FootOntologyPlus

Giacomo Leo Bertuccioli
0001136879

Alex Mazzotti
matricola

Indice

Capitolo 1

Introduzione

Footology [1] è un knowledge representation system progettato per rappresentare e organizzare le informazioni rilevanti del mondo del calcio. L'ontologia consente di modellare i diversi aspetti e concetti del gioco, come ad esempio partite, squadre di calcio e giocatori. L'idea dietro al progetto è quella di facilitare l'integrazione dei dati, l'interoperabilità e fornire sistemi di interrogazione avanzati per applicazioni riguardanti il calcio.

La nostra estensione FootOntologyPlus mira ad espandere l'originale, modellando meglio alcuni dei concetti inclusi da quest'ultima o aggiungendone di nuovi. L'entità delle nostre modifiche è discussa nei capitoli successivi.

Il nostro progetto è stato realizzato usando l'editor **Protegé**, un programma open source per visualizzare e modificare ontologie.

Capitolo 2

Ontologia di partenza

Capitolo 3

Modifiche all'ontologia

Capitolo 4

Inserimento di individui

Capitolo 5

Regole SWRL

SWRL (**Semantic Web Rule Language**) è un linguaggio usato nell'ambito del Semantic Web per esprimere regole di inferenza e vincoli logici più complessi di quelli realizzabili usando il solo OWL.

Nella nostra ontologia abbiamo inserito numerose regole, utili a:

1. inferire nuovi collegamenti tra i dati
2. individuare casi di errore sui dati

Le sezioni che seguono riportano tali regole, non in un ordine particolare.

5.1 HomeFormationRule

Questa regola "raffina" il collegamento tra una formazione e una partita, esplicitando che la prima è usata dalla squadra in casa. Questo perché `isAwayFormationIn` è una sottoproprietà di `isFormationIn`.

```
isHomeTeam(?t, ?m) ^ hasFormation(?t, ?f) ^ isFormationIn(?f, ?m) ->
    isHomeFormationIn(?f, ?m)
```

5.2 AwayFormationRule

Questa regola funziona in modo analogo alla precedente, ma riguardo la squadra in trasferta.

```
isAwayTeam(?t, ?m) ^ hasFormation(?t, ?f) ^ isFormationIn(?f, ?m) ->
    isAwayFormationIn(?f, ?m)
```

5.3 DuplicateSubstituteRule

Questa regola verifica se un giocatore segnato come riserva partecipa a multiple sostituzioni come giocatore entrante, e in tale caso gli assegna una classe di errore.

```
hasReservePlayer(?f, ?p) ^ hasEnteringPlayer(?s1, ?p) ^
    hasEnteringPlayer(?s2, ?p) ^ differentFrom(?s1, ?s2) ^
    hasTeamFormation(?m, ?f) ^ hasSubstitution(?m, ?s1) ^
    hasSubstitution(?m, ?s2) -> DuplicateSubstituteError(?p)
```

5.4 DuplicateSubstitutedRule

Il funzionamento di questa regola è analogo alla precedente, ma riguardante il giocatore uscente. Questa regola copre però solo i giocatori titolari (ovvero i giocatori che partecipano alla partita dall'inizio).

```
hasStarterPlayer(?f, ?p) ^ hasExitingPlayer(?s1, ?p) ^
    hasExitingPlayer(?s2, ?p) ^ differentFrom(?s1, ?s2) ^
    hasTeamFormation(?m, ?f) ^ hasSubstitution(?m, ?s1) ^
    hasSubstitution(?m, ?s2) -> DuplicateSubstitutedError(?p)
```

5.5 StarterAsSubstituteRule

Questa regola verifica se un giocatore titolare è il giocatore entrante in una sostituzione, e in tale caso gli assegna una classe di errore.

```
hasStarterPlayer(?f, ?p) ^ hasEnteringPlayer(?s, ?p) ^
    hasTeamFormation(?m, ?t) ^ hasSubstitution(?m, ?s) ->
    StarterAsSubstituteError(?p)
```

5.6 FriendlyMatchInTournamentRule

Non è corretto assegnare una partita amichevole ad un torneo, qualsiasi sia il suo tipo, e se ciò viene fatto allora una classe di errore è assegnata alla partita.

```
includedInTournament(?m, ?t) ^ FriendlyMatch(?m) ->
    FriendlyMatchInTournamentError(?m)
```

5.7 GoalScoredByTeamRule

Questa regola collega un goal ad una squadra, sapendo che il goal è stato fatto da un giocatore che giocava in tale squadra nella partita.

```
scores(?p, ?g) ^ hasEvent(?m, ?g) ^ isPlayerInFormation(?p, ?f) ^
    hasFormation(?t, ?f) ^ competesIn(?t, ?m) -> scoredByTeam(?g, ?t)
```

