# Multi-Agent Programming Contest Contest Scenario Description (2008 Edition)

**Revised 18.04.2008**

http://cig.in.tu-clausthal.de/agentcontest2008/

Tristan Behrens Mehdi Dastani Jürgen Dix
Peter Novák

## 1 Scenario: Cows and Herders

An unknown species of cattle was recently discovered in the unexplored flatlands of Lemuria. The cows have some nice features: their carbondioxyde- and methane-output is extremely low compared to the usual cattle and their beef and milk are of supreme quality and taste.

These facts definitely caught the attention of the beef- and milk-industries. The government decided to allow the cows to be captured and bred by everyone who is interested and has the capabilities. Several well-known companies decided to send in their personnel to the fields to catch as many of them as possible. This led to an unprecedented rush for cows. To maximise their success the companies replaced their traditional cowboys by *artificial herders*.

In this year's agent contest the participants have to compete in an environment for cows. Each team controls a set of herders in order to direct the cows into their own corral. The team with the most cows in the corral at the end wins the match.

### 1.1 General Description

Before the tournament, agent teams will be randomly divided into several groups if necessary. In the case of a small number of participating teams, these will form one single group.

Each team from one group will compete against all other teams in the same group in a series of matches. The winners from these groups form a new group. Each team in a new group will again play against all other teams in the group in a series of matches. A single match between two competing teams will consist of several (odd number of) simulations. A simulation between two teams

is a competition between them with respect to a certain configuration of the environment.

Winning a simulation yields 3 points for the team, a draw is worth 1 point and a loss 0 points. The winner of the whole tournament is evaluated on the basis of the overall number of collected points in all the matches during the tournament. In the case of equal number of points, the winner will be decided on the basis of the absolute number of captured cows.

Details on the number of simulations per match and the exact structure of the competition will depend on the number of participating teams and will be specified later.

In the contest, the agents from each participating team will be executed locally (on the participant's hardware) while the simulated environment, in which all agents from competing teams perform actions, is run on the remote contest simulation server run by the contest organizers.

The interaction/communication between agents from one team should be managed locally, but the interaction between individual agents and their environment (run on the simulation server) will be via Internet. Participating agents connect to the simulation server that provides the information about the environment.

Each agent from each team should connect and communicate to the simulation server using one TCP connection. After the initial phase, during which agents from all competing teams connect to the simulation server, identify and authenticate themselves and get a general match information, the competition will start. The simulation server controls the competition by selecting the competing teams and managing the matches and simulations. In each simulation, the simulation server, in a cyclic fashion, provides sensory information about the environment to the participating agents and expects their reactions within a given time limit.

Each agent reacts to the received sensory information by indicating which action (including the *skip* action) it wants to perform in the environment. If no reaction is received from the agent within the given time limit, the simulation server assumes that the agent performs the *skip* action. Agents have only a *local view* of their environment, their *perceptions can be incomplete*, and their *actions may fail*. That means that agents can receive incomplete information about the environment from the simulation server. The simulation server can omit information about particular environment cells, however, the server never provides incorrect information. Also, agent's action can fail. In such a case the simulation server evaluates the agent's action in the simulation step as the skip action.

After a finite number of steps the simulation server stops the cycle and participating agents receive a notification about the end of a simulation. Then the server starts a new simulation possibly involving the same teams.

## 1.2 Preparation Stage and Communication Protocol

Several days before the start of the competition, the contest organisers will contact participants via e-mail with details of time and Internet coordinates (IP addresses/ports) of the simulation server. Participants will also receive agent IDs and passwords necessary for authentication of their agents for the tournament. Agents communicate with the simulation server using TCP protocol and by means of messages in XML format. The details about communication protocol and message format will be specified later.

## 1.3 Initial Phase

At the announced start time of the tournament, the simulation server will go on-line, so that agents from participating teams will be able to connect. After a successful initial handshake during which agents will identify themselves by their IDs and receiving acknowledgment from the server, they should wait for the simulation start. The initial connecting phase will take a reasonable amount of time in order to allow agents to be initialised and connected and will not be less than 5 minutes. The details will be announced later.

