Cada solución propuesta será evaluada en función de los criterios mencionados.
* Se asignarán valores numéricos a cada criterio, donde un valor mayor indica una mejor satisfacción del criterio.
* La puntuación total de cada solución se calculará sumando los valores ponderados de cada criterio.

|  |
| --- |
| **Representación de la Galaxia como un Grafo:** |
| * Complejidad Técnica: 3 |
| * Experiencia del Usuario: 4 |
| * Dinámica del Juego: 3 |
| * Eficiencia en la Búsqueda de Rutas: 3 |
| * Variabilidad en las Reglas del Juego: 2 |
| * Inteligencia Artificial (Opcional): 1 |
| **Puntuación Total: 16** |

|  |
| --- |
| **Búsqueda de Rutas Óptimas (BFS):** |
| * Complejidad Técnica: 3 |
| * Experiencia del Usuario: 4 |
| * Dinámica del Juego: 4 |
| * Eficiencia en la Búsqueda de Rutas: 4 |
| * Variabilidad en las Reglas del Juego: 2 |
| * Inteligencia Artificial (Opcional): 1 |
| **Puntuación Total: 18** |

|  |
| --- |
| **Dinámica del Juego y Aumento de la Dificultad:** |
| * Complejidad Técnica: 4 |
| * Experiencia del Usuario: 5 |
| * Dinámica del Juego: 5 |
| * Eficiencia en la Búsqueda de Rutas: 3 |
| * Variabilidad en las Reglas del Juego: 3 |
| * Inteligencia Artificial (Opcional): 2 |
| **Puntuación Total: 22** |

|  |
| --- |
| **Algoritmo A:\*** |
| * Complejidad Técnica: 4 |
| * Experiencia del Usuario: 4 |
| * Dinámica del Juego: 4 |
| * Eficiencia en la Búsqueda de Rutas: 5 |
| * Variabilidad en las Reglas del Juego: 2 |
| * Inteligencia Artificial (Opcional): 1 |
| **Puntuación Total: 20** |

|  |
| --- |
| **Variación en las Reglas del Juego:** |
| * Complejidad Técnica: 3 |
| * Experiencia del Usuario: 4 |
| * Dinámica del Juego: 4 |
| * Eficiencia en la Búsqueda de Rutas: 3 |
| * Variabilidad en las Reglas del Juego: 4 |
| * Inteligencia Artificial (Opcional): 2 |
| **Puntuación Total: 20** |

**Elección de la Mejor Solución:**

La Idea No. 3, "Dinámica del Juego y Aumento de la Dificultad", es seleccionada como la mejor solución debido a su puntuación total más alta y su capacidad para cumplir con múltiples criterios clave.