Knowledge Graph Project - 三体

Julia Rennert (李珠丽), 邓可立, 陆渝霖, 倪雨洲

2024 年 1 月 2 日

1 Objective

We early on agreed that we wanted to construct a knowledge graph for a book, computer game or other fandom. As our group members were from multiple countries and we had to carter to the knowledge of two nations, we decided on the internationally well-known book The Three Body Problem (三体) and its sequels The Dark Forst (黑暗森林) and Death's End (死神永生) by Liu Cixin (刘慈欣), extrapolating its characters and plot points.

Since one of our group members had a specialization in the construction of ontologies and ontologies had a major focus later on in the lecture, we decided on writing an owl file that heavily focused on the aspect of knowledge by inference. Our goal was not only to depict the relevant characters and scenes of the books, but to remodel the structure of the plot and allow for deductions relations and involvements into the plot.

2 Related Works and Sources

While OpenGK already has data in the form of a json file with only about 20 characters, it is far from comprehensive and lacked the detail knowledge and relations we needed to form comprehensive deductions. We therefore disregarded it, drawing our knowledge from the Chinese and English fandom wikis, wikipedia and books themselves. For links to the sources, please see the end of the report.

3 Our Ontology

This part of the report explains our ontology, starting from the T-Box over the A-Box to the SWRL rules. Informations about the R-Box are implicitly covered in the T-Box section. The ontology was developed using Notepad++ and Protégé.

3.1 Remark about the Reasoner

The ontology was initially constructed with the Pellet reasoner in mind. As the work progressed and the ontology became more complex, we switched to HermiT, noting that the complex SWRL rules we had constructed for Pellet, also worked for HermiT 1.4.3.456 while also reducing the reasoning time by a multiple. Therefore, the reasoning with the ontology should be done with the aforementioned HermiT reasoner. Some hints on the former use of Pellet like for example the encoding of transitive roles into SWRL rules can still be found.

3.2 T-Box

The T-Box is held comparatively simple, only consisting of the following classes: Person, Species, Book, Era, Scene, Organization, Fraction

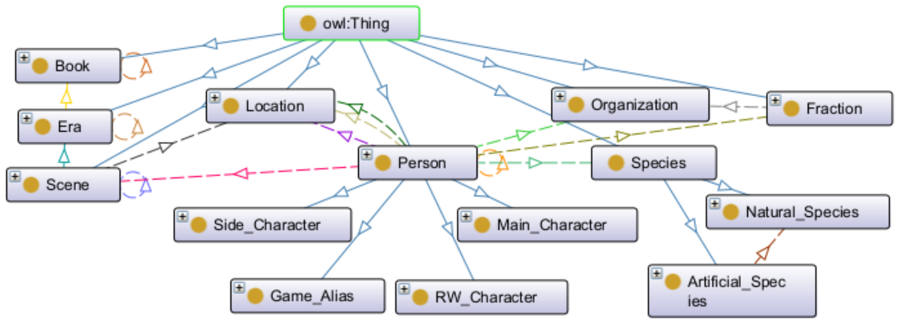


图 1: Classes and object properties of the T-Box. The subclass relations are marked by blue arrows, the other arrows display the most important object properties.

and Location. To allow a better understanding of the person and its relevance, the class Person is subdivided into the subclasses Main Character, Side Character, Game Alias and Real World Character, while the Species has the subclasses Artificial Species and Natural Species in order to allow creation and ownership of artificial beings like robots. The classes are distinct and the parent classes have the necessary closing axioms to facilitate reasoning.

The Book class is constructed in a way that allows for books to have an order, its individuals creating a linked list. The object properties "right after book" and "right before book" are bilateral links between the neighbors. "book after book" and "book before book" are adding a transitive component and facilitate a link to all previous or later books. In the same way, eras and scenes are ordered. This order allows that situations that occurred in previous scenes can have a consequence on later scenes. Please note that the order is chronological, not temporal.