## 1.4 Team, Match, and Simulation

An agent team consists of 6 software agents with distinct IDs. There are no restrictions on the implementation of agents, although we encourage the use of approaches based on the state-of-the-art tools, methodologies and languages for programming agents and multi-agent systems as well as the use of computational logic based approaches.

The tournament consists of a number of matches. A match is a sequel of simulations during which two teams of agents compete in several different settings of the environment.

For each match, the server will 1) pick two teams to play it and subsequently 2) start the first simulation of the match. Each simulation in a match starts by notifying the agents from the participating teams and sending them the details of the simulation. These will include for example the size of the grid, corral position, the number of steps the simulation will perform, etc.

A simulation consists of a number of simulation steps. Each step consists of 1) sending a sensory information to agents (one or more) and 2) waiting for their actions and 3) processing agents' replies and calculating the next state of the environment. In the case that an agent does not respond within a timeout (specified at the beginning of the simulation) by a valid action, it is considered to perform the *skip* action in the given simulation step.

### 1.4.1 Environment Objects

The (simulated) environment is a rectangular grid consisting of cells. The size of the grid is specified at the start of each simulation and is variable. However, it cannot be more than $150 \times 150$ cells. The $[0, 0]$ coordinate of the grid is in the

top-left corner (north-west). The simulated environment contains two corrals—one for each team—which serve as a location where cows should be directed to. The environment can contain the following objects in its cells:

- obstacle (a cell with an obstacle cannot be visited by an agent),
- cow,
- agent,
- corral (a cell to which cows are to be directed in order to earn points in a simulation).

There can be only one object in a cell, except that an agent or a cow can enter cells containing corral. At the beginning of a simulation the grid contains obstacles, cows and agents of both teams. Distribution of obstacles, cows and initial positions of agents can be either hand crafted for the particular scenario, or completely random. At the start of each simulation agents will get the details of the environment (grid size, corral position, etc.). Agents will get information about their initial position in the perception information of the first simulation step.

## 1.5 Perception

Agents are located in the grid and the simulation server provides each agent with the following information:

- absolute position of the agent in the grid,
- the content of the cells surrounding the agent and the content of the cell in which the agent currently stands in. Each agent has a viewing range that describes a square of visible cells around the agent with the width which is an odd number and the agent in the center of that square,
- agents will perceive identifiers of cows in every perception,
- agents will perceive the position of their own corral but not the position of the corral of the other team.

If two agents are standing in each other's field of view, they will be able to recognise whether they are enemies, or whether they belong to the same team. The corral position will be made available at the beginning of each match. Note that all perceptions except for the agent's and the corral's position can be omitted by the server, whereas the server never gives wrong informations.

## 1.6 Actions

Agents are allowed to perform one action in a simulation step. The following actions are allowed:

- *skip* – the agent does nothing,
- *north* – the agent moves to the north,
- *northeast* – the agent moves to the northeast,
- *east* – the agent moves to the east,
- *southeast* – the agent moves to the southeast,
- *south* – the agent moves to the south,
- *southwest* – the agent moves to the southwest,
- *west* – the agent moves to the west,
- *northwest* – the agent moves to the northwest.

Note that the $[0, 0]$ coordinate of the grid is in the top-left corner (north-west).

All actions, except the *skip* action, can fail. The result of a failed action is the same as the result of the *skip* action. An action can fail either because the conditions for its successful execution are not fulfilled or because of the information distortion.

The agent can do nothing or move into one of the eight directions of the compass. The execution of the *skip* action has no influence on the local state of the environment around the agent (under the assumption that other agents did not change it). When an agent does not respond to a perception information provided by the simulation server within the given time limit, the agent is considered as performing the *skip* action. The execution of the move actions changes the position of the agent one cell to the respective direction. A movement action succeeds only when the cell to which an agent is about to move does not contain an obstacle and is not outside of the grid. In the case two agents stand in the adjacent cells and one of them tries to step into the cell the second agent stands in while the second agent performs e.g. a *skip* action, the movement action fails. In the case two agents attempt to enter the same cell, only one of the two movement actions succeeds. Exactly which agent succeeds in entering the cell is determined randomly.

To make matters clearer:

- it is impossible for two agents/cows to swap places,
- a cow/agent can only enter a cell in the next step of the simulation if the cell is empty in the current step,
- agents can enter their own corral cells and the ones of the other team,
- if two or more agent/cows intend to enter the same empty cell it is left to chance which agent/cow succeeds.

