

**Software Engineering Department**

**Ort Braude College**

Course 61401: Extended Project in Software Engineering

**Open-Pit-Mining Operational Planning(OPMOP) using MAS**



Final Project in Software Engineering (Course 61401)

Carmiel – December 27, 2016

**Authors**

Ahdab Serhan

Bolous Abu Jaber

Contents

1. Introduction
   1. Abstract and keywords
   2. Introduction
   3. Problem definition
2. Theory
   1. Background
      1. Agent
      2. Multi Agent Systems
      3. Communication
      4. Negotiation
      5. Task Allocation
      6. Path planning
      7. Software models
   2. Related work
3. **Introduction**
   1. **Abstract and keywords**

This booklet aims to provide a preliminary research and design of a MAS approach to solving OPMOP.

**Keywords**: OPMOP, Pit Mines, MAS, Software, AI

* 1. **Introduction**

Open-pit mining is a technique used for extracting rocks and minerals from the earth by iteratively deepening the drilling layer by layer in a circular pattern, it differs from tunneling into the earth and is usually used to extract large deposits of minerals found close to the surface, or when the area itself is not suitable for tunneling. Heavy and expensive machinery is used to extract, refine and transport the minerals and waste to the surface. There are three main types of machines used in pit mining, Shovels, Loaders and hauling trucks, these machines also differ in specification, in terms of capacity, efficiency and rate. It is important to note that the number of machines is limited due to their high cost. Multi agent systems is a new paradigm for understanding and building distributed systems, it consists of multiple autonomous entities initiating a negotiation in order to achieve a goal. our project aims to explore and provide a software solution based on multi agent system, in order to achieve optimized task allocation and optimal efficiency and performance, Specifically, we aim to find a better solution for task planning and allocation in the dynamic environment of an open-pit mine under specific constrains and focusing on the goal of maximizing the performance of the mine.

* 1. **Problem definition**

The operational costs in open-pit mines consist of drilling, blasting, loading, haulage, dumping, and general services, the cost of each operation is extremely high, the machines involved are extremely costly and expensive.

The problem becomes even harder when we consider that there are machines of different productivities and their number is limited than the number of pits they can be allocated to. Also, the trucks used to transport the material (ore and waste rock) may have different capacities, speeds, and operational costs.

Haulage is generally the most expensive operation; large amounts of ore and waste must be delivered from inside the pits to the surface through relatively long and steep-grade haulage routes. It has been frequently reported that haulage costs account for more than 50% of the total operation cost.

Where TCT is the total Truck Cycle Time, LT is the loading time, TL is the travel time moving uphill, DT is the dumping time, TE is the travel time moving downhill (empty), and AD is the average delay time including both waits and delays. Since the truck cycle time is closely related to the total number of trucks that are required to complete the planned haulage work, the efficiency of the truck haulage operation can be improved if the time for each element of the truck cycle time is reduced. Improved efficiency is linked directly to the curtailment of haulage cost.

**The objective**

Open-Pit-Mining Operational Planning(OPMOP) problem deals with dynamic resources allocation including optimal task allocation and path planning. The objective is to optimize mineral extraction in the mines and lowering the costs by minimizing the quantity of mining equipment used and optimizing the tasks and paths each machine uses to meet production goals and quality requirements.

We are highly motivated to use MAS approach to solve the problem, where every machine is represented as an agent, and through collaboration and negotiation (auctions) we aim to reach a ‘very good’ solution for this NP-hard problem.

**Task Allocation**

We denote the set of agents as **A = {a1,...,a|A|}**, and the set of tasks as **T = {t1,...,t|T |}.** An agent ai ∈ A can perform any one task in **Ti ⊆ T**, the set of tasks ai can perform. Similarly, we denote the set of agents that can perform task tj as Aj.

