Comisión 3 - Lautaro Carranza y Leandro Mara.

Nota: A

Implementación:
Nota: A

.) Actualización de creencias:

- En primer lugar actualizan los at/2 y atPos/2 por ausencia de información.
  Recorren los nodos de la percepción, lo que puede resultar más costoso que recorrer los at de la base de creencias.
- Luego actualizan time/1, node/3 agregando los nodos que nunca percibieron.
- Para actualizar los at/2 y atPos/2:
  - Si ya tenían el mismo at no hacen nada
  - Si tenían un at distinto borran el viejo y agregan el nuevo
  - Si tenían un has/2 con esa entidad, borran el has y agregan el at.
  - En cualquier otro caso, simplemente agregan el at.

  Similarmente para atPos/2.
- Para los has/2, si viene en la percepción un has que ya tenían, no hacen nada. Si no, hacen un retractall de at por la entidad contenida, y agregan el has.
  En el informe dicen que si viene un has en la percepción y tenían uno diferente, entonces eliminan el viejo y agregan el nuevo. Eso es correcto pero la implementación está mal (Nunca va a entrar al tercer caso de rev(has(_,_)) porque el segundo nunca falla).
  También falta la actualización por ausencia de información de los has. Por ejemplo, si tenían en has(contenedora, contenida) en la base de creencia y en la nueva percepción viene información sobre la contenedora pero no viene el has, entonces significa que ese has ya no es verdad y debe ser borrado de la base de creencias del agente.
- La actualización del entity_descr/2 puede consistir simplemente en borrar y agregar la nueva. Como son pocos, puede ser igual o mejor que hacer comparaciones y chequeos.

.) Búsqueda:

- La implementación de la búsqueda correcta. En el caso base chequean que el primer nodo de la frontera sea meta. Si no lo es, generan los vecinos, los agregan a la frontera, luego ordenan la frontera por F(N) = C(N) + menor heurística y siguen la búsqueda con la nueva frontera y los visitados actualizados.
- En el agregar los vecinos a la frontera, consideran todos los casos necesarios, comparando el vecino con la frontera y con los visitados que traían.
- El ordenamiento de la frontera se hace con quicksort.
- En la generación de vecinos, para cada adyacente al nodo recuperan su vector, calculan el mejor valor de heurística a una meta y el costo que le asignan al vecino es el costo del padre + el costo del paso del padre a este adyacente + la heurística hallada.
- Utilizan una estructura nodo/3, donde llevan Id, costo y camino, y otra nodoM/2 donde llevan Id y vector, solo para las metas. Eso habría que explicarlo en el informe.
  En el caso de nodoM, capaz debería llamarse meta/2. También para destacar que cuando arman la lista de metas, con el predicado armarListaMetas/2, usan el vector del

nodo, no el vector de posición de la entidad (en estos casos las reliquias). Esto está bien, porque en definitiva la búsqueda los va a guiar hacia un nodo, no hacia la posición exacta de un objeto. Noten nomás que si quisieran usar el vector de la entidad, el vector de una reliquia por ejemplo, debería usar atPos/2 de esa entidad.

.) Decisión:

- Al decidir la siguiente acción, si tienen un plan de movimiento actual, utilizan dicho plan. Cuando no tienen un plan de movimiento, arman una lista de metas (donde hallan los nodos donde haya una reliquia) y buscan un plan de desplazamiento con esas metas.

Informe:
Nota: A

La explicación de la actualización de creencias es bastante simple y correcta.

Para la parte de búsqueda, está bien explicado, aunque no en detalle. Alcanza para ver que entienden cómo funciona la búsqueda, pero hubiera sido bueno que se metieran más en la implementación. Por ejemplo usan dos estructuras auxiliares, nodo/3 y nodoM/2, que no son mencionadas.

Pruebas con el agente:
Nota: A

Funciona de acuerdo a lo esperado. Se probó con más de un agente y todos se comportaron de manera correcta.

En un par de ocasiones sucedió que un agente falló repetidamente en realizar un movimiento. Simplemente revisen que no sea algo que ocurre todo el tiempo. En mi caso no.

Resumen:

+) Cumplen con lo pedido en la consigna.

-) Para corregir para la segunda etapa del proyecto:

1) Actualización de creencias: lo mencionado en la actualización del has.
2) Mejorar prolijidad en el código. Comentar adecuadamente lo que implementan, borrar comentarios que ya no sirvan, borrar predicados que no se usen (hay uno en agent_template.pl), mejorar indentación.
3) Para el próximo informe, releer a conciencia para detectar alguna equivocación que pudo surgir del corrector del editor de texto, o alguna oración que puede estar mal redactada o ser poco clara.