Cyclic, Complete Graphs and Time Dependencies Represented in a Solar System Model Based on Newton's laws of Motion

Student ID: 1060530

Abstract

Astronomical models exclusively based on gravity rules currently defines the social understanding of the planetary movement in the Solar System. However, even the simplified construal of a Solar System based on Newton's laws of motion, reveals that in order to establish the dependencies among observables it is required to define cyclic and complete graphs that deserves discussion from conceptualization and implementation perspective in the definitive notation context.

1. INTRODUCTION

The Universe has had several representations that fit with the experience, religion and science knowledge of each individual, these models scale out to a social cosmological conceptualization creating accepted truths that have rule the societies during several centuries.

However, even starting from the times of the celestial model (Crowe, 2001) to the 21th century the experience of the common people is limited to the perceived movement of the Sun, the Moon, the starts, the change of seasons, tides and the day and night. Moreover, nowadays the most advance technology still quite powerless in relation to the available methods and tools that could help to induce change in astronomical entities to facilitate experiments, observations and validations. Hence the only effective alternative is to construct models and support our understanding and experience through simulations.

Advanced or simple models of planetary systems can be represented in construals where astros are the agents which specific properties that create forces that induce change on others agents' properties, thus creating dependencies that can be represented using definitive notations (Beynon, 1986).

This paper presents the conceptualization and implementation of a Solar System model that uses tkeden for its realization and describes the experience of the developer as a user from the point of view of the Empirical Modeling (EM) concepts.

2. CONCEPTUALIZATION

2.1 Model Definition

The selected model aims to permits the user to explore different scenarios by allow him to have an active role in the system. One of the goals is to permit to add and remove dependencies in order to experience the system changes after his agency.

In order to support clearly the model definition, an algebraic definition is given.

The model can be defined as the set $S = \{ A_i \mid 1 \leq i \leq n\}$, where n is the number of agents, A_i is each astral body and S represent the model, construal or system..

The only type of dependency used in the model is defined in the universal law of gravity shown Fig.1, which indicates the relevant properties to take into account. The figure defines F_g as $F_{(1,2)}$ and its generalization will be F_{ij} as the force between the astro A_i and A_j with mass M_i and M_j.

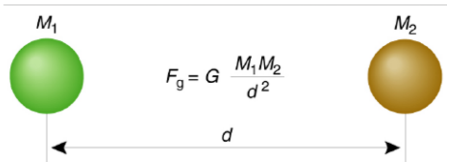

Fig.1. Universal law of gravity formula (Rowan, 2006)

From Fig.1, could be deduced that each body must have mass and a position, the former is considered a static property while the latter change over the time. The position is defined as the pair $P_i=[x_i,y_i]$ restricting the model to a 2D representation. The position allows calculating the distance between a pair of bodies with the formula shown the Fig.2.

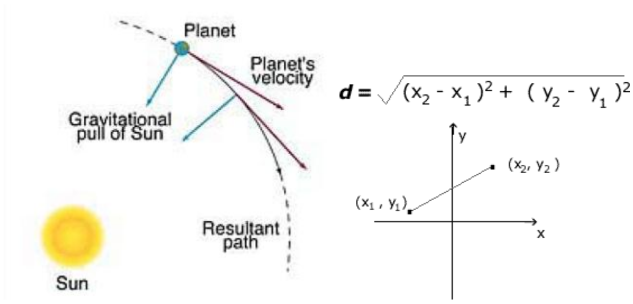

Fig.2. Velocity vector and Distance Formula(Rowan, 2006)

In addition, each agent must have a speed and direction in order to preserve the initial and past effect of the gravitational dependency. This is represented by the velocity vector $V_i = [Vx_i, Vy_i]$.

On the other hand, due to the model aims to allow the user to create and remove dependencies and create different scenarios, each body must have an associated set of Astros which can influence its dynamic properties and will be represented as

$D_i = A_i \rightarrow A_j \wedge A_j \in S \wedge A_i \neq A_j$.

Therefore in order to define an mutual gravity dependency, it is necessary that $(A_i, A_j) \in D_i \wedge (A_j, A_i) \in D_j$.

In relation to time, the whole model defines T as the global past time at which P_i and V_i had taken their values and T_{next}, the time where the model must update. Additionally, the model defines $T_{span} = T_{next} - T$, thus representing the span of time since the previous state and the next state.

