Supongamos que tenemos un individuo representado por el vector [1, 0, 2, -1, 1]. Esto se interpretaría de la siguiente manera:

1. Primer gen (1): Corresponde al punto negro A.
   * El valor 1 indica que estamos eligiendo la segunda combinación de recursos para el punto negro A: (2 operarios, 0 grúas, 2 hormigoneras).
2. Segundo gen (0): Corresponde al punto negro B.
   * El valor 0 indica que estamos eligiendo la primera combinación de recursos para el punto negro B: (1 operario, 2 grúas, 0 hormigoneras).
3. Tercer gen (2): Corresponde al punto negro C.
   * El valor 2 indica que estamos eligiendo la tercera combinación de recursos para el punto negro C: (8 operarios, 0 grúas, 0 hormigoneras).
4. Cuarto gen (-1): Corresponde al punto negro D.
   * El valor -1 indica que no estamos asignando recursos para resolver el punto negro D.
5. Quinto gen (1): Corresponde al punto negro E.
   * El valor 1 indica que estamos eligiendo la segunda combinación de recursos para el punto negro E: (2 grúas, 2 operarios, 0 hormigoneras).

### **Interpretación de los Valores**

* Cada valor en el vector puede ser:
  + Un número entero que indica la elección de una combinación específica de recursos para el punto negro correspondiente.
  + -1 para indicar que no se asignan recursos a ese punto negro (es decir, ese punto negro no se aborda en ese individuo).

### **Requisitos de Recursos para Cada Punto Negro**

Los requisitos de recursos para cada punto negro se definen en una estructura de datos (por ejemplo, un diccionario) como se muestra a continuación:

puntos\_negros = {

'A': [(4, 0, 0), (2, 0, 2)], # (operarios, grúas, hormigoneras)

'B': [(1, 2, 0), (6, 0, 0)],

'C': [(1, 1, 1), (3, 0, 1), (8, 0, 0)],

'D': [(5, 0, 2), (9, 0, 0)],

'E': [(4, 1, 0), (2, 2, 0)]

}

### **Evaluación de la Aptitud**

La función de aptitud se encarga de calcular cuántos puntos negros se pueden resolver con los recursos disponibles para cada individuo. Esta función considera los recursos totales disponibles (grúas, operarios y hormigoneras) y verifica si las asignaciones en el vector de cada individuo son viables sin exceder estos recursos. La aptitud se define como el número de puntos negros resueltos, penalizando las asignaciones que superen los recursos disponibles.

### **Ejemplo de Cálculo de Aptitud**

Para un individuo [1, 0, 2, -1, 1], la función de aptitud evaluaría los recursos necesarios para cada punto negro de la siguiente manera:

1. Punto negro A:
   * Asignación: (2 operarios, 0 grúas, 2 hormigoneras).
2. Punto negro B:
   * Asignación: (1 operario, 2 grúas, 0 hormigoneras).
3. Punto negro C:
   * Asignación: (8 operarios, 0 grúas, 0 hormigoneras).
4. Punto negro D:
   * No se aborda (-1).
5. Punto negro E:
   * Asignación: (2 operarios, 2 grúas, 0 hormigoneras).

El cálculo de aptitud comprobará si la suma de estos recursos no supera los recursos disponibles totales. Si es así, sumará el número de puntos negros resueltos.

Este enfoque garantiza una evaluación eficaz de las posibles soluciones, explorando combinaciones diversas hasta encontrar la asignación óptima de recursos para resolver el mayor número de puntos negros en un día.

EJERCICIO 2:

Para modelar un agente autónomo que sustituya al jugador en PacMan, podemos utilizar una máquina de estados finitos (FSM). Esta máquina de estados nos permitirá definir y gestionar el comportamiento del agente según las diferentes situaciones que puede enfrentar en el juego. A continuación, se detalla el diseño de la máquina de estados para el agente jugador en PacMan:

### **Estados de la Máquina de Estados**

1. Buscar Bolitas Pequeñas
2. Huir de Fantasmas
3. Comer Bolita Grande
4. Invulnerable - Perseguir Fantasmas
5. Cambiar Dirección
6. Comer Fruta
7. Reaparecer en el Centro

