Simulación Multiagente
Gestión de Proyectos.

Roger Fuentes Rodríguez
Kevin Manzano Rodríguez
Jackson Vera Pineda

September 18, 2024

1 Introducción

En este informe presentamos el desarrollo de una simulación multiagente basada en una arquitectura BDI (Beliefs, Desires, Intentions). El enfoque de esta simulación está en la interacción entre dos tipos de agentes: el **Project Manager (PM)** y los **Trabajadores**. Además, se modeló un entorno dinámico donde estos agentes interactúan con tareas, recursos, riesgos y oportunidades, todo representado mediante ontologías.

El objetivo principal fue simular la gestión de proyectos en un entorno de trabajo utilizando técnicas de inteligencia artificial, incluyendo **algoritmos genéticos**, **lógica difusa**, y **aprendizaje reforzado**, lo que permitió optimizar la asignación de tareas y la resolución de problemas en tiempo real.

2 Arquitectura BDI

El modelo BDI permite a los agentes razonar sobre sus creencias, deseos e intenciones, adaptándose a situaciones dinámicas y cambiantes en el contexto del proyecto. Los agentes toman decisiones basadas en su percepción del entorno y en los objetivos que se les asignan.

2.1 Beliefs (Creencias)

El agente Project Manager mantienen una representación interna de su entorno, que incluye información sobre el estado del proyecto, los recursos disponibles, los riesgos identificados y el progreso de las tareas.

Los agentes Trabajadores mantienen una representación interna de las tareas que tienen asignadas, el progreso en las mismas, la motivación del equipo y algunas relaciones con otros agentes.

2.2 Desires (Deseos)

Los deseos reflejan los objetivos que el agente intenta alcanzar, como completar un número máximo de tareas, evitar problemas o minimizar los recursos utilizados.

2.3 Intentions (Intenciones)

Las intenciones son los planes que el agente decide llevar a cabo en función de sus deseos, sus creencias y sus percepciones; como asignar recursos a tareas específicas o escalar problemas al Project Manager.

3 Algoritmo Genético para la Planificación de Tareas

El **Project Manager** utiliza un algoritmo genético para encontrar la mejor permutación posible de las tareas del proyecto debido a que no hay garantía en que exista una permutación en la que se cumplan todas las restricciones de las tareas. A continuación, describimos el proceso:

3.1 Población y Generación de Individuos

Se parte de una población inicial compuesta por permutaciones siguiendo una heurística **greedy** para garantizar como mínimo las dependencias entre las tareas. Cada individuo de la población representa una permutación de las tareas del proyecto. Obtenemos un orden topológico de las tareas, con una modificación en el momento de la elección de los nodos con indegree 0, este se elige de manera aleatoria. Luego de establecido el orden, los nodos quedan divididos en niveles, sobre estos niveles se realizan cambios de manera aleatoria.

3.2 Selección y Cruzamiento

Para seleccionar los individuos que pasarán a la siguiente generación, implementamos un método de **selección por torneo**, teniendo en cuenta un factor de elitismo que puede variar. Este método selecciona a los individuos

más aptos para cruzarse y producir nuevos descendientes. Posteriormente, los individuos se cruzan y las mutaciones se incorporan como nuevos individuos, lo que permite introducir variabilidad en la población. Para el cruce de dos permutaciones dadas, p_1, p_2, se elige un punto de corte aleatorio de la primera permutación luego se van agregando nodos en el orden de la segunda permutación que no incumplan con las dependencias, y cuando termina este proceso puede darse el caso de que falten elementos por colocar, por lo tanto se rellenan con elementos en el orden de la primera permutación pues ya en esta se ha garantizado que se cumplen las dependencias.

3.3 Mutación

La mutación en este contexto implica la creación de nuevas permutaciones de tareas, introduciendo cambios aleatorios menores en los individuos seleccionados para diversificar la búsqueda del óptimo global. Se prioriza la diversidad de la dificultad entre las tareas sucesivas, lo que permite un mejor balance en la distribución de las tareas. Dada una permutación p, se elige un nodo aleatorio, luego se "empuja" este nodo en los niveles del orden toplólogico, de manera que se garantice que no se rompan las dependencias (los nodos que dependen de el también se empujaran).

