

Protégé Tutorial

Protégé – What and Where

What is Protégé? (from their webpage)

A free, open-source ontology editor and framework for building intelligent systems

Protégé is supported by a strong community of academic, government, and corporate users, who use Protégé to build knowledge-based solutions in areas as diverse as biomedicine, e-commerce, and organisational modelling.

Where to get it: <http://protege.stanford.edu/>

Useful resources

- ▶ http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf

NOTE: the manual is for version 4, but the current version is 5.1

- ▶ http://protegewiki.stanford.edu/wiki/Main_Page

Protégé – What and Where (cont'd)

Specifically, Protégé is

- ▶ a java-based application (multi-platform)
- ▶ thought for a variety of people (more than 300 thousands users)
- ▶ a GUI to help the editing of ontologies
creation, modification, reasoning, debugging, . . .

Syntax – DL, OWL, Manchester

Protégé uses the Manchester syntax

DL	OWL	Manchester
\top	owl:Thing	owl:Thing
\perp	owl:Nothing	owl:Nothing
Concept name	Class	Class
Role name	Object property	Object property
$\neg C$	ObjectComplementOf(C)	not C
$C \sqcup D$	ObjectUnionOf(C D)	C or D
$C \sqcap D$	ObjectIntersectionOf(C D)	C and D
$\exists r.C$	ObjectSomeValuesFrom(r C)	r some C
$\forall r.C$	ObjectAllValuesFrom(r C)	r only C
$(\geq n \ r.C)$	ObjectMinCardinality(n r C)	r min n C
$(\leq n \ r.C)$	ObjectMaxCardinality(n r C)	r max n C
$(= n \ r.C)$	ObjectExactCardinality(n r C)	r exactly n C

<https://www.w3.org/TR/owl2-manchester-syntax/>

Syntax – DL, OWL, Manchester – Example

DL

Person $\sqcap \exists \text{hasGender.Male}$

$(= 2 \text{ hasWheel.FrontWheel}) \sqcap (= 2 \text{ hasWheel.RearWheel})$

OWL (omitting “Object” for succinctness)

IntersectionOf(Person SomeValuesFrom(hasGender Male))

IntersectionOf(ExactCardinality(2 hasWheel FrontWheel)
ExactCardinality(2 hasWheel RearWheel))

Manchester

Person and (hasGender some Male)

(hasWheel exactly 2 FrontWheel) and (hasWheel exactly 2
RearWheel)

Convention

- ▶ concept names begin with an uppercase letter
- ▶ role names begin with a lowercase letter
- ▶ CamelBack notation for both concept and role names

An Ontology about Video Games

Assume we want to build an ontology about video games as follows.

self-standing	modifiers	relations	definable
- Game	- Genre	hasDifficulty	MultiPlatform
- NamedGame	- SinglePlayer	hasPlatform	PuzzleGame
- LoL	- MultiPlayer	hasGenre	HardGame
- Chess	- Puzzle		NormalGame
- Sudoku	- RolePlayGame		EasyGame
- WoW	- Online		LinuxGame
- Platform	- Difficulty		WindowsGame
- Windows	- Hard		MacOSXGame
- MacOSX	- Normal		...
- Linux	- Easy		

Adding Classes

Make sure to have the “Classes” tab open
Window → Tabs → Classes

The screenshot shows the Protégé interface with the 'Classes' tab selected. The 'Class hierarchy' pane on the left shows 'owl:Thing' as the root class. A red arrow points to the 'Add Subclass' button (a yellow circle with a plus sign) next to 'owl:Thing'. A dialog box titled 'Create a new OWLClass' is open in the foreground, with the following fields and options:

- Name: Game
- IRI: <http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25#Game>
- New entity options... button
- OK button
- Cancel button

Below the dialog, there are several options with plus icons:

- Instances +
- Target for Key +
- Disjoint With +
- Disjoint Union Of +

At the bottom of the window, a status bar reads: 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences' (checked).

Adding Classes

Make sure to have the “Classes” tab open
Window → Tabs → Classes

The screenshot displays the Protégé ontology editor. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window has a tab bar with 'Active Ontology', 'Entities', 'Classes', 'Object Properties', 'Individuals by class', 'OWLviz', and 'DL Query'. The 'Classes' tab is active, showing a 'Class hierarchy' panel on the left and a 'Description' panel on the right. The 'Class hierarchy' panel shows a tree structure with 'owl:Thing' as the root and 'Game' as a child. Above the hierarchy are three icons: a plus sign (Add Sibling), a minus sign (Remove Child), and an X (Delete Class). Two red arrows point from the text 'Add Sibling' and 'Delete Class' to these respective icons. The 'Description' panel on the right shows various properties for the 'Game' class, such as 'Equivalent To', 'SubClass Of', 'General class axioms', 'SubClass Of (Anonymous Ancestor)', 'Instances', 'Target for Key', 'Disjoint With', and 'Disjoint Union Of'. The bottom status bar indicates 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences' is checked.

untitled-ontology-25 (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>) : (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>)

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>) Search...

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLviz x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Game Annotations: Game

owl:Thing
Game

Add Sibling
Delete Class

Description: Game

Equivalent To +
SubClass Of +
General class axioms +
SubClass Of (Anonymous Ancestor)
Instances +
Target for Key +
Disjoint With +
Disjoint Union Of +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Class Hierarchies

It allows us to speed up the process of adding classes.

