

**Gerstner Laboratory  
for Intelligent Decision Making and Control**



**Adversarial Behavior Testbed**

**Eduard Semsch, Viliam Lisý, Branislav Bošanský,  
Michal Jakob, Dušan Pavlíček, Jan Doubek,  
Michal Pěchouček**

{semsch, lisy, bosansky}@labe.felk.cvut.cz

<b>Report No.</b>	<i>GLR 90/09</i>	<b>Date</b>	<i>06.02.2009</i>
-------------------	------------------	-------------	-------------------

*Gerstner Laboratory  
Czech Technical University in Prague  
Faculty of Electrical Engineering  
Technická 2, 166 27 Prague 6  
Czech Republic*

<http://cyber.felk.cvut.cz/gerstner>

# 1 Introduction

Whenever a number of autonomous entities operate in a shared environment, situations of conflict or cooperation can arise if the interests of involved parties are not perfectly aligned. Formalized as *games*, such situations have been long studied in game theory and a wider range of related disciplines (economics, sociology, artificial intelligence).

Independently, *agent-based simulation* has become a popular approach to studying behavior and properties of complex systems. So far, however, a vast majority of such simulations have used relatively simple reactive techniques for controlling the behavior of entities inhabiting the simulated scenarios. Such techniques are often insufficient to realize the more complex reasoning required by decision-making in conflict situations, in particular the explicit considerations of other agents and their actions.

In order to reliably simulate realistic conflict situations the explicit consideration of game aspects of the situations is necessary and there are several techniques how to introduce reasoning about adversaries (players) into these complex systems [1], [2]. Moreover, in order to experimentally evaluate these techniques an appropriate testbed is needed. Hence, in this report we describe our adversarial reasoning testbed together with the design of the one specific scenario, called Tsunami Recovery Game, that we implemented and used for the experiments.

## 2 Adversarial Reasoning Testbed

This section describes the adversarial reasoning testbed implemented for experimenting with adversarial reasoning and planning in complex adversarial domains. We start with identifying the key properties of real-world conflict situations, giving rise to the definition of the class of *complex asymmetric games* which provides the target domain for the techniques we want to experimentally evaluate. Following the general definition, a particular instance of complex asymmetric games, the Tsunami recovery game modelled after a disaster recovery operation in a politically unstable environment is described.

### 2.1 Complex Asymmetric Games

The testbed should support exploring adversarial reasoning and planning in complex, asymmetric domains with properties similar to those found in real-world adversarial situations (also termed *complex asymmetric games* throughout the report). Such situations are characterized by the following properties:

- *multiple players* – the number of players is higher than two and can change during the game
- *asymmetric utility* – players have different goals which are not necessarily antagonistic; players and their groups can be mutually cooperative, competitive and/or neutral to varying degrees
- *partial knowledge* – players do not have access to the full state of the game; furthermore, different players can have differing views on the game state
- *non-deterministic actions* – effects of actions are not fully predictable either due to inherent stochasticity of the actions or due to limited information a player has
- *concurrent* – players carry out actions in parallel

	Chess	Bridge	Robocup soccer	RTS	War-gaming
multiple player				•	
partial knowledge		•		•	•
non-deterministic		•	•	•	•
continuous			•	•	•
concurrent			•	•	•
asymmetric utilities					
asymmetric resources					

Table 1: Properties of selected well-known games

We searched for an existing experimental domain with the characteristic of complex asymmetric games. We evaluated some of the classical games as well as real-time strategies and some war-gaming simulators such as OneSAF. In Table 2.1, we briefly analyze them with respect to the properties introduced above. As can be seen, most of the games that have been subject of research in the past lack most of the properties of complex asymmetric games.

## 2.2 Supported Class of Games

The developed testbed supports large set of games that have following properties:

- Entities in the game have a *position*. Some of the entities may change their position using actions.
- Entities have *characteristics* that are not changeable. The most notable characteristic is the unique name.
- Some entities have *inner state* that is changeable as a result of actions or inner dynamics.
- Some entities may perform *actions*.
- Actions are evaluated in *discrete time steps* and in parallel.
- Some entities have *sensors* defined that provide the entity with the information about the world state.
- Some entities may *communicate* with each other.

We differentiate six types of entities that form the basis of all implementable games:

- *Locations*, which are distinct places in space.
- *Roads* connecting locations creating a planar graph (*map*) of locations and roads which define possible positions of all other entities.
- *Cities* that are located in *locations* and that have an inner state.
- *Carriers* that may be located anywhere on the *map* and that have a inner state.
- *Actors* that may be located anywhere on the *map* and that may perform actions in the world. Actors do not have an inner state in the domain. They may be affiliated with a player.