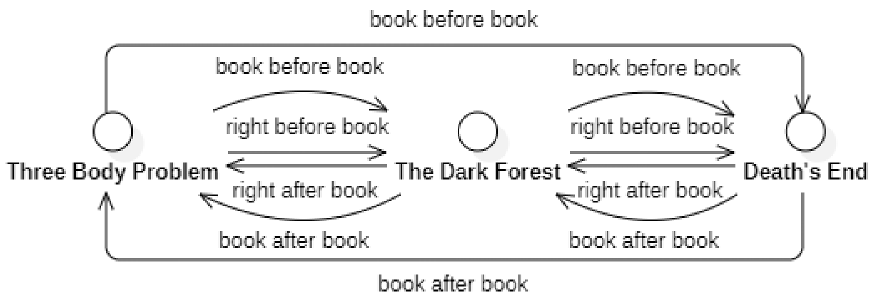


图 2: Bidirectional list structure example on basis of the book class and its individuals.

Besides the classes and roles, the ontology structure also consists of some data properties. For example, the scenes have a description and a year, while the persons have a gender, occupations, titles, birth date, and death date. Some of those properties like the description of the scenes are fully informative, others like the gender of the person are used during the reasoning process. The ontology consists of 12 data properties and 51 object properties. A full list is attached.

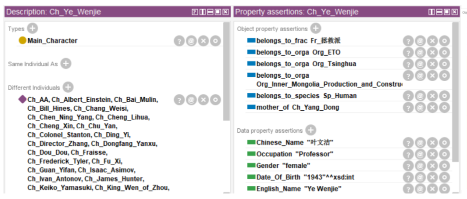
Class Name	Number of Individuals
Person	65
Species	3
Scene	41
Era	7
Book	3
Location	22
Organisation	26
Fraction	3

图 3: Number of Individuals per Class

3.3 A-Box

The A-Box holds the actual individuals. Figure 2 lists the existing individuals of the ontology. While our main focus was on the characters and scenes, we extracted the locations where the scenes take place and organizations the characters belong to to combine them and facilitate reasoning. While the enumeration of characters and the attributes given to them is detailed, not every single scene of the three books is integrated into the ontology, as our focus was on reasoning on them rather than retelling the whole series.

Figure 4a shows an example of the character Ye Wenjie with its asserted properties. The character is of the type Main Character and therefore by subsumtion also of type Person. Therefore it can be part of an organization and the fractions of an organization, as seen in the object property assertions. Each character also belongs to a species, in Ye Wenjie's case human. As you can see, Ye Wenjie has her daughter Yang Dong annotated. The data assertions include her name in both Chinese and English, her occupation as Professor, gender, and date of birth. Who has read the book knows that she has more relations than just her daughter. These relations are implicitly in the ontology as seen in the inferred property assertions after reasoning in figure 4b. The individuals of each class are just as the classes themselves distinct of each other.


Description: Ch_Ye_Wenjie

Types
Main_Character

Same Individual As

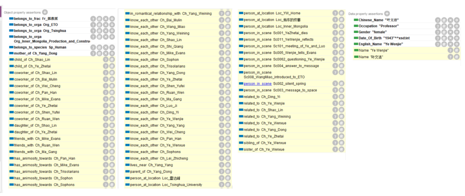
Different Individuals
Ch_AA, Ch_Albert_Einstein, Ch_Bai_Mulin, Ch_Bill_Hines, Ch_Chang_Weisi, Ch_Chen_Ning_Yang, Ch_Cheng_Lihua, Ch_Cheng_Xin, Ch_Chu_Yan, Ch_Colonel_Stanton, Ch_Ding_Yi, Ch_Director_Zhang, Ch_Dongfang_Yanxu, Ch_Dou_Dou, Ch_Fraisse, Ch_Frederick_Tyler, Ch_Fu_Xi, Ch_Guan_Yifan, Ch_Isaac_Asimov, Ch_Ivan_Antonov, Ch_James_Hunter, Ch_Keiko_Yamasuki, Ch_King_Wen_of_Zhou,

Property assertions: Ch_Ye_Wenjie

Object property assertions
belongs_to_frac Fr_拯救派
belongs_to_orga Org_ETO
belongs_to_orga Org_Tsinghua
belongs_to_orga Org_Inner_Mongolia_Production_and_Constru
belongs_to_species Sp_Human
mother_of Ch_Yang_Dong