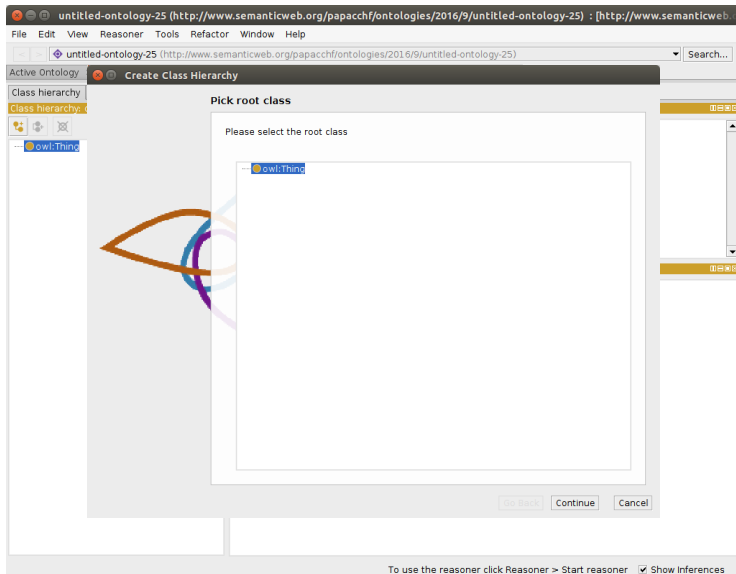
Tools → Create class hierarchy...

The screenshot shows the Protégé ontology editor interface. The title bar indicates the ontology is 'untitled-ontology-25' located at 'http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25'. The 'Tools' menu is open, showing options: 'Create class hierarchy...', 'Create axioms from Excel workbook...', 'Export OWLDoc...', 'Generate java code...', and 'Usage...'. The 'Create class hierarchy...' option is highlighted. Below the menu, the 'Class hierarchy: owl:Thing' panel shows a tree structure with three classes: 'Game', 'Genre', and 'Platform', all listed as 'SubClassOf owl:Thing'. The 'Description: owl:Thing' panel is also visible, showing various logical constraints like 'Equivalent To', 'SubClass Of', 'General class axioms', 'SubClass Of (Anonymous Ancestor)', 'Instances', 'Target for Key', 'Disjoint With', and 'Disjoint Union Of', each with a plus icon for expansion. At the bottom, a status bar reads 'To use the reasoner click Reasoner > Start reasoner' with a checked 'Show Inferences' checkbox.

Adding Class Hierarchies

It allows us to speed up the process of adding classes.

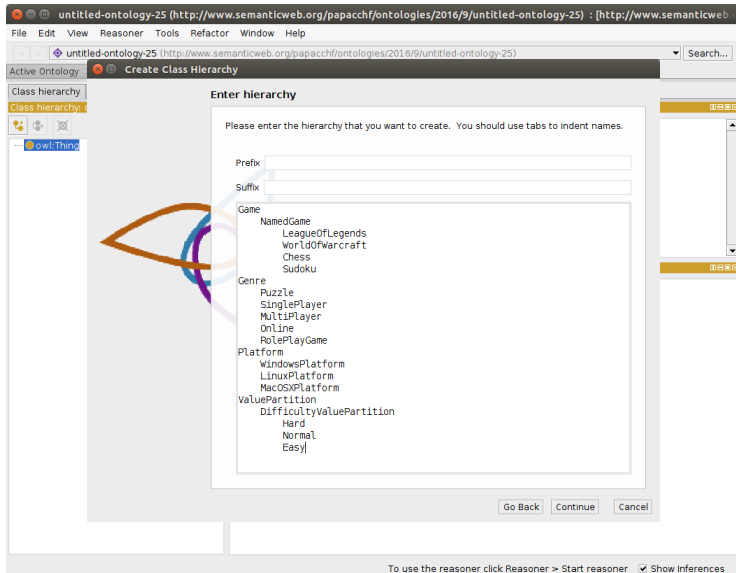
Tools → Create class hierarchy...



Adding Class Hierarchies

It allows us to speed up the process of adding classes.

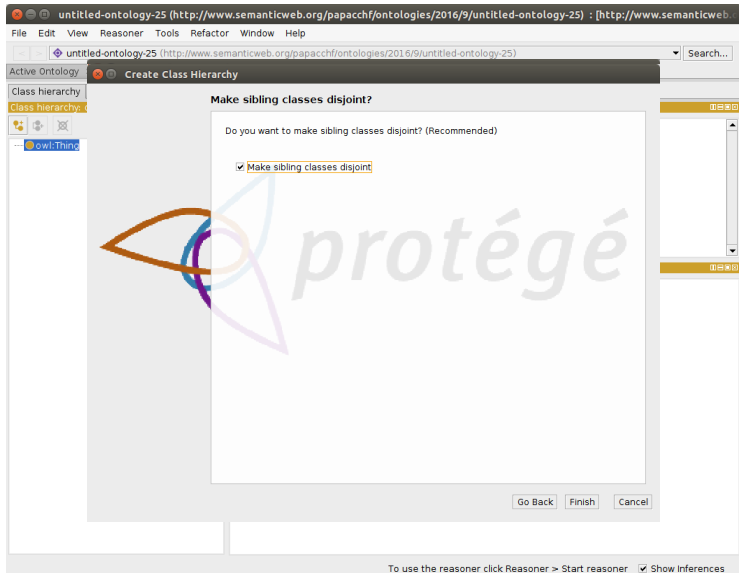
Tools → Create class hierarchy. . .