Furthermore, in order to be coherent with the use of time each of the dynamic properties must have a current and next representation. The former represented by the already defined, P_i and $V_{i,}$ and the latter by $P_{i\ next}$ and $V_{i\ next}$, which refers to the next state in $A_{i\ next}$.

Given the previous definitions, only rest to the express the dependencies based on next formulas.

$V_{i\ next} = computeV_{next}\ (V_i, T_{span}, D_i)$

$P_{i\ next} = computeP_{next}\ (P_i\ , V_{i\ next})$

The simple algebraic definitions will help to contextualize clearly the model definitions thought this document especially the Definite Notation section.

2.2 Definitive Notation

Universe construals are conceived initially more with the purpose of understand our cosmos without an explicit intention of solving and specific problem, though mayor changes in its conceptualization has been related with big leaps toward an advanced human society.

Thus, the interaction with the Solar System model gives to the user an additional opportunity to observe and experience the accepted theories that rule our cosmos and formulate new configurations based on its individual interests.

In relation with the conversation between the user and the model, simulations initially seem restricted to the initial definition of the properties and dependencies amongst each agent, i.e. astral bodies, and the ignition of the simulation. The ignition which triggers a controlled chain reaction amongst the astral bodies A_i and the user just contemplates without additional involvement.

The passive role during simulations execution it is usually a desired property due to guarantee that their behavior does not depend on factors not initially conceived. However, the experience of the users is enriched by the ability to modify the relations (D_i) during the model execution and the possibility to control.

The algebraic model definition facilitates the conceptualization of the construal by means of definitive notation (Beynon, 1986), thus is straightforward establish the relation with user's understanding model as follows:

(a) Each astro A_i represents its state by the tuple (M_i, P_i, V_i, $P_{i\ next}$, $V_{i\ next}$).

(b) The functional relations are defined by $computeV_{next}$ and $computeP_{next}$.

Therefore, the model behavior is expressed in simple functional declarative sentences that describe the dynamics and relations of the system.

In order to have a broad view of the internal dependencies in each astro, it is useful to use the spreadsheet analogy in conjunction with the defined algebra to depict as shown in Fig.3 the internal relations amongst all A_i properties.

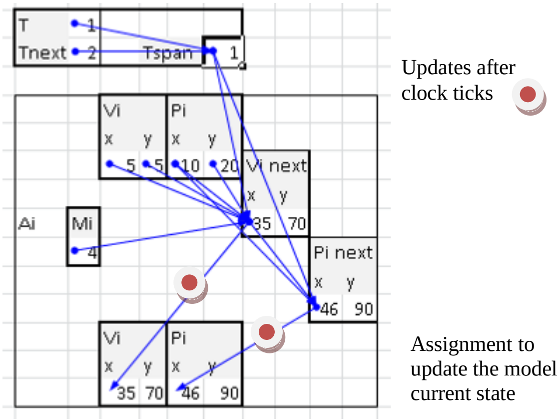

Fig.3 SpreedSheet View

The spreadsheet serves to express the state of dependencies as it is used in the spreadsheet ←→ definitive analogy and later will be used to describe certain properties of the model.

3. Implementation

This section presents the most relevant aspects that were taken into account or found during the development that has relevancy in the EM context.

3.1 Numerical Stability

An important characteristic of an astronomical model is that it handles with astronomical measures as the mass of a star in a scale above 10^{30}, planets and satellites whose mass usually above 10^{22} and the gravitational unity is $6.67e^{-11}$.

Fortunately, due to our interest is focused in those astros in the scale above comets and in order to avoid floating point problems the mass used in the model is $m_i = m_{i\ real} * 10^{-20}$. Moreover distances are represented in scales around 10^{10} and in relation to time year are represented in a scale of seconds.

At first glance, it does not seem to have too much relation with EM concepts. However is surprisingly how quickly a model behaves erratically in case that these technical issues are not handled. Thus, the model must deal with extremely huge and tiny values that must be calculated with precision in order to present a construal that reflect the user understanding of the universe.

Furthermore, the fundamental concept of dependency in EM induces particular effects of how numerical approximation errors could propagate in the calculations.

In the 2.2 Definitive Notation section the dependencies were defined as:

$V_{i\ next} = computeV_{next}\ (V_i, T_{span}, D_i)$

$P_{i\ next} = computeP_{next}\ (P_i\ , V_{i\ next})$

The point to highlight is that it means that in order to compute the $V_{i\ next}$, must be considered at the same time the state of A_j in the range of D_i and the resulting $V_{i\ next}$ is just considered one time to calculate $P_{i\ next}$.