- *Players* that have no location on the *map* and that have no inner state represented in the domain. Players may give commands to actors. Actors may or may not belong to a player.

Note that the inner state is the state of an entity in the simulated world (physical state) and it must be distinguished from the inner state of the intelligence that is controlling the entity. Furthermore, all the entities have unique identifiers in the simulated world.

### 2.2.1 World State

We understand the world state to be the union of inner states of all entities that exists there. The states of the simulated world may change in two ways:

- *Environmental dynamics* – the state of the world can change as functions of time and previous states.
- *Actions* – the state of the world is changed by actions carried out by actors in the world.

Time in the simulated world is discrete divided to arbitrarily short time steps (simulation ticks). Actions of individual actors have integer durations and are executed in parallel. For the action execution there are certain rules that define the order in which are actions executed in a single time step, creating a well defined joint action of all the actors. The environment dynamics are applied after actions.

### 2.2.2 Actors

The actors in the simulated world perform actions to change the world state. They may have an arbitrary level of autonomy. They may or may not belong to a player. The fact that an actor belongs to a player is an unchangeable characteristic of the actor and it can be used in the process of evaluation of the actions. The actors may or may not communicate with each other.

### 2.2.3 Players

A player in the game is an entity with a goal defined in terms of desirable state(s) of the world. A player does not modify the state of the world directly but through coordinated actions of actors it has on its disposal. The players in the simulated world have goals that may or may not be asymmetric. This means that they try to optimize different criteria or to reach different world states. The players may or may not communicate with each other.

### 2.2.4 Sensors

The players and the actors have views of the world state defined by their sensors. The sensors are filters of the world state. There are presently three implemented types of filters. The first one is distance-based where the filtering function is the euclidean distance between the sensing entity and its environment. This means that the actor/player 'sees' other entities in the circle of predefined range around itself. The second filter is on the types of entities the actor/player 'sees'. This means that the sensing entity may

'see' only entities of predefined type. The third filter is on the states of the entities. This means that the sensing entity may 'see' only the predefined states of the other entity. The distance-based sensor may be combined with the other two sensors to provide a distance-based sensor sensing only predefined types of entities or predefined states of entities.

### 3 Adversarial Tsunami Recovery Game

After introducing the general class of games currently supported by the project, we now describe a specific game used as a testbed for the techniques developed. The implemented game is modeled after a disaster recovery operation involving three players<sup>1</sup>: government, non-governmental humanitarian organization and separatists. An overview scheme of the game is in Figure 1. The game takes place on an island broadly divided into two parts. The northern part has been hit by a natural disaster while the southern part has been left unaffected.

As the result of the disaster, the cities in the northern part lack *infrastructure* (indicated by the light grey number in the city status) and they do not have *government head quarters* (a blue *GovHQ* indicator). This together means that the government has no control over the cities. The cities are already provided with food (the green number in the city status). The cities controlled by government have a full-width blue bar shown next to them, the cities that are not controlled by government and have enough food have a half-width green bar, and a city without food has a small red bar.

Activities that need to be performed by the government are depicted in blue. The government has to first transport *explosives* from *explosives factory* (an orange *Fact* indicator) to a *quarry* (a yellow *Quarry* indicator) where they are used to produce *stones* that are later used in damaged cities by *engineer* units to repair the infrastructure and build the headquarters. The transportation of stones and explosives is performed by *trucks*.

However, there are separatists (red) trying to prevent the government from achieving control in the cities. The separatists steal the explosives from the government trucks and store them in their *separatist camps* (a red *GngHQ* indicator). If they have enough explosives they carry out a *suicide bomb attack* that destroys the government headquarters in a chosen city.

Finally, the third player in the game is a non-governmental organization (green) whose goal is to transport food from *farms* (a green *Farm* indicator) to cities that need it. The food in the cities is consumed over time.

Below we give a more formal description of the game.

#### 3.1 World States

In the implemented scenario the cities  $C_i$  have these (unchangeable) characteristics<sup>2</sup>:

- Explosives capacity.  $ExplosivesCapacity(C_i, P_j) \geq 0$  where  $P_j$  denotes the player to whom the explosives belong.
- Stones capacity.  $StonesCapacity(C_i) \geq 0$ .

<sup>1</sup>In the basic set up – it is possible to have multiple instance of each player type

<sup>2</sup>We use the term *characteristic* to denote a constant state



Figure 1: A scheme of the implemented Adversarial Tsunami Recovery Game – a disaster recovery operation performed by the government with a non-governmental organization maintaining food supply and separatists trying to prevent the government from regaining control.