### **Transiciones entre Estados**

1. Buscar Bolitas Pequeñas
   * Transición a Comer Fruta: Si una fruta está a menos de 2 cuadros de distancia.
   * Transición a Huir de Fantasmas: Si detecta fantasmas cerca.
   * Transición a Comer Bolita Grande: Si hay una bolita grande más cerca que un fantasma.
   * Transición a Cambiar Dirección: Si choca con una pared.
2. Huir de Fantasmas
   * Transición a Comer Fruta: Si una fruta está a menos de 2 cuadros de distancia.
   * Transición a Buscar Bolitas Pequeñas: Si no hay fantasmas cerca.
   * Transición a Comer Bolita Grande: Si hay una bolita grande más cerca que un fantasma.
   * Transición a Cambiar Dirección: Si choca con una pared.
3. Comer Bolita Grande
   * Transición a Invulnerable - Perseguir Fantasmas: Si se come la bolita grande.
   * Transición a Cambiar Dirección: Si choca con una pared.
   * Transición a Buscar Bolitas Pequeñas: Si no hay bolita grande cerca.
4. Invulnerable - Perseguir Fantasmas
   * Transición a Comer Fruta: Si una fruta está a menos de 2 cuadros de distancia.
   * Transición a Buscar Bolitas Pequeñas: Si se acaba el tiempo de invulnerabilidad.
   * Transición a Cambiar Dirección: Si choca con una pared.
5. Cambiar Dirección
   * Transición a Buscar Bolitas Pequeñas: Seleccionar una nueva dirección aleatoria y moverse.
   * Transición a Huir de Fantasmas: Si detecta fantasmas cerca después de cambiar de dirección.
   * Transición a Comer Bolita Grande: Si hay una bolita grande cerca después de cambiar de dirección.
   * Transición a Invulnerable - Perseguir Fantasmas: Si tiene invulnerabilidad después de cambiar de dirección.
6. Comer Fruta
   * Transición a Buscar Bolitas Pequeñas: Después de comerse la fruta.
   * Transición a Cambiar Dirección: Si choca con una pared.
7. Reaparecer en el Centro
   * Transición a Buscar Bolitas Pequeñas: Después de reaparecer en el centro del escenario.

El agente descrito para el videojuego PacMan es un agente basado en objetivos.

### **Razonamiento:**

1. Definición de Agente Basado en Objetivos:
   * Un agente basado en objetivos actúa para cumplir con uno o más objetivos específicos.
   * Evalúa sus acciones basándose en el impacto que estas tienen en el logro de sus objetivos.
   * A menudo, tiene la capacidad de tomar decisiones considerando diferentes estados del entorno para alcanzar el objetivo deseado.
2. Comportamiento del Agente en PacMan:
   * Objetivo Principal: Comer todas las bolitas pequeñas del escenario para maximizar la puntuación.
   * Objetivo Secundario: Evitar los fantasmas para no perder vidas.
   * Subobjetivos: Comer bolitas grandes para volverse invulnerable y perseguir fantasmas; comer frutas cuando están cerca.
   * El agente toma decisiones basadas en la proximidad de fantasmas, bolitas grandes y frutas, evaluando cuál acción contribuye mejor a sus objetivos en cada momento.
3. Evaluación de las Acciones:
   * El agente evalúa constantemente su entorno (posiciones de las bolitas, fantasmas y frutas).
   * Cambia su comportamiento para maximizar la probabilidad de cumplir sus objetivos (evitar fantasmas, comer bolitas y frutas).
   * Por ejemplo, si detecta un fantasma cerca y hay una bolita grande más cerca, prioriza comer la bolita grande para obtener invulnerabilidad y cambiar el objetivo a perseguir fantasmas.
4. Adaptación y Decisión Basada en el Entorno:
   * La máquina de estados que modela el comportamiento del agente permite adaptarse a diferentes situaciones del entorno dinámico del juego.
   * Las transiciones entre estados (como huir de fantasmas, comer bolitas grandes, cambiar de dirección) son decisiones orientadas a cumplir sus objetivos.

En resumen, el agente en PacMan es un agente basado en objetivos porque su comportamiento se dirige a alcanzar objetivos específicos (comer bolitas pequeñas, evitar fantasmas, obtener invulnerabilidad, y comer frutas) y toma decisiones evaluando cómo las acciones disponibles contribuyen al logro de estos objetivos.