From the user perspective could be easier to define individual dependencies in the following way:

$P_{i\ next}$, $V_{i\ next}$ = computeGravitationalEffect (T_{span} , A_i , A_i)

However, the previous dependency definition propagates more the error generated during the computation.

Therefore, with the purpose to be faithful to the model the dependencies in some way define a process in order to reduce floating point effects.

3.2 Cyclic Dependencies

The spreadsheet representation in Fig.3 reveals that even though the model definition explain clearly the model dynamics, in the end, such expected behavior does not happen until $V_{i\ next}$, $P_{i\ next}$ transfers its state to V_i , P_i and produces real motion.

Definitive notations in its earliest conceptualization, constrains the type of dependencies that could be established to those that do not create directly or indirectly a cyclic evaluation (Beynon, 1986). In general this is a best practice that helps to avoid ambiguity. However, this is a hard restriction, perhaps motivated from the design problems that create an infinity loop, the possible waste of computational resources or simply to avoid a crash in the system.

Nonetheless, the concept could be released of this constraint and delegate the problem to the language or interpreter, in order to enable the initial dependency definition to be close to the normal user.

Therefore, as an example the definition of computeP_{next} could be simplified to P_i = computeP (P_i ,V_i).

The advantage of having this representation is that there is no need to create additional mechanism that must fill the missing link between the current and next state of an observable. Nevertheless, it does not mean that computeP is a better representation than computeP_{next}, due to the latter it is clearer and has less ambiguity,

This option to define cyclic dependencies has a realization in DOSTE (Pope, 2011), which defines the “will be” statement. This new operator, allow the definition of dependencies whose effect is generate in a near future state.

This feature was considered during the initial project stages as a good approach to the development of a Solar System model, sadly, the availably of CADENCE during the time assigned to the development does not allowed its use, but from the EM perspective this characteristic permits to defines our world’s understanding in an easier way.

Therefore, the solution required the definition of a coordinator process that copies the value of the next properties to the current observables. The execution of this process is tailored to a time model pulse based on eden-clocks.

3.3 Deep and Multiple Dependencies

It is not uncommon to find relations with a cardinality superior than 1 in the world, even it is sometimes hard to find models that do not have it and is evident it database schemas where the 1..* cardinality is one of the fundamental types of relationships.

In the simplified Solar System model, the many to many relationship amongst astros established in D_i , which represents one of the main advantages of the model, but also one of the challenges from the implementation point of view.

While EDEN is an excellent interpreter that allows the definition of dependencies between simple variables, it lacks of the option to define a 1 to many dependency, as is the case of:

$V_{i\ next}$ = computeV_{next} (V_i, T_{span} , D_i)

Currently, EDEN support a list type that is enough to create any kind of data structures, however the *is* operator is not triggered when the value of any of elements have changed, but only when an operation have change the actual reference of any of the items that the list contains.

The problem then arises from the fact that there is no dependency amongst the list and the internal state of its items.

In general terms, this available type of dependency in EDEN could be classified as a shallow dependency, whereas the desired one could be called deep dependency.

An alternative approach that was tried, was the used of the EDEN operator ~>, which in the right hand side (RHS) defines a list of the observables that have a dependency on the value defined in the left hand side (LHS). However, even though a query to the LHS operand shows the expected dependencies, a change in the observable do not trigger an action on the RHS;

The lack of these features became a major problem, since one of the goals was to implement a construal following a definitive notation and thus following an EM approach. Besides, CADENCE does not define clearly a list structure and hence multidependency is no viable.

Therefore, the solution was to use the same coordinator process used to resolve the cyclic dependencies in order to delegate him the responsibility to trigger the functions that initially were expected to be defined directly using is operators.

3.4 Weak dependencies

An additional type of dependency that could be defined is a weal dependency, that differs to the others in that indicates that certain circumstances it could be ignored and perhaps removed, thus no just the model might achieve better performance but the understanding of the user can became more easier.

The existence of this type of dependency is revealed in the universal law of gravitation Fig.1, where is evident that in the case of a huge d, the total effect of the dependency could be ignored and the model could be simplified.