Adding Class Hierarchies

It allows us to speed up the process of adding classes.

Tools → Create class hierarchy...



Adding Class Hierarchies

It allows us to speed up the process of adding classes.

Tools → Create class hierarchy. . .

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main toolbar contains icons for navigating between different views: Class hierarchy, Class hierarchy (inferred), Annotations, and Usage. The 'Usage' tab is currently selected, showing a list of axioms involving the 'Game' class. A red arrow points to the 'Usage' tab, and another red arrow points to the 'Game' class in the left-hand class hierarchy tree.

The class hierarchy tree on the left shows the following structure:

- owl:Thing
 - Game
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSXPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition
 - Easy
 - Hard
 - Normal

The 'Usage: Game' panel on the right displays the following axioms:

- Found 10 uses of Game
 - Game
 - SubClassOf owl:Thing
 - Class: Game
 - DisjointClasses: Game, Genre, Platform, ValuePartition
 - Genre
 - DisjointClasses: Game, Genre, Platform, ValuePartition
 - hasDifficulty

The 'Description: Game' panel on the right shows the following information:

- Equivalent To: +
- SubClass Of: +
 - owl:Thing
- General class axioms: +
- SubClass Of (Anonymous Ancestor):
- Instances: +
- Target for Key: +
- Disjoint With: +
 - ValuePartition, Genre, Platform
- Disjoint Union Of: +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Class Hierarchies

It allows us to speed up the process of adding classes.

Tools → Create class hierarchy...

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main toolbar contains icons for various ontology operations. The left pane shows the 'Class hierarchy' for 'Game', with a red arrow pointing to the 'Game' class. The right pane shows the 'Usage' tab for 'Game', displaying a list of uses and a detailed description of the class.

Class hierarchy: Game

- owl:Thing
 - Game
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSXPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition
 - Easy
 - Hard
 - Normal

Usage: Game

Show: ☒ this ☒ disjoint ☒ named sub/superclasses

Found 10 uses of Game

- Game
 - SubClassOf owl:Thing
 - Class: Game
 - DisjointClasses: Game, Genre, Platform, ValuePartition
- Genre
 - DisjointClasses: Game, Genre, Platform, ValuePartition
- hasDifficulty

Description of the class "Game"

Equivalent To: +

SubClass Of: +

- owl:Thing

General class axioms: +

SubClass Of (Anonymous Ancestor):

Instances: +

Target for Key: +

Disjoint With: +

- ValuePartition, Genre, Platform

Disjoint Union Of: +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Class Hierarchies

It allows us to speed up the process of adding classes.

Tools → Create class hierarchy...

The screenshot shows the Protégé ontology editor interface. The left pane displays a class hierarchy starting from `owl:Thing`, with `Game` highlighted by a red arrow. The right pane shows the 'Usage: Game' tab, indicating that `Game` is a subclass of `owl:Thing` and is disjoint with `Genre`, `Platform`, and `ValuePartition`. The bottom pane shows the 'Description: Game' tab, which is highlighted with a red box. In this tab, the 'SubClass Of' section shows `owl:Thing` as the parent class. To the right of the description pane, the text $Game \sqsubseteq T$ is displayed in red.

untitled-ontology-25 (<http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25>) : [<http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25>]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (<http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25>) Search...

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLviz x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Game Usage: Game

Show: ☒ this ☒ disjoint ☒ named sub/superclasses

Found 10 uses of Game

- Game
 - Game SubClassOf owl:Thing
 - Class: Game
 - DisjointClasses: Game, Genre, Platform, ValuePartition
- Genre
 - DisjointClasses: Game, Genre, Platform, ValuePartition
- hasDifficulty

Description: Game

Equivalent To +

SubClass Of +
owl:Thing

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +
ValuePartition, Genre, Platform

Disjoint Union Of +

$Game \sqsubseteq T$

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Class Hierarchies

It allows us to speed up the process of adding classes.

Tools → Create class hierarchy...

The screenshot shows the Protégé ontology editor interface. On the left, the 'Class hierarchy' panel displays a tree structure starting from 'owl:Thing'. A red arrow points to the 'Game' class, which is highlighted. Below 'Game', several subclasses are listed: 'NamedGame', 'Chess', 'LeagueOfLegends', 'Sudoku', 'WorldOfWarcraft', 'Platform', 'LinuxPlatform', 'MacOSXPlatform', 'WindowsPlatform', 'Genre', 'MultiPlayer', 'Online', 'Puzzle', 'RolePlayGame', 'SinglePlayer', 'ValuePartition', 'DifficultyValuePartition', 'Easy', 'Hard', and 'Normal'. The 'Genre' class is also highlighted. On the right, the 'Usage: Game' panel shows 'Found 10 uses of Game'. Below this, the 'Description: Game' panel is visible, showing various properties and axioms. A red box highlights the 'Disjoint With' section, which lists 'ValuePartition', 'Genre', and 'Platform'. To the right of the screenshot, three logical expressions are written in red:

$$Game \sqcap ValuePartition \sqsubseteq \perp$$
$$Game \sqcap Genre \sqsubseteq \perp$$
$$Game \sqcap Platform \sqsubseteq \perp$$

At the bottom of the screenshot, a footer bar contains the text: 'To use the reasoner click Reasoner > Start reasoner' and a checked checkbox for 'Show Inferences'.