Data property assertions
Chinese_Name "叶文洁"
Occupation "Professor"
Gender "female"
Date_Of_Birth "1943"^^xsd:int
English_Name "Ye Wenjie"

(a) Asserted


Object property assertions
belongs_to_frac Fr_拯救派
belongs_to_orga Org_ETO
belongs_to_orga Org_Tsinghua
belongs_to_orga Org_Inner_Mongolia_Production_and_Constru
belongs_to_species Sp_Human
mother_of Ch_Yang_Dong
child_of Ch_Shao_Lin
child_of Ch_Ye_Zhetai
coworker_of Ch_Shao_Lin
coworker_of Ch_Bai_Mulin
coworker_of Ch_Wei_Cheng
coworker_of Ch_Pan_Han
coworker_of Ch_Mike_Evans
coworker_of Ch_Ye_Zhetai
coworker_of Ch_Shen_Yufei
coworker_of Ch_Ruan_Wen
daughter_of Ch_Shao_Lin
daughter_of Ch_Ye_Zhetai
friends_with Ch_Mike_Evans
friends_with Ch_Ruan_Wen
friends_with Ch_Ma_Gang
has_animosity_towards Ch_Pan_Han
has_animosity_towards Ch_Mike_Evans
has_animosity_towards Ch_Trisolarians
has_animosity_towards Ch_Sophon
has_animosity_towards Ch_Sophons
in_romantical_relationship_with Ch_Yang_Weining
know_each_other Ch_Bai_Mulin
know_each_other Ch_Wang_Miao
know_each_other Ch_Yang_Weining
know_each_other Ch_Shao_Lin
know_each_other Ch_Shi_Qiang
know_each_other Ch_Mike_Evans
know_each_other Ch_Sophon
know_each_other Ch_Trisolarians
know_each_other Ch_Yang_Dong
know_each_other Ch_Ye_Zhetai
know_each_other Ch_Shen_Yufei
know_each_other Ch_Ruan_Wen
know_each_other Ch_Ma_Gang
know_each_other Ch_Luo_Ji
know_each_other Ch_Ding_Yi
know_each_other Ch_Ye_Wenjie
know_each_other Ch_Yang_Yang
know_each_other Ch_Wei_Cheng
know_each_other Ch_Pan_Han
know_each_other Ch_Ye_Wenxue
know_each_other Ch_Sophons
know_each_other Ch_Lei_Zhicheng
lives_near Ch_Yang_Yang
parent_of Ch_Yang_Dong
person_at_location Loc_雷达峰
person_at_location Loc_Tsinghua_University
person_at_location Loc_YW_Home
person_at_location Loc_杨东的坟墓
person_at_location Loc_Inner_Mongolia
person_in_scene Sc001_YeZhetai_dies
person_in_scene Sc011_YeWenjie_reflects
person_in_scene Sc101_meeting_of_Ye_and_Luo
person_in_scene Sc005_Wenjie_tells_Evans
person_in_scene Sc0062_questioning_Ye_Wenjie
person_in_scene Sc004_answer_to_message
person_in_scene Sc008_WangMiao_introduced_to_ETO
person_in_scene Sc002_silent_spring
person_in_scene Sc003_message_to_space
related_to Ch_Ding_Yi
related_to Ch_Ye_Wenjie
related_to Ch_Shao_Lin
related_to Ch_Yang_Weining
related_to Ch_Ye_Wenxue
related_to Ch_Yang_Dong
related_to Ch_Ye_Zhetai
sibling_of Ch_Ye_Wenxue
sister_of Ch_Ye_Wenxue

Data property assertions
Chinese_Name "叶文洁"
Occupation "Professor"
Gender "female"
Date_Of_Birth "1943"^^xsd:int
English_Name "Ye Wenjie"
Name "Ye Wenjie"
Name "叶文洁"

(b) Inferred property assertions

图 4: Individual Ye Wenjie with her type, distinctions and property assertions.