Entity	Characteristics	States	Derived States
City	ID, food capacity, explosives capacity, stones capacity, maximum infrastructure level, farm presence, quarry presence, explosives factory presence	amount of food, amount of explosives, amount of stones, infrastructure level, HQ presence, SCamp presence	wellbeing, government control, number of police units, number of separatists
Carrier	ID, food capacity, explosives capacity, stones capacity, type of the car (fighter or engineer or suicide bombing), player affiliation, speed	food load, explosives load, stones load	$\emptyset$
Actor	ID, player affiliation	$\emptyset$	$\emptyset$
Player	ID	$\emptyset$	$\emptyset$

Table 2: States and characteristics of entities in the Tsunami recovery game

- Food capacity.  $FoodCapacity(C_i) \geq 0$ .
- Max infrastructure level.  $MaxInfrastructureLevel(C_i) \geq 0$ .
- Presence of farm.  $Farm(C_i)$ .
- Presence of explosives factory.  $ExplosivesFactory(C_i)$ .
- Presence of quarry.  $Quarry(C_i)$ .

The cities have these (changeable) inner states:

- Amount of explosives  $0 \leq Explosives(C_i, P_j) \leq ExplosivesCapacity(C_i, P_j)$ .
- Amount of stones  $0 \leq Stones(C_i) \leq StonesCapacity(C_i)$
- Amount of food  $0 \leq Food(C_i) \leq FoodCapacity(C_i)$
- Infrastructure level  $0 \leq Infrastructure(C_i) \leq MaxInfrastructureLevel(C_i)$ .
- Presence of the government headquarters.  $HQ(C_i)$ .
- Presence of the separatist camp.  $SCamp(C_i)$ .

The cities also have derived states (i.e. states whose value derives from other states):

- Wellbeing in city.  $Wellbeing(C_i)$ . Defined as  $Wellbeing(C_i)$  iff  $Food(C_i) \geq WellbeingConstant$  where  $WellbeingConstant > 0$ .
- Number of police in city.  $Police(C_i)$ .
- Number of separatists in city.  $Separatists(C_i)$ .
- Government control in city.  $Control(C_i)$ .  
Defined as  $Control(C_i)$  iff  $(Wellbeing(C_i) \wedge HQ(C_i)) \vee$   
 $(Wellbeing(C_i) \wedge (Police(C_i) - Separatists(C_i) \geq PolicePresentConstant1)) \vee$   
 $(Police(C_i) - Separatists(C_i) \geq PolicePresentConstant2)$   
where  $PolicePresentConstant2 > PolicePresentConstant1$ .

The carriers  $Car_i$  have these characteristics:

- Affiliation to a player.  $BelongsTo(Car_i, P_j)$  where  $P_j$  denotes the player to whom the carrier belongs.
- Explosives capacity.  $ExplosivesCapacity(Car_i)$ .
- Stones capacity.  $StonesCapacity(Car_i)$ .
- Food capacity.  $FoodCapacity(Car_i)$ .
- Is the car a fighter car.  $Fighter(Car_i)$ .
- Is the car an engineer car.  $Engineer(Car_i)$ .

- Is the car a suicider car.  $Suicider(Car_i)$ .
- Speed of car.  $Speed(Car_i) > 0$ .

The carriers have these inner states:

- Number of explosives.  $0 \leq Explosives(Car_i) \leq ExplosivesCapacity(Car_i)$ .
- Number of stones.  $0 \leq Stones(Car_i) \leq StonesCapacity(Car_i)$ .
- Number of food.  $0 \leq Food(Car_i) \leq FoodCapacity(Car_i)$ .

The actors  $A_i$  have only one characteristic and no inner state:

- Affiliation to a player.  $BelongsTo(A_i, P_j)$  where  $P_j$  denotes the player to whom the actor belongs.

Then there are these predicates defined that further specify the world state:

- Entity location.  $At(E_i, N_j)$  where  $E_i$  is either a city or a carrier or an actor and  $N_j$  is an identifier of a location.
- Linking actors with carriers.  $Linked(A_i, Car_j)$  where  $A_i$  is an actor and  $Car_j$  is the carrier the actor is linked with.
- Connectedness of locations.  $Connected(N_i, N_j)$  where  $N_i$  and  $N_j$  are locations.
- Time it takes a carrier to move between locations.  $TraverseTime(N_i, N_j, Car_k)$  where  $N_i$  and  $N_j$  are locations and  $Car_k$  is a carrier.

## 3.2 Environment Dynamics

There are the following rules describing environment dynamics in the game:

- Food creation. If the city has a farm it produces a  $FoodProductionAmount$  units of food in  $FoodCreationPeriod$  time steps.
- Food consumption. All cities consume  $FoodConsumptionAmount$  units of food in  $FoodConsumptionPeriod$  time steps.
- Explosives generation. If the city has an explosives factory the city produces  $ExplosivesProductionAmount$  units of explosives in  $ExplosivesProductionPeriod$  time steps.
- Explosives to stones transformation. If the city has a quarry and there are explosives it produces up to  $StonesProductionAmount$  units of stones in  $StonesProcutionPeriod$  time steps while consuming the same amount of explosives.

