# Incorrect Moves and Testable States

Dimiter Dobrev

Institute of Mathematics and Informatics
Bulgarian Academy of Sciences

The second goal of the agent in the Reinforcement learning proses is to collect maximal rewards.

The second goal of the agent in the Reinforcement learning proses is to collect maximal rewards.

The first goal is:

To understand the world!

Arbitrary world: $\boldsymbol{<S, s_0, World, View>}$

S - the set of internal states
$s_0$ - the initial state
World - the function which gives the next state

$\boldsymbol{s_{i+1}=World(s_i , a_{i+1})}$

View - the function which say what we see

$\boldsymbol{v_i=View(s_i)}$

The result is the sequence: action, view, action, view ...

$\boldsymbol{a_1, v_1, a_2, v_2, \ldots}$

View = <Info, Rewards>

We consider two cases:

full observability $\boldsymbol{View(s_i) = s_i}$
partial observability $\boldsymbol{View(s_i) \neq s_i}$

To understand the world means to reduce the problem of partial observability to the problem of full observability.

The case of full observability. This case is trivial. It can be solved by filling in the following table:

| | $a_1$ | $a_2$ | $a_3$ | … | $a_i$ | … | $a_n$ |
|---|---|---|---|---|---|---|---|
| $s_1$ | | | | | | | |
| $s_2$ | | | | | | | |
| $s_3$ | | | | | | | |
| … | | | | | | | |
| $s_i$ | | | | | $s_x$ | | |
| … | | | | | | | |
| $s_{last}$ | | | | | | | |

The result of this algorithm is a program with a delayed growth. It needs a huge (almost infinitely long) time for education.

$$View(s_i) = v_i = \langle v^1_i, \ldots , v^m_i \rangle$$

We will add more coordinates to vector $v_i$, so the new vector will give a full description of the state of the world $s_i$.

This means that if two states are distinguishable, then two different vectors will correspond to them.

$$\langle v^1, \ldots , v^m, v^{m+1} , v^{m+2}, \ldots \rangle$$

What will be this new coordinates? First, we will add coordinates which say which action is correct and which one is not.

# What is incorrect action?

In the chess game, we cannot move the bishop as a knight, so it is natural to assume that some of our moves are incorrect or impossible.

We will suppose that the function ***World*** is not total and that the ***domain(World)*** describes the set of all correct actions. In different moments (at different states) different actions are correct.

If we know which move is correct, it will give us additional information about the state of the world. If we know we can move along the diagonal, we will know that we hold in our hand the bishop or the queen.

We assume that we can get the information about which move is correct for free (without any effort and without any punishment).

For example, if we pass alongside a door then we can check is it locked (can we open it). This is without effort because we are there anyway. If we lose time to do this test then ‘to push the handle’ is a correct action and ‘the door is locked’ is a testable state.

If at a specific moment we know what we see and what moves are correct, we know a lot, yet we do not know everything. We will generalize the concept of ‘incorrect move’ to the concept of ‘testable state’. If we add to the input vector the values of all testable states, we will get an infinite-dimensional vector that fully describes the state of the world.

# What is testable state?

A testable state is the result of an experiment. Here are two examples:

**The door is locked**
**If I push the handle ⇒ the door will open.**

**The stove is hot**
**If I touch the stove ⇒ I will get burned.**

Here we have the testable state and the experiment, which must be made in order to obtain the value of the testable state. The result of the experiment can be ‘Yes’ or ‘No’.

From the above two examples you might get the impression that the experiment consists of any actions that we need to do, but it is not so. The experiment could be something that happens without our intervention. Here is an example:

**The roof is damaged**
**If it rains ⇒ the roof will leak.**

**Definition**: A simple testable state is a statement of the type: The j^th^ coordinate of the input has some value $v^j_i = Constant$

or

The k^th^ move is correct $<s_i, a_k> \in domain(World)$

The simple testable state is a Boolean function which returns true or false for every state of the world.

**Definition**: A testable state is a simple testable state with some precondition or postcondition or both.

The testable state is function that depends on the history around the current moment. It is defined only under certain circumstances (when the precondition and the postcondition are true).

# To describe the testable state we need a theory.

This theory will give us:

1. Continuation of the function. We believe the stove is hot or cold, whether we touched it or not.

2. Prediction of the future.
**If I turn on the stove ⇒ It will be hot.**

3. Prediction of the past.
**If the lunch is ready ⇒ The stove was hot in the morning.**

# Continuation of the function

We want to find other circumstances when the testable state is true or false.

**If the stove is red ⇒ It is hot.**
**If the stove is with ice ⇒ It is cold.**

We find these rules through the statistic. Many times, one happened and never happened two:

1. When the stove was red, we touched it and burned.
2. The stove was red, we touched it and did not burn.

**The stove is red with ice ⇒ contradiction.**

# Probability Rules

If one rule is wrong once or twice, we will not throw it out. We will use this rule as a probability rule.

We will not assume that this rule is true with some probability because this is too strong assumption. The maximum we can assume is that the rule is true with probability in the range $[a, b]$.

# Testable States are Inert

To predict the past and the future, we will use the rules obtained through the statistics. Also we will use the assumption that testable states are inert and change only if specific events occur. That is, if between two checks, none of these events has occurred, we can assume that the value of the testable state has not changed.

For example, if a door has been locked at a specific moment and shortly thereafter we check again, we assume that it will be locked again, especially if during this time no specific events have occurred (for example, to hear a click of door locks).

# A castle with many doors

More complicated theory we will obtain if we imagine that our world is a castle with many doors. Then if we say: “The door is locked” arise the question: “Which one?”

We will make a theory with many inert variables that match the different doors. Some of these variables will be constant because some doors are permanently locked or unlocked. The theory also will include a map of the castle (a finite-state machine) and a variable that shows where we are on the map.

# A Crowded World

Why do some doors change their status? If we assume that we are alone in our world then it will be very difficult to find an explanation of the behavior of the doors.

Let’s we assume that there is some kind of creature inside the castle, which unlocks and locks the doors in its sole discretion. Then we can predict whether a door is locked or unlocked predicting the behavior of that creature.

We will not make a dissection of this creature. We will consider him as a black box and we will assume only such things like that he is our friend or he is our opponent.