3.4 SWRL rules

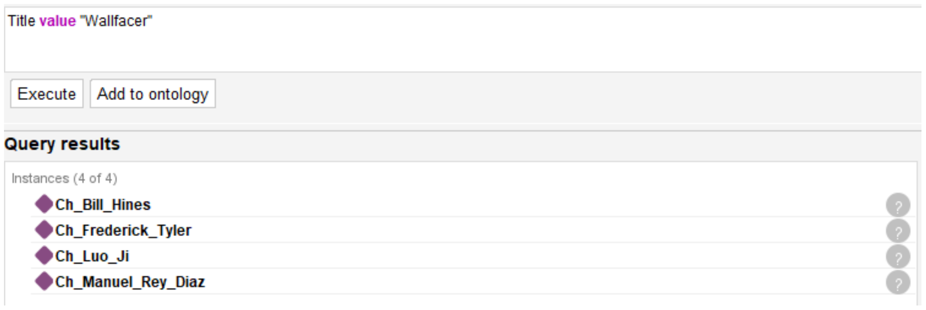
The SWRL rules cover transitivity and role axioms to facilitate reasoning both with HermiT and Pellet, as much as rules for asserting information about the on-goings in the books and the relations between the characters. The latter have to cope with the delicacies of the open world assumption. While the individuals and classes are both distinct which eases the task of reasoning, it would have been to complex and not in the spirit of an open world ontology to close all the roles. Therefore, the used negative statements has to be handled with care. Nonetheless, our ontology allows comprehensive reasoning, as can be seen in the next section. There are 27 rules in total. A complete list is in the appendix.

4 Example Queries

Now let's test the ontology with a few examples. The following subsections include each a goal, a DL query, an explanation of the rules and axioms that are triggered and the output of the inquiry. Other possible inquiries would be for example which characters interacted during a scene, on which locations they must have been, which characters were coworkers or which characters must know each other. For simplicity, we assume that characters who are related or are in the same organization also know each other.

4.1 Who are the Wallfacers?

The easiest type of query is just to extract pre-coded knowledge, for example, all of the characters that have the title "Wallfacer". In this case, we only need to query all the individuals and find those have the attribute Title with the value "Wallfacer".



```
Title value "Wallfacer"
```

Query results

Instances (4 of 4)

- Ch_Bill_Hines
- Ch_Frederick_Tyler
- Ch_Luo_Ji
- Ch_Manuel_Rey_Diaz

4.2 Who of the male characters has a child?

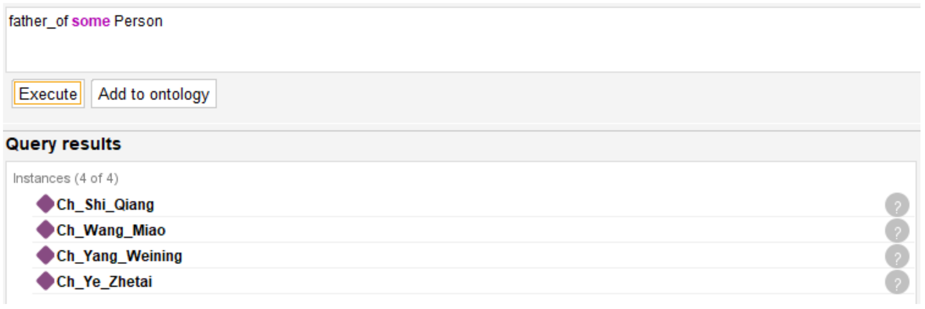
To assert which of the characters in the ontology have a child, we need the child_of object property, the Gender data property and SWRL rule for fatherhood. The former two are part of the static information we have embedded into our ontology. The SWRL rule has the following form:

```
Gender(?x, "male") ^ parent_of(?x, ?y) ->
father_of(?x, ?y)
```

The rule implies that somebody who is male and the parent of a child, is also automatically the father of that child. We can now use this rule to find out who is a father by using this rule.