Adding Class Hierarchies

It allows us to speed up the process of adding classes.

Tools → Create class hierarchy...

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main toolbar contains icons for creating, deleting, and editing classes. The left sidebar shows the ontology structure, with 'owl:Thing' at the root. A red arrow points to the 'Game' class under 'owl:Thing'. The right sidebar shows the 'Class hierarchy' and 'Usage' tabs. The 'Class hierarchy' tab displays the hierarchy for 'Game', including 'NamedGame', 'Chess', 'LeagueOfLegends', 'Sudoku', 'WorldOfWarcraft', 'Platform', 'LinuxPlatform', 'MacOSXPlatform', 'WindowsPlatform', 'Genre', 'MultiPlayer', 'Online', 'Puzzle', 'RolePlayGame', 'SinglePlayer', 'ValuePartition', 'DifficultyValuePartition', 'Easy', 'Hard', and 'Normal'. The 'Usage' tab shows the 'Game' class is used in 10 instances, including 'SubClassOf owl:Thing', 'Class: Game', and 'DisjointClasses: Game, Genre, Platform, ValuePartition'. The bottom panel shows the 'Description: Game' tab, which lists various axioms for the 'Game' class. A red box highlights the 'Delete axiom' and 'Edit axiom' buttons in the bottom right corner of the description panel.

untitled-ontology-25 (http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25) : [http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (http://www.semanticweb.org/papacch/ontologies/2016/9/untitled-ontology-25)

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLviz x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Game Usage: Game

Show: ☒ this ☒ disjoint ☒ named sub/superclasses

Found 10 uses of Game

Game

- Game SubClassOf owl:Thing
- Class: Game
- DisjointClasses: Game, Genre, Platform, ValuePartition

Genre

- DisjointClasses: Game, Genre, Platform, ValuePartition

hasDifficulty

Description: Game

Equivalent To +

SubClass Of +

- owl:Thing

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

- ValuePartition, Genre, Platform

Disjoint Union Of +

Delete axiom

Edit axiom

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

What Now?

What we have...

- ▶ all non-definable classes
- ▶ an initial class hierarchy
- ▶ basic (among siblings) disjoint axioms

What we need to add...

- ▶ object properties
- ▶ relations between classes
- ▶ definable classes

Object Properties (Domain and Range)

Make sure to have the “Object Properties” tab open
Window → Tabs → Object Properties

The screenshot shows the Semantic Web editor interface. The 'Object Properties' tab is active, displaying the 'hasPlatform' property. The left pane shows the 'Object property hierarchy: hasPlatform' with a tree structure: owl:topObjectProperty, hasPlatform, hasDifficulty, and hasGenre. The right pane shows the 'Usage: hasPlatform' section, indicating 'Found 1 uses of hasPlatform' and listing 'ObjectProperty: hasPlatform'. A red box highlights the 'Characteristic' and 'Description' sections. The 'Characteristic' section on the left lists various property types (Functional, Inverse function, Transitive, Symmetric, Asymmetric, Reflexive, Irreflexive) with checkboxes. The 'Description: hasPlatform' section on the right shows various logical constraints (Equivalent To, SubProperty Of, Inverse Of, Domains (intersection), Ranges (intersection), Disjoint With, SuperProperty Of (Chain)) with plus icons. A red arrow points to the 'Domains (intersection)' plus icon, which is currently set to 'Game'. Below the red box, the logical expression $\exists \text{hasPlatform.T} \sqsubseteq \text{Game}$ is written in red.

untitled-ontology-25 (http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25) : [http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25)

