# Practical Examples Using Semantic Simulation With Reinforcement Learning

D. Gueorguiev 12/4/2022

# The Game Addition

Let us consider the game described in *Blackwell’s Theory of Games and Statistical Decisions* (Blackwell & Girshik, 1978, p. 14):

and alternatively choose integers, each choice being one of the integers and each choice made with the knowledge of all preceding choices. As soon as the sum of the chosen integers exceeds , the last player to choose pays his opponent one unit.

The situation at which player finds himself at his th move is described by a sequence with each being one of the integers and

Denote by the set of possible sequences where and denotes the closest integer which does not exceed . A strategy for consists of a set of functions , where is a function defined on assuming only values : specifies ’s th move when the previous history of the play is . Similarly, a strategy for is a set of functions , where is defined for the set of all sequences with each being one of the integers and

Define and inductively for ,

(this induction describes the manner in which a referee would carry out the instructions of the players) and let be the largest for which is defined. Then

# Constructing semantic universe for the game

Let us consider the following thought experiment – we have two players playing the game described earlier. Each player is represented by semantic simulation which has its own set of semantic structures and semantic template which recognizes the rules of the game. Let us start our experiment by looking in the semantic template which recognizes the rules of the game which we will name *semantic recognizer*. That is - we are interested in what the semantic recognizer might be taking as an input and producing as an output and how the semantic recognizer template would be interacting with the rest of the semantic structures running in the simulation.

Let us assume that the semantic simulation corresponding to each of the two players and is limited to the simply connected regions and in semantic space. Additionally, we introduce an Arbiter which will be assigned its own simply connected region in semantic space. Let . Let us assume that where is finite, closed and simply connected region of semantic space with the same number of dimensions . We will denote as the *common simulation region*.

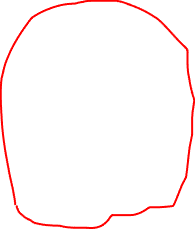
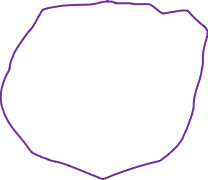
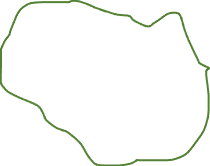


Figure 1: Layout of the simulation space in Blackwell’s game *Addition*

*Definition*: *neutral point* of a simply connected region in metric space

Let is a simply connected region in some dimensional metric space. Then the point is a neutral point *iff* it is the center of the largest dimensional sphere which can fit entirely in the simply connected region without including any points outside of . Formally,

With we denote *the neutral point* of the common simulation region . The neutral point will be the attraction center for all outputs from player and ’s as well as the arbiter simulations. Both players and as well as the Arbiter will produce an output which will be a semantic particle starting its existence at a point inside their respective regions shown on Figure 1.

*A couple notational conventions which will simplify the discussion:*

Let us have a template defined over the semantic region .

*Region over which a template is defined*

In the future we will denote the region over which the template is defined with the appropriate symbol denoting the region in parentheses; Thus indicates that is defined over .

*Trajectory of semantic particle*

A particle having trajectory intersecting with specific region will be denoted with the following notation .

*Template match*

We denote a template match, that is the template has matched the input represented by with the following symbolic notation .

*Chaining of template actions*

The notation ( are predicates) indicates that triggering causes particle p to be emitted which if matched will trigger the predicate .

Let us have the following templates and and is semantic particle.

Then the notation

indicates that the particle being present in triggers which in turn emits a new particle which if present in will trigger a *chained template* .

*Here is how the game simulation will proceed:*

For simplicity let us assume that the game parameter defined earlier is given and it is known by the two players and the Arbiter. Also, we will assume that the Arbiter will make decision who will be the first of the two players to play; for simplicity the decision-making process of the Arbiter will be omitted from the discussion. Let us represent this decision-making process of the Arbiter by the semantic template ( for start of the game). The template accepts an input indicating the start of the game.

The input indicating the start of the game will be represented as a particle with specific signature which we will denote with . As soon as the arbiter template detects that the signature of is present in it sends either a particle to region or to region .

In case of a template which belongs to Player will recognize the signature of that is will be triggered: . On a match will send a messenger particle to another template of Player - . In turn the inference structure of sends an information particle toward in . The information particle is a composite semantic particle and contains two sub-particles:

* sub-particle conveying the information that it has been created by a template which belongs to Player
* sub-particle conveying the information that Player has chosen the number on his current move

As is sent toward a template which belong to the Arbiter is looking for specific patterns.

is the so called *end-of-the-game recognizer*. This template will create different response depending on the pattern it detects. One of the patterns recognizes is single particle in C. We can write this sequence as:

Start (1a)

We can simplify the notation above, writing short-hand:

Start (1b)

In the case when is sent to a template which belongs to Player will recognize the signature of . In this case we write

Start (2a)

Similar to (1b) we write short-hand:

Start (2b)

In case of (1) the Inference Structure of will create particle sent toward . Alternatively, in case of (2) the Inference Structure of will create particle sent toward .