4.3 Is Mike Evans already death at in the last scene of the first book?

To answer this question, we need to know in which scene Mike Evens died and whether this scene is before or after the scene "Ye Wenjie reflects"


father_of some Person

Query results

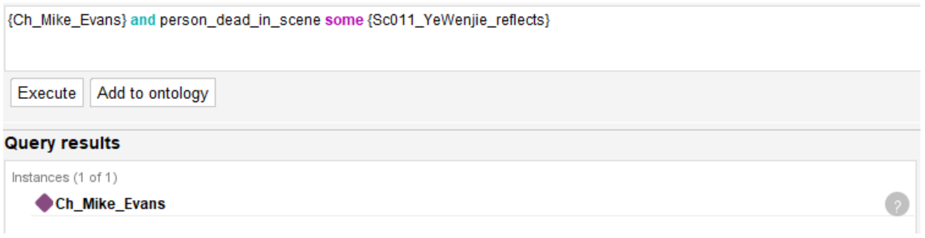
Instances (4 of 4)

Ch_Shi_Qiang
Ch_Wang_Miao
Ch_Yang_Weining
Ch_Ye_Zhetai

which is the last scene of the first book. Here, the list structure in which we ordered the scenes helps to find a conclusion since we have a strict order for each and every scene and therefore can compare them and deduce information for a future scene.

```
person_dies_in_scene(?x, ?y) ^ scene_after_scene(
?y1, ?y) ->  person_dead_in_scene(?x, ?y1)
```

In the book, Mike Evans dies during the ambush on his ship R̈edemption Dayẅhich happens a few chapters before the conclusion of the book. As can be seen, the output tells us that Mike Evans indeed has already died before the final scene.


{Ch_Mike_Evans} and person_dead_in_scene some {Sc011_YeWenjie_reflects}

Query results

Instances (1 of 1)

Ch_Mike_Evans

4.4 Who hates whom?

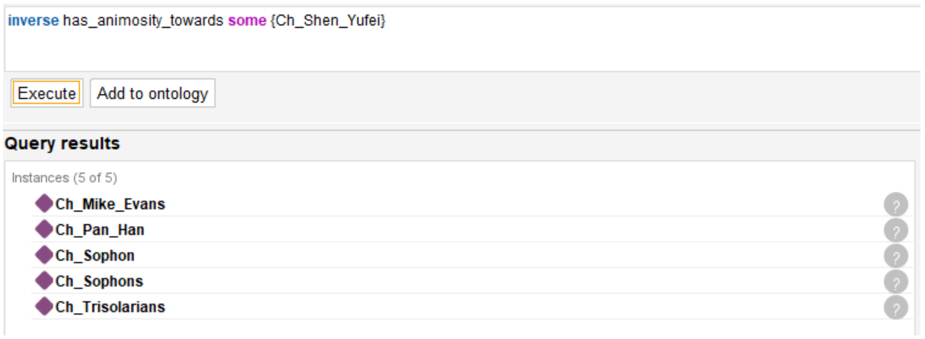
The rules state two different reasons why a Person could have an animosity towards another Person. The first one is, that they belong to the same organization, but different fractions, as is the case for several members of the ETO. The rule stating this fact looks as follows:

```
belongs_to_frac(?p1, ?f1) ^ belongs_to_frac(?p2,
?f2) ^ differentFrom(?f1, ?f2) ^
fraction_of(?f1, ?o) ^ fraction_of(?f2, ?o)
-> has_animosity_towards(?p1, ?p2)
```

The second reason is, that the individuals belong to rivaling species, for example the Trisolarians and the humans. This rule is quite easily to embed:

```
belongs_to_species(?p1, ?s1)
^ species_hates_species(?s1, ?s2)
^ belongs_to_species(?p2, ?s2)
-> has_animosity_towards(?p1, ?p2)
```

Now we also want that, a species who belongs to another species like the Sophons, shares its owner's animosity. We therefore add a role axiom using the species_belongs_to_species role.



```
inverse has_animosity_towards some {Ch_Shen_Yufei}
```

Query results

Instances (5 of 5)

```
Ch_Mike_Evans
Ch_Pan_Han
Ch_Sophon
Ch_Sophons
Ch_Trisolarians
```

An additional useful restriction of the rule would be that people of organizations that have a positive inclination towards a species are exempt from hating it. But this rule is hindered by the open world assumption. If one was to set an organization as positive not only would all characters belonging to this organization be exempt from hating the species in question, but also all other persons in the ontology. The reason is that while it isn't explicitly stated, the ontology allows the possibility that all characters belong to the organization. Therefore they also could all be positive towards the species. The rule would be rendered useless. Here we come to the limits of ontologies based on the open world assumption.