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLViz x DL Query x

Object property hierarchy: hasPlatform

Annotations Usage

Usage: hasPlatform

Show: ☒ this ☒ disjoints

Found 1 uses of hasPlatform

hasPlatform

ObjectProperty: hasPlatform

Characteristic

Description: hasPlatform

☐ Functional

☐ Inverse function

☐ Transitive

☐ Symmetric

☐ Asymmetric

☐ Reflexive

☐ Irreflexive

Equivalent To +

SubProperty Of +

Inverse Of +

Domains (intersection) +

Game

Ranges (intersection) +

Disjoint With +

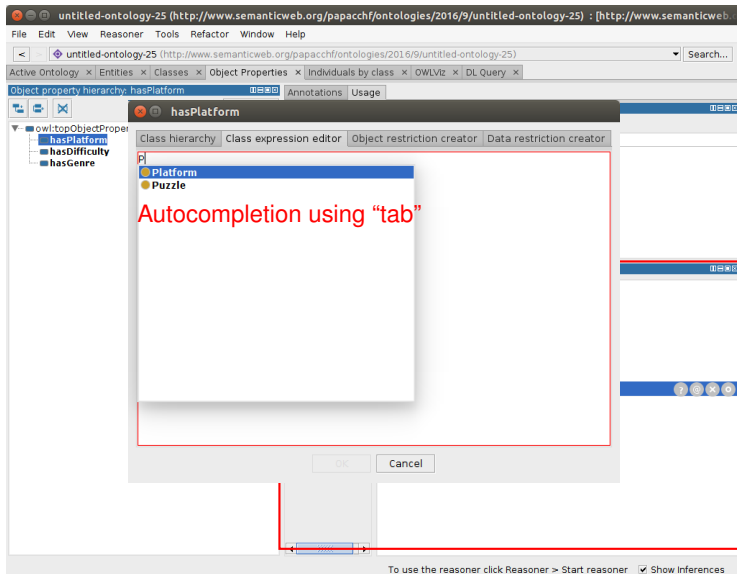
SuperProperty Of (Chain) +

$\exists \text{hasPlatform.T} \sqsubseteq \text{Game}$

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Object Properties (Domain and Range)

Make sure to have the “Object Properties” tab open
Window → Tabs → Object Properties



Object Properties (Domain and Range)

Make sure to have the “Object Properties” tab open
Window → Tabs → Object Properties

The screenshot shows the Semantic Web editor interface. The top bar displays the URL 'untitled-ontology-25'. The main menu includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The 'Object Properties' tab is active, showing the 'hasPlatform' property hierarchy. The 'Usage' tab is selected, displaying 'Found 1 uses of hasPlatform' and a list of 'ObjectProperty: hasPlatform'. A red box highlights the 'Characteristic' and 'Description' panels. The 'Characteristic' panel lists various property attributes, and the 'Description' panel shows the domain 'Game' and range 'Platform'. The logical expression
$$T \sqsubseteq \forall \text{hasPlatform}. \text{Platform}$$
 is written in red at the bottom of the red box.

untitled-ontology-25 (http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25) : [http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25)

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLViz x DL Query x

Object property hierarchy: hasPlatform

Annotations Usage

Usage: hasPlatform

Show: ☒ this ☒ disjoints

Found 1 uses of hasPlatform

hasPlatform

ObjectProperty: hasPlatform

owl:topObjectProperty

hasPlatform

hasDifficulty

hasGenre

Characteristic

Functional

Inverse function

Transitive

Symmetric

Asymmetric

Reflexive

Irreflexive

Description: hasPlatform

Equivalent To

SubProperty Of

Inverse Of

Domains (intersection)

Game

Ranges (intersection)

Platform

Disjoint With

SuperProperty Of (Chain)

$$T \sqsubseteq \forall \text{hasPlatform}. \text{Platform}$$

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Axioms

Which axioms?

- ▶ only axioms of the following forms
 - ▶ $A \sqsubseteq C$ (necessary condition for A)
 - ▶ $A \equiv C$ (sufficient and necessary condition for A – definition)
- ▶ for each subclass of NamedGame we need to insert axioms expressing something like
 - ▶ Chess can be installed on any platform
 - ▶ League of Legends is an online game
- ▶ DifficultyValuePartition need to be properly defined
(i.e., its values can only be Hard, Normal, or Easy)
- ▶ adding definable classes

$A \sqsubseteq C$ – Example

- ▶ Natural language specification

Chess can be installed on any platform

- ▶ Rephrase the specification using the ontology vocabulary

Chess has platform Windows, has platform MacOSX, and has platform Linux

- ▶ Write it in description logic syntax (optional)

$Chess \sqsubseteq \exists hasPlatform.WindowsPlatform$

$Chess \sqsubseteq \exists hasPlatform.MacOSXPlatform$

$Chess \sqsubseteq \exists hasPlatform.LinuxPlatform$

- ▶ Write it in Manchester syntax (the right-hand side is enough)

hasPlatform some WindowsPlatform

hasPlatform some MacOSXPlatform

hasPlatform some LinuxPlatform

Adding Axioms to the Class “Chess”

The screenshot shows the Semantic Web editor interface for an ontology named "untitled-ontology-25". The left pane displays the "Class hierarchy: Chess" with a tree structure. The "Chess" class is highlighted with a red arrow. The right pane shows the "Usage: Chess" section, which lists 6 uses of the class. Below this, the "Description: Chess" section is highlighted with a red box. It contains several axioms, including "SubClass Of" and "Disjoint With", both of which are highlighted with red arrows. The "SubClass Of" axiom shows "Chess" as a subclass of "NamedGame". The "Disjoint With" axiom shows "Chess" is disjoint with "Sudoku", "WorldOfWarcraft", and "LeagueOfLegends".

untitled-ontology-25 (<http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>) : (<http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>)

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (<http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>) Search...