3.4 Función de Optimización

La función de optimización utilizada para evaluar la calidad de cada permutación de tareas incluye varios factores que pueden ajustarse mediante reglas de **lógica difusa**. Esto nos permitió modificar la función de optimización de acuerdo a los objetivos prioritarios del proyecto, ya que dichos objetivos pueden variar; por ejemplo otorgar mayor peso al número de tareas posibles a completar frente a la calidad de las mismas.

Se incorporaron penalizaciones basadas en el incumplimiento de dependencias entre tareas, el exceso de tareas de dificultad similar y las tareas que exceden su *deadline*. Además, se aplican bonificaciones por mantener una variedad de dificultades consecutivas y completar las tareas de alta prioridad a tiempo.

4 Interacción entre Agentes

Una vez que el **Project Manager** obtiene la mejor permutación de tareas, comienza la interacción con los **Trabajadores**. El PM asigna tareas a los

trabajadores, supervisa su desarrollo y ajusta las estrategias en función del progreso.

4.1 Asignación de Tareas Basada en Habilidades

La asignación de tareas se basa en las habilidades de los trabajadores y la dificultad de las tareas. El PM utiliza una heurística que busca minimizar la diferencia entre la dificultad de la tarea y la capacidad de resolución de problemas del trabajador, garantizando una asignación eficiente.

4.2 Generación de Hitos

El PM genera **hitos** basados en proyecciones del avance del proyecto. Estos hitos no son estáticos, sino que se generan dinámicamente según la situación actual del proyecto. Para esto, el agente evalúa el estado del proyecto utilizando **lógica difusa**, lo que le permite proyectar los hitos de manera flexible, ajustándose a cambios en el entorno. Dependiendo del estado del equipo (motivación, productividad, disponibilidad de recursos), los hitos pueden ser conservadores, normales o entusiastas.

5 Modelado del Proyecto con Ontologías

Para representar los proyectos y su estructura, utilizamos una **ontología** que incluye los siguientes componentes:

- **Recursos**: Elementos que el proyecto requiere para completarse.
- **Riesgos**: Posibles problemas que podrían surgir durante el proyecto, como retrasos en la ejecución o falta de personal.
- **Oportunidades**: Situaciones favorables que pueden mejorar el desempeño del proyecto, como alta motivación o ahorro de costos.
- **Tareas**: Actividades específicas que deben completarse, cada una con su duración, prioridad, dificultad y dependencias.

Este enfoque permite una representación estructurada y detallada del proyecto, facilitando la toma de decisiones por parte del PM, incluyendo la creación de hitos personalizados.

6 Procesamiento de Tareas desde Texto

Para modelar la entrada de tareas en formato de texto natural, utilizamos una **API de Geminis** con un *prompt* diseñado para convertir un texto de entrada en una lista estructurada de tareas, cada una con sus respectivos atributos y dependencias. Esto nos permitió automatizar la creación de tareas dentro de la simulación y garantizar que las tareas ingresadas manualmente siguieran una estructura consistente.

7 Resumen de la Simulación

Dado el contenido registrado de la simulación, se le envia esta información al LLM para que genere un informe que contenga el estado del proyecto, el progreso de las tareas y las recomendaciones para la siguiente iteración.

8 Aprendizaje Reforzado

El **Project Manager** no solo asigna tareas y supervisa a los trabajadores, sino que también incorpora un sistema de **aprendizaje reforzado**. A través de la experiencia, el PM aprende a optimizar la asignación de tareas basándose en las cualidades de los trabajadores y sus respuestas ante situaciones complejas. El aprendizaje reforzado permite que el PM mejore con el tiempo y optimice las decisiones de asignación de tareas de manera adaptativa, maximizando la eficiencia y el uso de los recursos del proyecto.

9 Estudio Estadístico

10 Conclusiones

La simulación multiagente con la arquitectura BDI nos permitió modelar un entorno de trabajo colaborativo donde los agentes toman decisiones complejas de manera autónoma. El uso de un algoritmo genético para la optimización de tareas, junto con la lógica difusa para la evaluación de las situaciones, resultó en una mayor flexibilidad y adaptabilidad del sistema. La incorporación de aprendizaje reforzado mejoró las capacidades del **Project Manager**, haciéndolo más eficiente en la gestión de proyectos y la optimización del uso de los recursos.