5 Evalution with OOPS!

To validate our ontology, we used the ontology pitfall scanner, better known as OOPS!. This online app points out the major flaws of an ontology. The pitfalls are classified as critical, important or minor. Since the whole ontology was too big for OOPS! to process, we excluded individuals and rules from the check and therefore searching the T-Boy for flaws.

The ontology had no critical errors, but some delectable minor issues and two important remarks. The majority of the pointed out flaws were attributed to stating the domains and ranges implicitly rather than writing them explicitly to every property even if they were already inferred. If the ontology is imported with its inferred axioms, this pitfall vanishes. We nonetheless added the domains and ranges to each property in order to fulfill the demands of the pitfall scanner. To declare a license, we imported dc and used its rights annotation property.

Most of the missing reverse relationships are not needed because they are either not relevant for our ontology or already declared by SWRL rules, but we added additional symmetric properties where needed. The property chain pitfall doesn't give any indication of the place in which it occurs nor does any property chain actually have only one property. The remaining annotation pitfall is minor and caused by actions like using special symbols and spaces in the annotated labels, so we simply interchanged forbidden symbols.

6 Critique and Conclusion

Our ontology introduces a comprehensive overview of the characters, organizations and scenes in the 三体 universe. It also spans a web, allowing

Pitfall	Importance	Number
Missing domain or range properties	Important	33
No license declared	Important	
Inverse relationships not explicitly declared	Minor	26
Misusing ontology annotations	Minor	36
Creating a property chain with just one property	Minor	1

图 5: Pitfalls detected by OOPS!

a deduction on the temporal order between the books, eras and scenes, and the relation of the characters to each other and to the happenings of the books.

To address possible shortcomings; as our group members are from diverse origin, the ontology is partly English, partly Chinese. This might impede the usage for individuals that are not able to understand both languages. It also displays the common problems of ontologies that are equipped with the whole force of both STROIQ(D) and SWRL, slowing down considerable with increasing size. We therefore had to do a trade-off between size and reasoning speed. Nonetheless, the ontology is a good example of the might and limits of complex ontologies.

As for further work, the ontology has the capacity to be further expanded by rules that infer interesting facts about the characters and books' contents. For example, while we annotated the dates of eras, scenes and events like death and birth, they are not yet included into the reasoning, because reasoning on dates and numbers is not in the scope of traditional ontologies. Nonetheless, they could be used for further information gathering.

7 Sources

- https://santi.huijiwiki.com/wiki/
- https://three-body-problem.fandom.com/wiki/Three_Body_Problem_Wiki

- https://www.douban.com/group/topic/16436745/?_i=3233689cM2QT3P
- https://zhuanlan.zhihu.com/p/397302462
- https://zhuanlan.zhihu.com/p/94194566
- https://www.jianshu.com/p/ad2d1c8f39d2
- https://de.wikipedia.org/wiki/Die_drei_Sonnen
- https://three-body-problem.fandom.com/wiki/Three_Body_Problem_Wiki

8 Appendix

8.1 Object Property List

Name	Domain	Range
belongs to frac	Person	Fraction
belongs to orga	Person	Organization
belongs to species	Person	Species
book after book	Book	Book
book before book	Book	Book
born at	Person	Location
boyfriend girlfriend of	Person	Person
brother of	Person	Person
child of	Person	Person
coworker of	Person	Person
daughter of	Person	Person
died at	Person	Location
era after era	Era	Era
era before era	Era	Era
era in book	Era	Book
father of	Person	Person
fraction of	Fraction	Organization
friends with	Person	Person