So, the sequence (1b) is extended as:

Start

And the ball is one more time in the field of the *end-of-the-game recognizer* .

Seeing moving toward the neutral point of , will either send towards or issue a signal for the end of the game.

Thus, we will end with one of the four sequences:

Start End of Game

or

Start End of Game

or

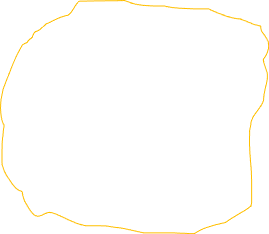
Start End of Game

or

Start End of Game

In either case the end of the game is recognized with a pattern, similar to the one shown on Figure 2 below.

Figure 2: Possible final arrangement of the semantic particles produced by the two players at the end of a game of *Addition*



It is important to realize that the pattern on Figure 2 can be *transformed* by some *known transformation* to the graph shown on Figure 3 below. This would be true with well chosen laws governing the motion of semantic particles. For details on the equations governing the positions and dynamics of semantic particles refer to documents [Modeling Attractive and Repulsive Forces in Semantic Properties](https://github.com/dimitarpg13/aiconcepts/blob/master/docs/ModelingAttractiveRepulsiveForcesInSemanticProperties.pdf) (section *Constructing the Property Tree: constraints and inequalities based on Binding Force*) and [On The Need Of Dynamic Simulation When Modeling Interactions of Semantic Particles](https://github.com/dimitarpg13/aiconcepts/blob/master/docs/OnTheNeedofDynamicSimulationWhenModelingInteractionsOfSemanticStructures.pdf) (section *Dynamic Modeling of Semantic Structure Aggregates*).

.

Figure 3: Semantic structure formed by the final arrangement of the output of the two players

Obviously in order the pattern shown on Figure 3

Here is one way to define the Pattern Matching structure of the *end-of-the-game recognizer*

Encode the number in the mass of the sub-particle .

Let us denote with the set of all particles which contain a sub-particles or and are in the region . Let us denote with the total mass of all sub-particles with signatures which belong to a particle in . The *end-of-the-game recognizer* is triggered by any incoming to the region particle or . Compare with the number (the parameter of the Blackwell’s game *Addition* defined in the beginning). In case and the incoming particle is then create and send a particle toward as the Game continues. In case and the incoming particle is then create and send a particle toward as the Game continues. In case and the incoming particle is stop the game and announce Player as a winner. In case and the incoming particle is stop the game and announce Player as a winner.

To summarize:

The following templates represent the entity known as the *Arbiter*:

The following templates represent the entity known as the *Player* :

The following templates represent the entity known as the *Player* :

Let us discuss now an implementation of

# Reinforcement Learning in the Blackwell’s Game of Addition

//TODO: finish this

# Bibliography

Bang-Jensen, J., & Gutin, G. (2007). *Diagraphs: Theory, Algorithms and Applications.* Odense, Denmark, London, UK: Springer-Verlag.

Bellman, R. (1972). *Dynamic Programming.* Princeton, NJ: Princeton University Press, Sixth Printing.

Blackwell, D. A., & Girshik, M. A. (1978). *Theory of Games and Statistical Decisions.* New York: Dover Publications; Illustrated edition.

Denardo, E. V. (1982). *Dynamic Programming: Models and Applications.* Englewood Cliffs, NJ: Prentice-Hall.

Francois-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). An Introduction To Deep Reinforcement Learning. *Foundations and Trends in Machine Learning: Vol. 11, No. 3-4*.

Gosavi, A. (2022). Reinforcement Learning: Tutorial and Recent Advances. *INFORMS Journal on Computing*.

Harmon, M. E., & Harmon, S. S. (Jan 1997). *Reinforcement Learning: A Tutorial.* Wright Patterson AFB OH 45433: Avionics DIrectorate, Wright Laboratory, Air Force Materiel Command.

Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020, Nov 1). *Offline Reinforcement Learning: Tutorial, Review, and Perespectives on Open Problems*. Retrieved from arxiv.org: https://arxiv.org/abs/2005.01643

Neapolitan, R. E. (2019). *Bayesian Networks.* Upper Saddle River, NJ: Prentice-Hall.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: Introduction, second edition.* Cambridge, Massachusetts: The MIT Press.