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLviz x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Chess Usage: Chess

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 6 uses of Chess

- Class: Chess
- Chess SubClassOf NamedGame
- DisjointClasses: Chess, LeagueOfLegends, Sudoku, WorldOfWarcraft

- LeagueOfLegends
 - DisjointClasses: Chess, LeagueOfLegends, Sudoku, WorldOfWarcraft
- Sudoku

Description: Chess

Equivalent To +

SubClass Of +

- NamedGame

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

- Sudoku, WorldOfWarcraft, LeagueOfLegends

Disjoint Union Of +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Axioms to the Class “Chess”

The screenshot shows the Protégé interface for an ontology named 'untitled-ontology-25'. The 'Class hierarchy' tab is active, displaying a tree structure of classes. The 'Chess' class is highlighted under the 'NamedGame' class. A red arrow points to the 'Chess' class in the hierarchy.

The 'Class expression editor' dialog is open, showing the 'Class hierarchy' tab. The text 'hasPlatform some WindowsPlatform' is entered in the editor. The dialog has 'OK' and 'Cancel' buttons at the bottom.

Below the dialog, a red box highlights a section of the interface. It shows a list of classes: 'Sudoku, WorldOfWarcraft, LeagueOfLegends'. Below this list is a 'Disjoint Union Of' button with a plus sign icon.

At the bottom of the interface, there is a status bar with the text: 'To use the reasoner click Reasoner > Start reasoner' and a checked checkbox for 'Show Inferences'.

Adding Axioms to the Class “Chess”

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the URL: <http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>. The main window is divided into several panes:

- Class hierarchy:** Shows a tree structure of classes. The 'Chess' class is highlighted under the 'NamedGame' class. A red arrow points to the 'Chess' class in the hierarchy.
- Usage:** Shows the usage of the 'Chess' class. It lists 6 uses of 'Chess' and shows the class hierarchy for 'Chess' (Class: Chess, SubClassOf: NamedGame, DisjointClasses: Chess, LeagueOfLegends, Sudoku, WorldOfWarcraft).
- Description:** Shows the description of the 'Chess' class. It includes the following axioms:
 - Equivalent To: (empty list)
 - SubClass Of:
 - hasDifficulty some Normal
 - hasGenre some MultiPlayer
 - hasGenre some SinglePlayer
 - hasPlatform some LinuxPlatform
 - hasPlatform some MacOSXPlatform
 - hasPlatform some WindowsPlatform
 - NamedGame
 - General class axioms: (empty list)
 - SubClass Of (Anonymous Ancestor): (empty list)
 - Instances: (empty list)
 - Target for Key: (empty list)

At the bottom of the window, there is a status bar that says: "To use the reasoner click Reasoner > Start reasoner" and a checkbox for "Show Inferences" which is checked.

Improving DifficultyValuePartition Definition

What needs to be done?

- ▶ add $\text{DifficultyValuePartition} \equiv \text{Hard} \sqcup \text{Normal} \sqcup \text{Easy}$

Note that Hard, Normal and Easy are already disjoint

- ▶ add domain and range of hasDifficulty
- ▶ make hasDifficulty functional

Improving DifficultyValuePartition Definition (cont'd)

untitled-ontology-25 (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>) : (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>)

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>) Search...

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLViz x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: DifficultyValuePartition Usage: DifficultyValuePartition

owl:Thing

- Game
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSXPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition
 - Easy
 - Hard
 - Normal

Found 6 uses of DifficultyValuePartition

- DifficultyValuePartition
 - DifficultyValuePartition SubClassOf ValuePartition
 - Class: DifficultyValuePartition
- Easy
 - Easy SubClassOf DifficultyValuePartition
- Hard
 - Hard SubClassOf DifficultyValuePartition

Description: DifficultyValuePartition

Equivalent To +

SubClass Of +

- ValuePartition

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

Disjoint Union Of +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Improving DifficultyValuePartition Definition (cont'd)

The screenshot displays the Protégé ontology editor interface. The main window shows the 'Class hierarchy' view for the 'DifficultyValuePartition' class. The hierarchy is as follows:

- owl:Thing
 - Game
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayingGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition
 - Easy
 - Hard
 - Normal

A red arrow points to the 'DifficultyValuePartition' class in the hierarchy. A modal dialog titled 'DifficultyValuePartition' is open, showing the 'Object restriction creator' tab. The text input field contains 'Hard or Normal or Easy'. Below the dialog, a red box highlights the 'Disjoint Union Of' button, which is used to define the 'DifficultyValuePartition' class as a disjoint union of its subclasses.

At the bottom of the window, the status bar indicates: 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences' is checked.

Improving DifficultyValuePartition Definition (cont'd)

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window shows the 'untitled-ontology-25' with a search bar and tabs for Active Ontology, Entities, Classes, Object Properties, Individuals by class, OWL Viz, and DL Query. The 'Classes' tab is active, showing a class hierarchy on the left and a description panel on the right.