## 1.7 Cow Movement Algorithm

Cows are simple creatures. They tend to move away from cells that they do not like and to move towards cells they do like. Cows want to move away from agents and trees. On the other hand, they are attracted by empty spaces and they want to stay close to other cows, however not too close. Cows have the tendency to form herds, which tend to be tighter in times when the animals are scared by cowboys.

The cows have two fixed visibility ranges. The number $r_c$ represents the width of the visibility-square with the cow in its center. The number $r_{cN}$ represents the width of the intimacy-square with the cow in the centre. Cows are attracted to other cows that are in the visibility-square and not in the intimacy-square and they are repelled by cows that are in their intimacy-square.

### 1.7.1 Cow Algorithm

Cows are slower than agents. Each cow only moves every three steps. Our simulation ensures that all cows do not move in the same step using this simple algorithm: At the beginning of the simulation each cow is given a random number $n_{cowID} \in \mathbb{N}$. Let $s \in \mathbb{N}$ be the current step of the simulation. The cow moves if the equation $s \bmod 3 \equiv n_{cowID} \bmod 3$ holds.

If it is time for a cow to move the following happens: For each cow *me* the set *Cells* is the contents of all the cells in the visibility range (which is a square with the agent in the centre) of the cow *me*. The content of a single cell can be either *cow*, *agent*, *tree* or *empty*.

Let $w : C \to \mathbb{Z}$ be a weight-function, that maps each cell to an integer according to its contents. A negative number indicates fear of the content and a positive one indicates attraction. The vectors $\vec{p}_c$ and $\vec{p}_{me}$ are the coordinate-vectors of the cell $c$ and the cow *me* respectively.

Now the position of a cow in the next simulation step is determined as follows:

1. Calculate the weighted linear combination

$$\vec{v} := \sum_{c \in Cells} w(c) \cdot \frac{(\vec{p}_c - \vec{p}_{me})}{|\ \vec{p}_c - \vec{p}_{me}\ |}$$

2. Move *me* according to the angle of $\vec{v}$.

Cows do not move if the resulting vector is zero. We use a number precision of two digits after decimal point without rounding.

Figure 1 shows how the directions of the compass correspond to angles: if the angle is in the range $[-22.5, 22.5)$ a cow moves east, if the angle is in the range $[22.5, 67.5)$ the cow moves to the northeast et cetera.

There are several constraints for the weights:

- $w(empty) > 0$, cows are attracted by empty spaces,

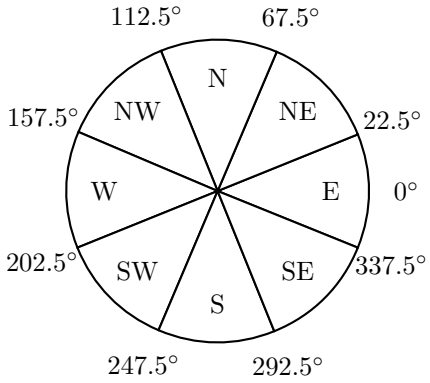


Figure 1: Angles and directions.

- $w(cow) > 0$, cows are attracted by other cows if these are not too close,
- $w(cow_{private}) < 0$, cows are attracted by other cows if these are too close,
- $w(agent) < 0$, cows are scared of agents,
- $w(tree) < 0$, cows are scared of trees,
- $w(tree) = -w(empty)$, cows are scared of trees as much as they are attracted by empty spaces,
- $\mid w(cow) \mid < \mid w(agent) \mid$, cows are less attracted by other cows than they are scared of agents, and
- $w(corral) = w(empty)$, cows do not distinguish between corral-cells and empty spaces.

The weights will be announced soon on the website and on our mailing list.

### 1.7.2 Example

Fig. 2 shows the contents of the cells that are in the visibility range around the cow *me*. In this example cows have the visibility range 3. Note that we go for bigger values in our simulation and that we just use 3 in this example for the sake of simplicity. The cell in the northwest of *me* contains a *tree*, the one in the northeast contains an *agent* and all the other cells are *empty*.