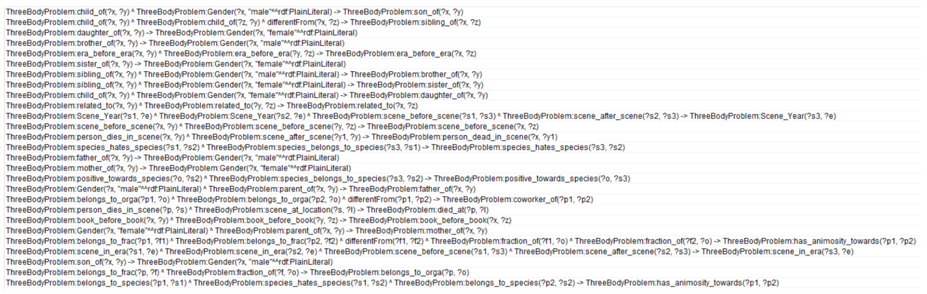
has animosity towards	Person	Person
has members	Organization	Person
in romantical relationship with	Person	Person
know each other	Person	Person
lives near	Person	Person
married to	Person	Person
mother of	Person	Person
parent of	Person	Person
person alive in scene	Person	Scene
person at location	Person	Location
person dead in scene	Person	Scene
person dies in scene	Person	Scene
person in scene	Person	Scene
positive towards species	Organization	Species
related to	Person	Person
right after book	Book	Book
right after era	Era	Era
right after scene	Scene	Scene
right before book	Book	Book
right before era	Era	Era
right before scene	Scene	Scene
scene after scene	Scene	Scene
scene at location	Scene	Location
scene before scene	Scene	Scene
scene in book	Scene	Book
scene in era	Scene	Era
scene includes person	Scene	Person
sibling of	Person	Person
sister of	Person	Person
son of	Person	Person
species belongs to species	Artificial Species	Natural Species
species hates species	Species	Species

was taught by	Person	Person

8.2 Data Property List

Name	Domain	Range
Chinese Name	Person	string
Date Of Birth	Person	int
Date Of Death	Person	int
English Name	Person	string
Era Begin	Era	decimal
Era End	Era	decimal
Gender	Person	string
Name	Person	string
Occupation	Person	string
Scene Description	Scene	string
Scene Year	Scene	decimal
Title	Person	string