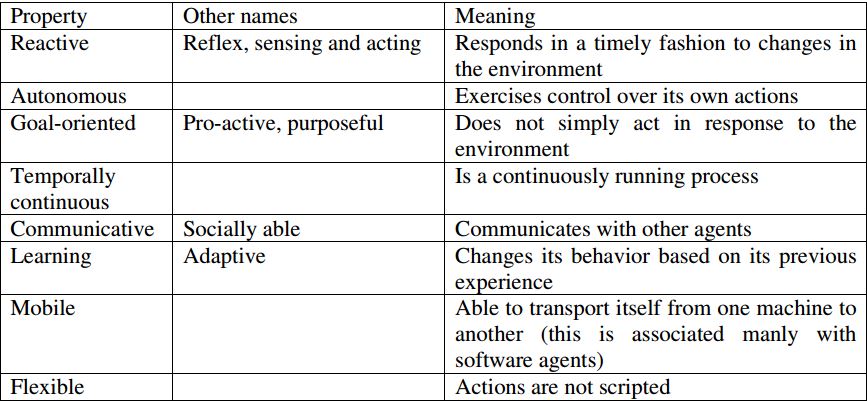
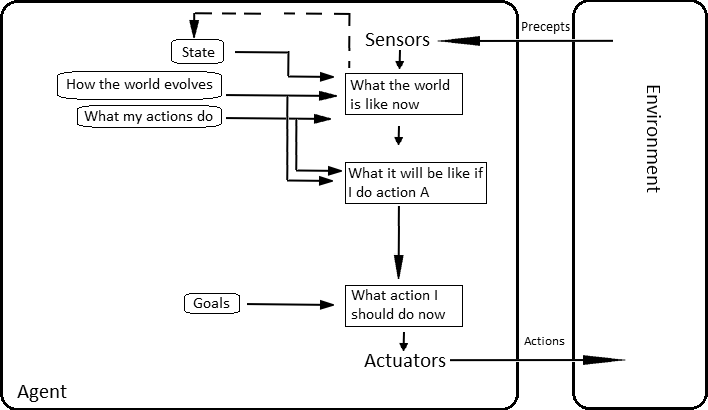
Moreover, we define the contribution of **ai to tj as δj (ai, Cj**), if coalition Cj performs tj, where **δj (ai, Cj) = V (Cj, tj) − V (Cj \ {ai}, tj**). Given this, and assuming that task utilities are independent.

our objective is to find the coalition structure **S∗ = {C∗ 1 ,...,C∗ |T |}** which maximizes the global utility:

1. **Theory**
   1. Background
      1. **Agent**

an intelligent agent (IA) is an [autonomous](https://en.wikipedia.org/wiki/Autonomous) software entity which observes its environment through sensors, reacts using actuators and directs its activity towards achieving a performance goal. an Intelligent agent may also [learn](https://en.wikipedia.org/wiki/Machine_learning) or use [knowledge](https://en.wikipedia.org/wiki/Knowledge_representation) to achieve their goals.

Following are a schematic view of an agent and a table that summarizes the properties and attributes of an intelligent agent.

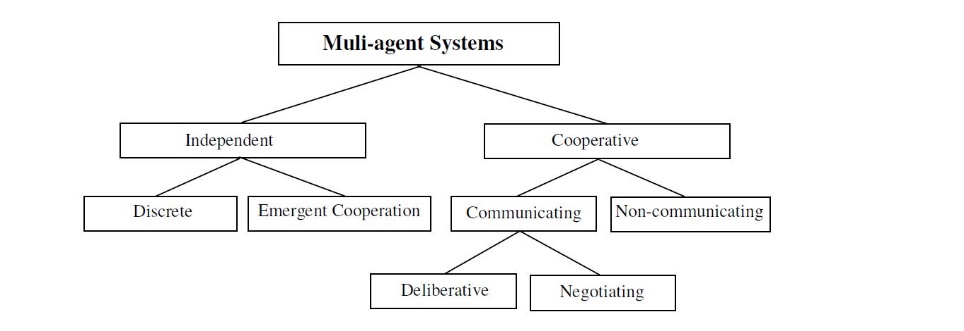


* + 1. **Multi Agent Systems**

Multi agent systems is a system that consists of multiple agents collaborating and negotiating to achieve a common goal, these systems are usually used to solve problems that are difficult or impossible for an individual monolithic agent to solve, these systems can provide a better solution and help reduce NP-Hard problems into decentralized distributed algorithms.