Let the weights be as follows (they will be slightly different in our simulation):

$$w(empty) = 1$$
$$w(tree) = -1$$
$$w(agent) = -2$$

The weighted sum is as follows:

$$\begin{aligned}\vec{v} &:= \left(-\frac{1}{\sqrt{2}}\right)\cdot\begin{bmatrix}-1\\-1\end{bmatrix} + \left(-\frac{2}{\sqrt{2}}\right)\cdot\begin{bmatrix}1\\-1\end{bmatrix} + \\ &\quad 1\cdot\left(\begin{bmatrix}0\\-1\end{bmatrix} + \begin{bmatrix}-1\\0\end{bmatrix} + \begin{bmatrix}1\\0\end{bmatrix} + \frac{1}{\sqrt{2}}\cdot\begin{bmatrix}-1\\1\end{bmatrix} + \begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{\sqrt{2}}\cdot\begin{bmatrix}1\\1\end{bmatrix}\right) \\ &= \left(\frac{1}{\sqrt{2}}\right)\begin{bmatrix}-1\\5\end{bmatrix}\end{aligned}$$

Determine the direction and move the cow: The angle of $\vec{v}$ is 258°, thus the cow moves south.

| | | |
|---|---|---|
| *tree* | *empty* | *agent* |
| *empty* | *me* | *empty* |
| *empty* | *empty* | *empty* |

Figure 2: Example for the cow-algorithm.

### 1.8 Final Phase

In the final phase, the simulation server sends a message to each agent allowing them to disconnect from the server. By this, the tournament is over.

## 2 Submission and Winning criteria

A submission consists of a 5 page description (analysis and design) of your solution and the participation of your agent team in the tournament. The winner of the contest will be the best performing team with the highest number of points from the tournament.

## 2.1 Relation to the Previous Contest Editions

The first two editions were organized in cooperation with the CLIMA workshop series, the third one was the first one in cooperation with the ProMAS workshop series. This scenario is not an extension of the last contests, which were hunts for gold. Instead it is completely new scenario.

The main differences to the gold-mining scenario are:

- the agents have a wider viewing range,
- the agents have only moving actions, now they can move into eight directions instead of four, the pushing action has been abandoned,
- cows were introduced that flock and disperse, and
- the goal is to direct as many cows as possible into one of the corrals.

We believe that the new scenario constitutes a new and very interesting challenge.

## 2.2 Network Miscellanea

Our simulation server does not provide a facility for inter-agent communication. Agents from a team are allowed to communicate and coordinate their actions locally. Based on the number of participants, organisers will decide whether to run the competition in just one or more rounds.

The continuous connection of agents from the first match to the last one cannot be guaranteed. In the case of agent-to-server connection disruption, agents are allowed to reconnect by connecting and performing the initial tournament phase message exchange again.

Generally, participants are responsible for maintaining connections of their agents to the simulation server. In the case of connection disruption during the running simulation, server will proceed with the tournament simulation, however the action of a disconnected agent will be considered as the *skip* action. In the case of a serious connection disruption, organizers reserve the right to consider each case separately.

The agents should inform the simulation server which action they want to perform within a timeout specified at the beginning of the simulation. The contest organisers do not take any responsibility for the speed of the Internet connection between the server and participating agents. Timeouts will be set reasonable high, so that even participants with a slow network connection will be able to communicate with the server in an efficient way. Simulation timeouts will not be lower than 2 and higher than 10 seconds per one simulation step.

A ping interface will be provided by the server in order to allow participating agents to test the speed of their connection during the initial phase of the tournament. Note, that only a limited number of ping requests will be processed from one agent in a certain time interval. Details on this limit will be announced later through our mailing list and posted on our website.

## 2.3 Technical Support and Organisational Issues

We are running a mailing list for all the inquiries regarding Multi-Agent Programming Contest 2008. Feel free to subscribe if you are interested in further details on the contest. Subscription for participants is mandatory. The list address: `agentcontest2008-general@in.tu-clausthal.de` To subscribe, please send an e-mail to `agentcontest2008-general-subscribe@in.tu-clausthal.de` The most recent information about the Multi-Agent Programming Contest 2008 can be found on the official web site `http://cig.in.tu-clausthal.de/agentcontest2008/`