5.8 InconsistentBallPossessionInMatchRule

Questa regola controlla se la somma del possesso palla indicata nelle statistiche di performance per le squadre che hanno giocato una partita non è 100%, e in tale caso assegna una classe di errore alla partita e alle statistiche.

```
teamStatsIn(?ps1, ?m) ^ teamStatsIn(?ps2, ?m) ^ differentFrom(?ps1,
    ?ps2) ^ BallPossession(?ps1, ?p1) ^ BallPossession(?ps2, ?p2) ^
    swrlb:add(?r, ?p1, ?p2) ^ swrlb:notEqual(?r, 100) ->
    InconsistentBallPossessionInMatchError(?m) ^
    InconsistentBallPossessionInMatchError(?ps1) ^
    InconsistentBallPossessionInMatchError(?ps2)
```

5.9 InconsistentContractDatesRule

Questa regola verifica se le date di un contratto sono inconsistenti (data di inizio futura alla data di fine).

```
ContractStartDate(?c, ?sd) ^ ContractEndDate(?c, ?ed) ^
    temporal:before(?ed, ?sd) -> InconsistentContractDatesError(?c)
```

5.10 InconsistentMatchScoresHome/AwayRule

Queste due regole verificano se il vincitore di una partita non è consistente con i punteggi assegnati alle due squadre. Sono state necessarie due regole in quanto i punteggi delle squadre sono legati alla partita e non direttamente alle squadre.

```
isHomeTeam(?t1, ?m) ^ isAwayTeam(?t2, ?m) ^ hasWinner(?m, ?t1) ^
    MatchHomeTeamScore(?m, ?s1) ^ MatchAwayTeamScore(?m, ?s2) ^
    swrlb:lessThanOrEqual(?s1, ?s2) -> InconsistentMatchScoresError(?m)
```

```
isHomeTeam(?t1, ?m) ^ isAwayTeam(?t2, ?m) ^ hasWinner(?m, ?t2) ^
    MatchHomeTeamScore(?m, ?s1) ^ MatchAwayTeamScore(?m, ?s2) ^
    swrlb:lessThanOrEqual(?s2, ?s1) -> InconsistentMatchScoresError(?m)
```

5.11 InvalidLeagueMatchFormatExtraTime/PenaltyRule

Le partite che fanno parte di una lega non possono avere supplementari o rigori, e queste due regole servono a identificare tali inconsistenze.

```
includedInTournament(?m, ?t) ^ League(?t) ^ ExtraTimePlayed(?m, true) ->
    InvalidLeagueMatchFormatError(?m)
```

```
includedInTournament(?m, ?t) ^ League(?t) ^ PenaltyShootoutPlayed(?m,
    true) -> InvalidLeagueMatchFormatError(?m)
```

5.12 InvalidMatchRule

Questa regola verifica se una squadra sta giocando contro sé stessa in una partita.

```
hasHomeTeam(?m, ?t) ^ hasAwayTeam(?m, ?t) -> InvalidMatchError(?m)
```

5.13 MultipleRedCardsRule

Questa regola controlla se un giocatore ha ricevuto multipli cartellini rossi in una partita.

```
hasReceived(?p, ?c1) ^ hasReceived(?p, ?c2) ^ RedCard(?c1) ^
    RedCard(?c2) ^ differentFrom(?c1, ?c2) ^ hasEvent(?m, ?c1) ^
    hasEvent(?m, ?c2) -> MultipleRedCardsError(?p)
```

5.14 NonGoalKeeperWithSavesRule

Questa regola verifica se le statistiche di un giocatore indicano il numero di parate sebbene esso non abbia giocato come portiere nella partita.

```
playsInPosition(?pl, ?p) ^ Outfield(?p) ^ hasPlayerStats(?pl, ?ps) ^
    Saves(?ps, ?s) ^ hasPlayer(?m, ?p) ^ hasPlayerPerformanceStats(?m,
    ?ps) -> NonGoalKeeperWithSavesError(?pl)
```

5.15 PlayerInMatchRule

Questa regola collega un giocatore ad una partita, sapendo che il giocatore era in una formazione utilizzata in tale partita.

```
hasPlayerInFormation(?f, ?p) ^ isFormationIn(?f, ?m) -> hasPlayer(?m, ?p)
```

5.16 TeamNotInMatchRule

Questa regola controlla se un evento in una partita è legato ad una squadra che non gioca in tale partita.

```
hasEvent(?m, ?e) ^ hasMatchEvent(?t, ?e) ^ isHomeTeam(?t1, ?m) ^
    isAwayTeam(?t2, ?m) ^ differentFrom(?t, ?t1) ^ differentFrom(?t, ?t2)
    -> TeamNotInMatchError(?t)
```

5.17 UncontractedPlayerInMatchRule

Questa regola verifica se un giocatore ha partecipato ad una partita senza far parte di una delle due squadre coinvolte, in quanto non si trovava sotto contratto con una delle due squadre al momento della partita. La regola copre soltanto i casi in cui il giocatore si trovava sotto contratto con un'altra squadra e se il contratto è terminato.

```
signs(?t1, ?c) ^ involvesPlayer(?c, ?p) ^ participatesIn(?p, ?m) ^
    includes(?m, ?t2) ^ includes(?m, ?t3) ^ differentFrom(?t1, ?t2) ^
    differentFrom(?t1, ?t3) ^ ContractStartDate(?c, ?sd) ^
    ContractEndDate(?c, ?ed) ^ MatchDate(?m, ?d) ^ temporal:before(?sd,
    ?d) ^ temporal:after(?ed, ?d) -> UncontractedPlayerInMatchError(?p)
```

Capitolo 6

Interrogazioni SPARQL

L'ultima parte del progetto è stata la realizzazione di interrogazioni usando **SPARQL** (**SPARQL Protocol and RDF Query Language**), un linguaggio simile a SQL per poter estrarre dati dall'ontologia.