8.3 Rules



```
ThreeBodyProblem:child_of(?x, ?y) ^ ThreeBodyProblem:Gender(?x, "male"^^rdf:PlainLiteral) -> ThreeBodyProblem:son_of(?x, ?y)
ThreeBodyProblem:child_of(?x, ?y) ^ ThreeBodyProblem:child_of(?z, ?y) ^ differentFrom(?x, ?z) -> ThreeBodyProblem:sibling_of(?x, ?z)
ThreeBodyProblem:daughter_of(?x, ?y) -> ThreeBodyProblem:Gender(?x, "female"^^rdf:PlainLiteral)
ThreeBodyProblem:brother_of(?x, ?y) -> ThreeBodyProblem:Gender(?x, "male"^^rdf:PlainLiteral)
ThreeBodyProblem:era_before_era(?x, ?y) ^ ThreeBodyProblem:era_before_era(?y, ?z) -> ThreeBodyProblem:era_before_era(?x, ?z)
ThreeBodyProblem:sister_of(?x, ?y) -> ThreeBodyProblem:Gender(?x, "female"^^rdf:PlainLiteral)
ThreeBodyProblem:sibling_of(?x, ?y) ^ ThreeBodyProblem:Gender(?x, "male"^^rdf:PlainLiteral) -> ThreeBodyProblem:brother_of(?x, ?y)
ThreeBodyProblem:sibling_of(?x, ?y) ^ ThreeBodyProblem:Gender(?x, "female"^^rdf:PlainLiteral) -> ThreeBodyProblem:sister_of(?x, ?y)
ThreeBodyProblem:child_of(?x, ?y) ^ ThreeBodyProblem:Gender(?x, "female"^^rdf:PlainLiteral) -> ThreeBodyProblem:daughter_of(?x, ?y)
ThreeBodyProblem:related_to(?x, ?y) ^ ThreeBodyProblem:related_to(?y, ?z) -> ThreeBodyProblem:related_to(?x, ?z)
ThreeBodyProblem:Scene_Year(?s1, ?e) ^ ThreeBodyProblem:Scene_Year(?s2, ?e) ^ ThreeBodyProblem:scene_before_scene(?s1, ?s3) ^ ThreeBodyProblem:scene_after_scene(?s2, ?s3) -> ThreeBodyProblem:Scene_Year(?s3, ?e)
ThreeBodyProblem:scene_before_scene(?x, ?y) ^ ThreeBodyProblem:scene_before_scene(?y, ?z) -> ThreeBodyProblem:scene_before_scene(?x, ?z)
ThreeBodyProblem:person_dies_in_scene(?x, ?y) ^ ThreeBodyProblem:scene_after_scene(?y1, ?y) -> ThreeBodyProblem:person_dead_in_scene(?x, ?y1)
ThreeBodyProblem:species_hates_species(?s1, ?s2) ^ ThreeBodyProblem:species_belongs_to_species(?s3, ?s1) -> ThreeBodyProblem:species_hates_species(?s3, ?s2)
ThreeBodyProblem:father_of(?x, ?y) -> ThreeBodyProblem:Gender(?x, "male"^^rdf:PlainLiteral)
ThreeBodyProblem:mother_of(?x, ?y) -> ThreeBodyProblem:Gender(?x, "female"^^rdf:PlainLiteral)
ThreeBodyProblem:positive_towards_species(?o, ?s2) ^ ThreeBodyProblem:species_belongs_to_species(?s3, ?s2) -> ThreeBodyProblem:positive_towards_species(?o, ?s3)
ThreeBodyProblem:Gender(?x, "male"^^rdf:PlainLiteral) ^ ThreeBodyProblem:parent_of(?x, ?y) -> ThreeBodyProblem:father_of(?x, ?y)
ThreeBodyProblem:belongs_to_orga(?p1, ?o) ^ ThreeBodyProblem:belongs_to_orga(?p2, ?o) ^ differentFrom(?p1, ?p2) -> ThreeBodyProblem:coworker_of(?p1, ?p2)
ThreeBodyProblem:person_dies_in_scene(?p, ?s) ^ ThreeBodyProblem:scene_at_location(?s, ?l) -> ThreeBodyProblem:died_at(?p, ?l)
ThreeBodyProblem:book_before_book(?x, ?y) ^ ThreeBodyProblem:book_before_book(?y, ?z) -> ThreeBodyProblem:book_before_book(?x, ?z)
ThreeBodyProblem:Gender(?x, "female"^^rdf:PlainLiteral) ^ ThreeBodyProblem:parent_of(?x, ?y) -> ThreeBodyProblem:mother_of(?x, ?y)
ThreeBodyProblem:belongs_to_frac(?p1, ?f1) ^ ThreeBodyProblem:belongs_to_frac(?p2, ?f2) ^ differentFrom(?f1, ?f2) ^ ThreeBodyProblem:fraction_of(?f1, ?o) ^ ThreeBodyProblem:fraction_of(?f2, ?o) -> ThreeBodyProblem:has_animosity_towards(?p1, ?p2)
ThreeBodyProblem:scene_in_era(?s1, ?e) ^ ThreeBodyProblem:scene_in_era(?s2, ?e) ^ ThreeBodyProblem:scene_before_scene(?s1, ?s3) ^ ThreeBodyProblem:scene_after_scene(?s2, ?s3) -> ThreeBodyProblem:scene_in_era(?s3, ?e)
ThreeBodyProblem:son_of(?x, ?y) -> ThreeBodyProblem:Gender(?x, "male"^^rdf:PlainLiteral)
ThreeBodyProblem:belongs_to_frac(?p, ?f) ^ ThreeBodyProblem:fraction_of(?f, ?o) -> ThreeBodyProblem:belongs_to_orga(?p, ?o)
ThreeBodyProblem:belongs_to_species(?p1, ?s1) ^ ThreeBodyProblem:species_hates_species(?s1, ?s2) ^ ThreeBodyProblem:belongs_to_species(?p2, ?s2) -> ThreeBodyProblem:has_animosity_towards(?p1, ?p2)
```