The implementation of this type of dependency could be associated with the concept of weak reference already implemented in some major languages that user garbage collector, where in the case of an object that is not frequently used, even though it is referenced, it could be removed from the memory in order to save resources.

In the case that a removed object is required again, usually using a factory pattern the object is regenerated or maybe loaded from its last persisted state.

The point in this case is that our universe defines this type of relations and construals might improve the user’s experience by presenting to the user a cleaner model, without losing too much generality.

3.5 Tree Dependencies

The Solar System has been conceptualized in uncountable ways, but Kepler's model is the first one that based on Tycho Brahe measurements, formulate a mathematical model that describes planets' orbits around the sun and its relation to time with mathematical accuracy support(Crowe, 2001).

In Kepler's theories, the Solar System could be represented in a hieratical structure where the planets are tied to the Sun with an invisible elastic band and the same happens with the relation amongst planet's and theirs satellites, thus describing an elliptical path during its translational movement.

Kepler's model differs substantially from algebraic model definition presented early in this document, given that they formulated the dependency as follows:

$P_{i\ next}$ = computeP_{next} (P_i , $P_{j\ next}$)

$V_{i\ next}$ = computeV_{next} (V_i, T_{span} ,)

However, these adapted definitions do not change the initial user's experience and observations, but they do restrict the model to a directed tree structure.

Although, the fixed directed tree structure that represents the dependencies in Kepler's model are very simple and effective, it do not explain other planetary systems like the ones with two starts and the effect of astronomical encounters of comets.

The initial configuration of the default model included represents a tree structure of the Solar System, which helps the users to experience the normal and expected behavior of the system. Nonetheless, in the next section will be presented the model achieved with the project.

3.6 Complete Graph Dependencies

Newton's laws of motion establish that this astronomical dependency could be explained by the gravitational force exerted amongst the sun and the planets and works with the same accuracy in satellites systems. In addition one of the highlighted conclusions of Newton's laws is that from its formulation could be derived the tree laws formulated by Kepler(Crowe, 2001).

Beside, in Newton's model the gravitational forces create a mutual relation that could be represented in a directed complete graph structure, but because of the relative distances between the sun, the planets and the satellites the effect could sometimes be ignored.

The possibility to create any kind of dependencies amongst the astral bodies, allows the user to create what if cases that helps to gain better understanding of the universe.

In the end, a complete graph represents the extreme configuration available to experiment with the universe/

4. DISCUSION

The use of algebraic definitive notations in order to define models ease the understanding of the dependencies and general behaviors expected in a construal. However if the dependencies are fixed then the models became just a static views of a specific scenario, like the preconceived tree layout of the Solar System.

A flexible approach founded in EM concepts is to allow the user to play with the internal structure of the model in order to build its own construal's starting from a referent but not limited to it.

Therefore the ability to change the general configuration of the scenario gives to the user a playground where he might really experience the properties and effects found in the referent by means of the construal without losing the model essence.

The Solar System model is a good scenario to conceptualize and generalize dependencies which might be found frequently in specific kind of scenarios.

However, it is noted that the tools available during the Solar System development project, do not allow implementing all the required types of dependencies in order to follow more closely a definitive notation. Thus, the alternative to fill the missing technical gaps were covered by a traditional procedural implementation.

A suggested approach is to study dependency concept and its implementations with the aim to establish specialization and generalizations approaches that can contribute to the development of construal that could follow closely the EM approach.

Nonetheless, is good to notice that tools like EDEN have the ability to create construals that could be use a mixed approach. Hence, in the end must be taken into account that following just a specific approach it is no necessary mandatory or even be considered the best solution.

5. CONCLUSION

To conclude, is good to notice that the experience gained during the development serves as the interaction with EM as a referent in order to build personal understanding of the paradigm and the project became a construal to observe an experiment, its current implementations limitations, strengths

As in any field, there is room for additional development and the good thing is that the paradigm deserves the effort.

REFERENCES

Beynon, M. (1986). *Definitive Notations for Interactions*. **In**: Coventry,Coventry: Warwick, Computer Science, Empirical Modelling.

Crowe, M. (2001). Theories of the World from Antiquity to the Copernican Revolution (New York: Dover Publication).

Pope, N. (2011). *Draft Doctoral Tesis Chapter 4*. **In**: Warwick,Warwick: University of Warwick.

Rowan (2006). *The Laws of Motion*. **In**: http://rigel.csi.cuny.edu/.