Class hierarchy: DifficultyValuePartition

- owl:Thing
 - Game
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSXPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition** (highlighted with a red arrow)
 - Easy
 - Hard
 - Normal

Usage: DifficultyValuePartition

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 7 uses of DifficultyValuePartition

- DifficultyValuePartition**
 - DifficultyValuePartition **SubClassOf** ValuePartition
 - DifficultyValuePartition **EquivalentTo** Easy or Hard or Normal
 - Class: DifficultyValuePartition
- Easy**
 - Easy **SubClassOf** DifficultyValuePartition
- Hard**

Description: DifficultyValuePartition

Equivalent To: **Hard or Normal or Easy**

SubClass Of: **ValuePartition**

General class axioms: +

SubClass Of (Anonymous Ancestor):

Instances: +

Target for Key: +

Disjoint With: +

Disjoint Union Of: +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Improving DifficultyValuePartition Definition (cont'd)

The screenshot displays the Protégé ontology editor interface for 'untitled-ontology-25'. The left sidebar shows the 'Object property hierarchy: hasDifficulty' with a red arrow pointing to the 'hasDifficulty' property. The main panel shows the 'Usage: hasDifficulty' view, indicating it is an asserted property with 4 uses. The 'Characteristics' tab is active, showing the 'Functional' checkbox checked. The 'Description' tab shows the property's configuration: 'Equivalent To' is empty, 'SubProperty Of' is empty, 'Inverse Of' is empty, 'Domains (intersection)' includes 'Game', and 'Ranges (intersection)' includes 'DifficultyValuePartition'. Red arrows point to the 'Game' and 'DifficultyValuePartition' entries in the domains and ranges lists. The bottom status bar indicates 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences' is checked.

untitled-ontology-25 (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>) : [<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-25 (<http://www.semanticweb.org/papacchf/ontologies/2016/9/untitled-ontology-25>) Search...

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLviz x DL Query x

Object property hierarchy: hasDifficulty

Annotations Usage

Asserted Usage: hasDifficulty

Show: ☒ this ☒ disjoints

Found 4 uses of hasDifficulty

- Chess
 - Chess SubClassOf hasDifficulty some Normal
- hasDifficulty
 - hasDifficulty Range DifficultyValuePartition
 - ObjectProperty: hasDifficulty
 - hasDifficulty Domain Game

Characteristics Description: hasDifficulty

☒ Functional

☐ Inverse function

☐ Transitive

☐ Symmetric

☐ Asymmetric

☐ Reflexive

☐ Irreflexive

Equivalent To +

SubProperty Of +

Inverse Of +

Domains (intersection) +

- Game

Ranges (intersection) +

- DifficultyValuePartition

Disjoint With +

SuperProperty Of (Chain) +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Definable Class “MultiPlayerGame”

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window displays the ontology "untitled-ontology-25" with a search bar and tabs for Active Ontology, Entities, Classes, Object Properties, Individuals by class, OWLviz, and DL Query.

The left pane shows the "Class hierarchy: MultiPlayerGame" with a tree structure. A red arrow points to the "MultiPlayerGame" class under the "Game" class. The tree structure is as follows:

- owl:Thing
 - Game
 - MultiPlayerGame**
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSXPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition
 - Easy
 - Hard
 - Normal

The right pane shows the "Usage: MultiPlayerGame" tab, which displays "Found 2 uses of MultiPlayerGame". The first use is "MultiPlayerGame" (Class: MultiPlayerGame) and the second use is "MultiPlayerGame SubClassOf Game".

The bottom pane shows the "Description: MultiPlayerGame" tab, which contains the following information:

- Equivalent To: +
- SubClass Of: +
 - Game
 - hasGenre some MultiPlayer
- General class axioms: +
- SubClass Of (Anonymous Ancestor):
- Instances: +
- Target for Key: +
- Disjoint With: +
- Disjoint Union Of: +

A red arrow points to the "hasGenre some MultiPlayer" axiom in the "SubClass Of" list.

At the bottom of the window, a status bar indicates: "To use the reasoner click Reasoner > Start reasoner" and a checkbox for "Show Inferences" is checked.

Adding Definable Class “MultiPlayerGame”

The screenshot shows the Protégé ontology editor interface. The title bar indicates the current ontology is 'untitled-ontology-25' with the URL <http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>. The 'Edit' menu is open, displaying various actions such as Undo, Redo, Cut, Copy, Paste, Delete, and Find. The 'Convert to defined class' option is highlighted in blue, with a mouse cursor pointing at it. Other visible options include 'Convert to primitive class', 'Add covering axiom', 'Make primitive siblings disjoint', 'Remove disjoint classes axioms for subclasses...', 'Make all individuals distinct...', and 'Remove all disjoint classes axioms...'. The background shows a class hierarchy with 'MultiPlayerGame' as a subclass of 'Game'. The bottom status bar indicates 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences' is checked.

File Edit View Reasoner Tools Refactor Window Help

Undo Ctrl-Z
Redo Ctrl+Shift-Z
Cut Ctrl-X
Copy Ctrl-C
Copy sub-hierarchy as tab indented text Ctrl+Alt-C
Paste Ctrl+V
Delete... Ctrl-Delete
Duplicate selected class... Ctrl+Shift-C
Find... Ctrl-F
Find in view... Ctrl+Shift-F
Create new Ctrl+Alt+Shift-E
Create child Ctrl-E
Create sibling Ctrl+Shift-E
Convert to primitive class Ctrl-P
Convert to defined class Ctrl-D
Add covering axiom
Make primitive siblings disjoint Ctrl-J
Remove disjoint classes axioms for subclasses...
Make all individuals distinct...
Remove all disjoint classes axioms...

