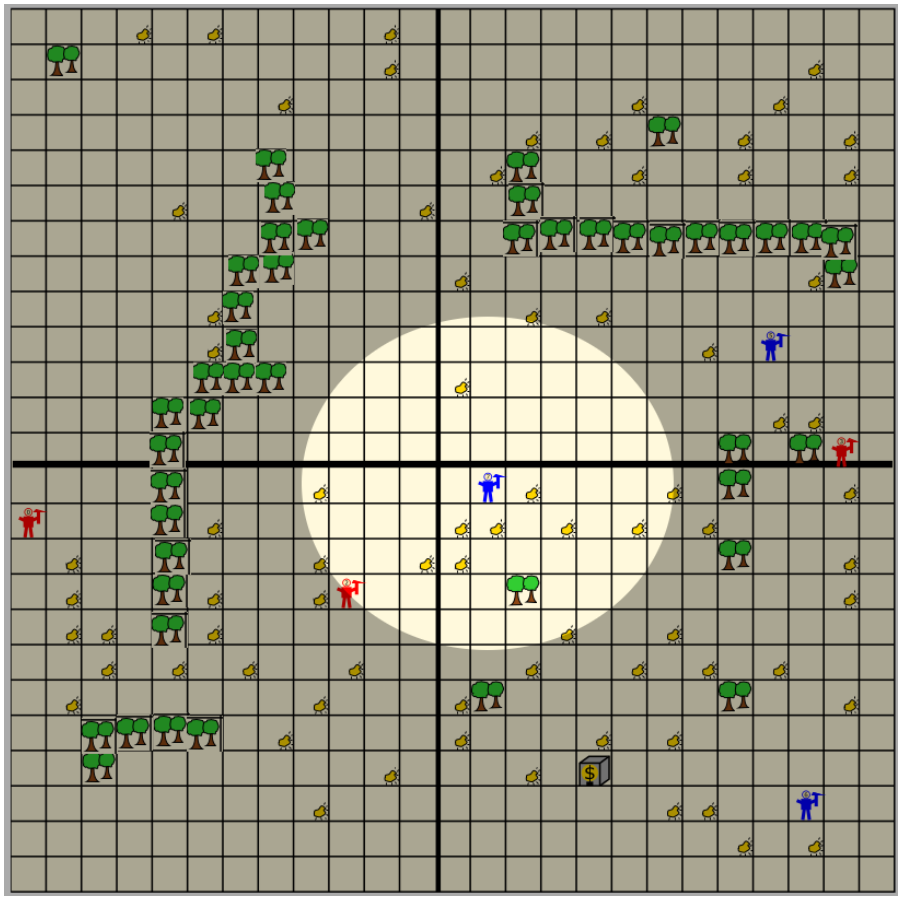
#2 Tutorial -- Designing a BDI agent in Java

14:30 - 16:00, 16:15-17:45

Running example: Goldminers

3 agents in a partially observable environment like this:



Percepts:

1. p = (x,y) agent's current position
2. G = {(x,y)} locations of gold stones the agent sees
3. A = {(x,y)} locations of other agents that the agent sees
4. F = {(x,y)} locations of forests the agent sees
5. d = (x,y) the location of the depot if the agents sees it
6. c the boolean flag determining whether the agent is carrying a gold stone

Actions:

{north, south, east, west, grab, drop, skip}

Rules:
- 1 point for each piece of gold carried to the depot
- trees and depot stay static
- agent cannot pass cells with forest, agent cannot pass cells occupied by other agents
- 1% of the actions will fill and end up with random effect

Implementing agents

Agent function:
Specifies the behavior of the agent: for each sequence of percepts, chooses an action
f: P* -> A

Rational agent:
Given some performance measure a rational agent always performs the action that maximizes its performance measure.

Perceive-Deliberate-Act cycle:

1. Perceive
2. Deliberate
3. Act

Agent architectures:

(According to Norvig and Russell)

1. Reflex (Reactive) Agent
    - chose action based on current percepts
    - (only consider the part of the world that is currently visible)
    - typically in form of if-then-rules

2. Model-based Reflex Agent
    - build model of the world that contains the expected state of the world that is unobserved
    - chose action based on the current model of the world

3. Model-based Goal-based Agent
    - the agent has a declaratively specified goal

    - performs actions that pursue the goal
    - Techniques:
        i. Classical planning (sequence of actions)
        ii. Planning with Uncertainty (policy)
        iii. Adversarial planning (policy)
        iv. Belief-Desire-Intention architecture (reactive planning -- can be used when full planning is not tractable)

4. Model-based Utility-based Agent
    - Each state of the world has a certain utility for the agent
    - Find sequence of actions (in deterministic env.) or policy (in non-deterministic env.) that maximizes the utility gathered over time
    - Techniques
        i. sequential decision making: MDP, POMDP (non-adversarial)
        ii. sequential games, imperfect information games (adversarial)

5. Learning-based Agent
    - The agent function is learnt through the interaction with the environment
    - Typically involves both learning the model and optimal policy
    - Techniques
        i. Reinforcement learning
            - agent receives percepts (observations) together with reward / penalty
            - learns transition table and Q-values: state X action -> R

Belief-Desire-Intention Architecture

Story:

- Programming rooted in psychology. Inspired by folk psychology (how people think other people think).
- Gave rise to specialized BDI languages such as Jason, 3APL, 2APL, GOAL, which however never really took off. Still not comparable to main-stream language in comfort of use.
- Could be implemented in any language
- Hoped to provide more computationally tractable way to compute intelligent behavior than full-scale planning, the behaviour is easier to understand to human and it is relatively easy to incorporate human expertise into the agent
- Many flavours, different authors, different opinions on why is BDI good and why is not

Main components (internal state divided into):

- Beliefs - what do I believe to hold in the world (model of the world)
- Desires - what would I like the world to look like (my goals)
- Intention - what goals am I committed to pursue
    - Unlike desires, intentions should satisfy some properties:
        - Agent should not intent something that he believes to be true
        - Agent should not intent something that he believes is unachievable
        - Agent should not intent something that is not desired
        - Intention should be **consistent**
        - Question: how long should an intention persist?
            - blind commitment: also referred to as fanatical commitment, the agent is intending the intention until it believes that it has been achieved (persistent intention)
            - single-minded commitment: besides above it intends the intention until it believes that it is no longer possible to achieve the goal
            - open-minded commitment: besides above it intends the intention as long as it is sure that the intention is achievable

Typical BDI Deliberation cycle

*** Process percepts / communication**

*** Deliberate about intentions/goals; drop the achieved ones**

*** Pursue intentions / apply rules**

How would you implement it in Java?