### 3.3 Players

There are three players in the scenario:

- Government (*GOV*)
- Non-governmental organization (*NGO*)
- Separatist (*SEP*)

Thus the set of players  $P$  is defined as  $P = \{GOV, NGO, SEP\}$ .

The players have these utilities:

- Government utility.  $U(GOV)$  that is defined as  $\|(C|Control(C))\|$  – the number of cities the government controls.
- Non-governmental organization utility.  $U(NGO)$  that is defined as  $\|(C|Wellbeing(C))\|$  – the number of cities that have enough food.
- Separatist utility.  $U(SEP)$  that is defined as  $\|(C|Wellbeing(C) \wedge \neg Control(C))\|$  – the number of cities that have enough food, but which the government does not control.

### 3.4 Actions

The actors are capable of performing the following actions:

- Move. The outcome of the action is that the actor (possibly with a carrier) is moved to another location connected with the current location by an arch. The action can take several simulation steps correspondingly to the length of the arch.
- Load food. Applicable if the actor has a truck that has *food* capacity and it is located in a city that has food. The outcome is that a unit of food is moved from the city to the truck.
- Unload food. Applicable if the actor has a truck that has *food* and the city has food capacity. The outcome is that a unit of food is moved from the truck to the city.
- Load explosives. Same as load food except the commodity are *explosives*.
- Unload explosives. Same as unload food except that the commodity is *explosives*.
- Load stones. Same as load food except the commodity are *stones*.
- Unload stones. Same as unload food except the commodity are *stones*.
- Build HQ. Applicable if the actor is an engineer and the city has enough food and there is a unit of stones. The outcome is that the city has government HQ.
- Create separatist camp. Applicable if the number of separatists in the city is greater than a certain number and there is not government HQ in the city. The outcome is that the city has a separatist camp which also means that there is explosives capacity for the separatists.

- Create suicide bomber. Applicable if the number of separatists' explosives in the city is greater than a certain number and there is not government HQ in the city. The outcome is that a new separatist actor and a suicide bombing car is created.
- Repair infrastructure. Applicable if there is a unit of stones in the city and an engineer and the infrastructure in the city is not yet repaired. The outcome is that the infrastructure level of the city is increased by one.
- Steal explosives. Applicable if there is a separatist that has a truck with explosives capacity, a truck with explosives, and the number of separatists in the location is greater than the number of policemen. The outcome is that the explosives are moved from the truck to the separatist.
- Destroy stones. Applicable if there is a separatist and a truck with stones and the number of separatists in the location is greater than the number of policemen. The outcome is that the stones from the truck are removed.
- Destroy food. Same as destroy stones except that the commodity is *stones*.
- Destroy separatist camp. Applicable if the city has a separatist camp and there are more policemen in the city than the separatists plus a certain number. The outcome is that the separatist camp is removed and also all separatists' explosives.
- Destroy government HQ. Applicable if the city has government HQ and there is a suicide bomber. The outcome is that the government HQ is removed.

Note that the actors can perform different actions depending on what car they are linked with. Lastly, the sensors are specified for full visibility of all states for all actors and players.

## 4 Summary

In this report we described in detail the adversarial reasoning testbed and the specific game scenario that we implemented using the testbed and used for the experimental evaluation of novel techniques in game playing in realistic scenarios. Moreover, every game that satisfies the properties specified in 2.2 can be implemented on the top of the testbed and therefore used as a basis for further game-playing algorithm research.

## 5 Acknowledgement

This research was partially funded by AFOSR USAF grant number FA8655-07-1-3083 and by the Ministry of Education of the Czech Republic grants ME09053 and MSM6840770038.

## References

- [1] Eduard Semsch, Michal Jakob, Jan Doubek, and Michal Pechoucek. Using player models to improve robustness of htn plans in multi-agent domains. In *Proceedings of the 27th Workshop of the UK Planning and Scheduling Special Interest Group*, 2008.

- [2] Viliam Lisý, Branislav Bošanský, Michal Jakob, and Michal Pěchouček. Goal-based adversarial search - searching game trees in complex domains using goal-based heuristic. In Joaquim Filipe, Ana Fred, and Bernadette Sharp, editors, *Proceedings of ICAART 2009 - First International Conference on Agents and Artificial Intelligence*, pages 53–60, Porto – Portugal, 2009. INSTICC (Institute for Systems and Technologies of Information, Control and Communication), INSTICC Press.