Agents in these systems have several important characteristics:

* Autonomy: the agents are at least partially independent, self-aware, [autonomous](https://en.wikipedia.org/wiki/Autonomous_agent)
* Local views: no agent has a full global view of the system, or the system is too complex for an agent to make practical use of such knowledge
* Decentralization: there is no designated controlling agent (or the system is effectively reduced to a monolithic system)



* Independent MAS: each individual agent pursues its own goals independently of the others.
* discrete MAS: agents are independent, and the goals of the agents bear no relation to one another. Discrete MAS involve no cooperation. However, agents can cooperate with no intention of doing so and if this is the case then the cooperation is emergent.
  + 1. **Advantages of MAS**

The main advantages of MAS are robustness and scalability. Robustness refers to the ability of sharing control and responsibilities among agents within MAS, the system can tolerate failures of one or more agents.

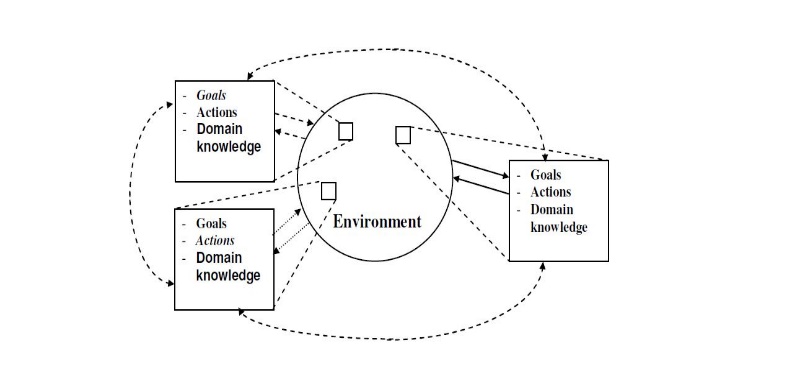
Scalability of MAS originates from its modularity. It should be easier to add new agents to a MAS than to add new capabilities to a monolithic system.

Moreover, MAS can provide solutions to problems that are impossible to solve in traditional ways, and it provides a more realistic representation of real life scenarios and environments.

* + 1. **Communication**

Individual agents (with different goals, actions, and /or domain knowledge) can communicate with one another in order to achieve collaboration or negotiation. The Two most important issues are: communication protocols and theories of commitment. A language set and a protocol should be defined for individual agents to use when communicating. Independent aspects of protocols are information content, message format, and coordination conventions.

Most of these protocols are based on Networks(Sockets) but really it can be anything, when agents communicate, they may decide to cooperate on a given task or for a given amount of time. In doing so, they make commitments to each other. Committing to another agent involves agreeing to pursue a given goal, possible in a given manner, regardless of how much it serves individual agent’s own interests.



* + 1. **Negotiation**

In systems composed of multiple autonomous agents, negotiation is a key form of interaction that enables groups of agents to arrive at a mutual agreement regarding some belief, goal or plan. The core of negotiation is reciprocal offer and counter-offer, argument and counter-argument in an attempt to agree upon outcomes mutually perceived as beneficial. Most of the protocols in negotiation are based on game theory, auctions is a great approach, where each agent proposes a local offer(solution) and the system allocates the task to the winner using some heuristic.

* + 1. **Task allocation.**

Task allocation is a fundamental problem in Multi-Agents system, given a set of complex Tasks, a set of agents, and deadlines. We want to find the best agent-task allocation to optimize the performance of the entire system. We also need the formation of a coalition of agents to cooperate together in order to carry out a complex task. yet finding the optimal allocation is NP-hard due to temporal and spatial constraints that require tasks to be executed sequentially by agents. the process of task allocation requires calculating the value of all the possible allocations, then determining which optimal.

* + 1. **Path planning**
    2. **Game theory**
    3. **Auctions**
    4. **ROS**
    5. **Software models**