Normal

SubClass Of (Anonymous Ancestor)

Instances +
Target for Key +
Disjoint With +
Disjoint Union Of +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Adding Definable Class “MultiPlayerGame”

The screenshot shows the Protégé ontology editor interface. The left pane displays the class hierarchy, with **MultiPlayerGame** highlighted under the **Game** class. A red arrow points to this class in the hierarchy. The right pane shows the **Usage: MultiPlayerGame** tab, which displays the class definition: **MultiPlayerGame** *EquivalentTo* **Game** *and* (*hasGenre* *some* **MultiPlayer**). A red arrow points to this definition. Below the definition, the **Description: MultiPlayerGame** tab is visible, showing the same definition. A red box highlights this tab, and a red arrow points to the definition. The bottom status bar indicates: "To use the reasoner click Reasoner > Start reasoner" and "Show Inferences" is checked.

untitled-ontology-25 (<http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>) : <http://www.semanticweb.org/papachf/ontologies/2016/9/untitled-ontology-25>

File Edit View Reasoner Tools Refactor Window Help

Active Ontology x Entities x Classes x Object Properties x Individuals by class x OWLviz x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: MultiPlayerGame Usage: MultiPlayerGame

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 2 uses of MultiPlayerGame

MultiPlayerGame

- Class: MultiPlayerGame
- MultiPlayerGame *EquivalentTo* Game *and* (*hasGenre* *some* MultiPlayer)

MultiPlayerGame \equiv *Game* \sqcap \exists *hasGenre*. *MultiPlayer*

Description: MultiPlayerGame

Equivalent To: **Game** *and* (*hasGenre* *some* **MultiPlayer**)

SubClass Of

General class axioms

SubClass Of (Anonymous Ancestor)

Instances

Target for Key

Disjoint With

Disjoint Union Of

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Reasoning

Protégé can be used for reasoning tasks such as classification

- ▶ configure the reasoner

Reasoner → Configure... (for this tutorial, check everything under Class inferences and Object property inferences)

- ▶ select a reasoner

for example, Reasoner → HermiT (other reasoners can be added, which one to use depends on several factors such as the expressivity of the ontology)

- ▶ finally, Reasoner → Start reasoner

Reasoning Example

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window is titled 'untitled-ontology-25' and shows a class hierarchy on the left and a description of the 'Game' class on the right.

Class hierarchy: Game

- owl:Thing
 - Game
 - LinuxGame
 - MacOSXGame
 - WindowsGame
 - NormalGame
 - EasyGame
 - HardGame
 - MultiPlatformGame
 - OnlineGame
 - SinglePlayerGame
 - MultiPlayerGame
 - NamedGame
 - Chess
 - LeagueOfLegends
 - Sudoku
 - WorldOfWarcraft
 - Platform
 - LinuxPlatform
 - MacOSXPlatform
 - WindowsPlatform
 - Genre
 - MultiPlayer
 - Online
 - Puzzle
 - RolePlayGame
 - SinglePlayer
 - ValuePartition
 - DifficultyValuePartition
 - Easy
 - Hard
 - Normal

Usage: Game

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 20 uses of Game

- EasyGame
 - EasyGame EquivalentTo Game and (hasDifficulty some Easy)
- Game
 - Game SubClassOf owl:Thing
 - Class: Game
 - DisjointClasses: Game, Genre, Platform, ValuePartition

Description: Game

Equivalent To: +

SubClass Of: +

- owl:Thing

General class axioms: +

SubClass Of (Anonymous Ancestor):

Instances: +

Target for Key: +

Disjoint With: +

- ValuePartition, Genre, Platform

Disjoint Union Of: +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

Reasoning Example

The screenshot displays a Semantic Web browser interface for an ontology named 'untitled-ontology-25'. The left pane shows a class hierarchy for 'Chess', with 'Chess' highlighted under 'MultiPlatformGame'. A red box highlights the 'Genre' class at the bottom of the hierarchy. The right pane shows the 'Usage: Chess' section, which lists 12 uses of 'Chess'. A red arrow points from the 'Inferred' tab in the left pane to the 'Usage: Chess' section. The 'Usage: Chess' section lists the following uses:

- Chess SubClassOf hasDifficulty **some** Normal
- Chess SubClassOf hasGenre **some** MultiPlayer
- Class: Chess
- Chess SubClassOf hasGenre **some** SinglePlayer

The 'Description: Chess' section shows the following properties:

- hasDifficulty **some** Normal
- hasGenre **some** MultiPlayer
- hasGenre **some** SinglePlayer
- hasPlatform **some** LinuxPlatform
- hasPlatform **some** MacOSXPlatform
- hasPlatform **some** WindowsPlatform
- NamedGame
- MultiPlatformGame
- MultiPlayerGame
- NormalGame
- SinglePlayerGame

The 'General class axioms' section shows the following axioms:

- SubClass Of (Anonymous Ancestor)
- Game and (hasGenre **some** MultiPlayer)
- Game and (hasDifficulty **some** Normal)
- Game and (hasPlatform **some** LinuxPlatform) and (hasPlatform **some** MacOSXPlatform) and (hasPlatform **some** WindowsPlatform)
- Game and (hasGenre **some** SinglePlayer)

At the bottom right, the status bar indicates 'Reasoner active' and 'Show Inferences' is checked.

Reasoning – Visually (Asserted)



Reasoning – Visually (Inferred)