Le interrogazioni che abbiamo scritto sono esempi che producono risultati sulla base degli individui d'esempio che abbiamo predisposto. Per realizzarle abbiamo usato le viste **SPARQL Query** e **Snap SPARQL Query** di Protegé.

6.1 Interrogazione sui contratti di un giocatore

Questa interrogazione recupera i contratti firmati da un giocatore (nell'esempio `Player1`) e i nomi dei team per cui i contratti sono stati firmati.

```
PREFIX : <http://visualdataweb.org/FootOntologyPlus/>

SELECT ?contract ?teamName
WHERE {
    ?team :signsPlayer ?contract ;
          :TeamName ?teamName .
    ?contract :involvesPlayer :Player1 .
}
```

?contract	?teamName
:Contract1	Milan^^xsd:string
:Contract61	Milan^^xsd:string

Figura 6.1: Risultato della query sulla nostra ontologia.

6.2 Interrogazione sui contratti non terminati

Questa interrogazione recupera tutti i contratti che non hanno una data di terminazione. Si assume in questo caso che tali contratti siano ancora attivi, ma in generale vige la "Open World assumption" riguardo alle informazioni mancanti.

```
PREFIX : <http://visualdataweb.org/FootOntologyPlus/>

SELECT ?player ?team ?contract ?startDate
WHERE {
    ?contract :isSignedForPlayerBy ?team ;
        :involvesPlayer ?player ;
        :ContractStartDate ?startDate .
    FILTER NOT EXISTS { ?contract :ContractEndDate ?endDate }
}
ORDER BY ASC (?player)
```

player	team	contract	startDate
Player1	Team1	Contract1	"2018-06-05T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player10	Team1	Contract10	"2014-11-28T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player11	Team1	Contract11	"2023-10-28T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player12	Team2	Contract12	"2022-01-01T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player13	Team2	Contract13	"2015-03-22T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player14	Team2	Contract14	"2015-04-25T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player15	Team2	Contract15	"2022-11-26T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player16	Team2	Contract16	"2015-06-28T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player17	Team2	Contract17	"2016-05-30T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player18	Team2	Contract18	"2019-05-15T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player19	Team2	Contract19	"2014-11-17T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player2	Team1	Contract2	"2014-12-30T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player20	Team2	Contract20	"2015-11-25T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player21	Team2	Contract21	"2015-03-03T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player22	Team2	Contract22	"2017-04-15T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Player23	Team3	Contract23	"2017-08-22T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>

Figura 6.2: Parte del risultato della query sulla nostra ontologia.

6.3 Interrogazione sui cartellini ricevuti dai giocatori

Questa interrogazione recupera le informazioni riguardanti i cartellini ricevuti dai giocatori, nello specifico il tipo di cartellino, la partita e il minuto di gioco in cui è stato ricevuto.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://visualdataweb.org/FootOntologyPlus/>

SELECT ?player ?cardType ?match ?minute
WHERE {
    ?player :hasReceived ?card .
    ?card :MinuteOfEvent ?minute ;
        :isEventIn ?match ;
        rdf:type ?cardType .
    ?cardType rdfs:subClassOf :Card .
    FILTER (?cardType != :Card) .
}
ORDER BY ASC (?player)
```

?player	?cardType	?match	?minute
:Player17	:RedCard	:Match1	53
:Player5	:YellowCard	:Match1	23

Figura 6.3: Risultato della query sulla nostra ontologia.

6.4 Interrogazione sulle sostituzioni effettuate in una partita

Questa interrogazione le informazioni riguardanti le sostituzioni effettuate in una partita (nell'esempio `Match1`), nello specifico il nome dei giocatori coinvolti e il minuto di gioco.

```
PREFIX : <http://visualdataweb.org/FootOntologyPlus/>

SELECT ?enteringName ?exitingName ?minute
WHERE {
    ?sub :isSubstitutionIn :Match1 ;
        :hasEnteringPlayer ?entering ;
        :hasExitingPlayer ?exiting ;
        :MinuteOfEvent ?minute .
    ?entering :FullName ?enteringName .
    ?exiting :FullName ?exitingName .
}
ORDER BY ASC (?minute)
```

?enteringName	?exitingName	?minute
Kirill Volkov^^xsd:string	Mohamed Khalifa^^xsd:string	55
Thiago Oliveira^^xsd:string	Jean Dubois^^xsd:string	75

Figura 6.4: Risultato della query sulla nostra ontologia.

6.5 Interrogazione sui goal segnati dai giocatori

Questa interrogazione recupera il numero di goal segnati da ogni giocatore in un torneo (nell'esemptio `League1`).

```
PREFIX : <http://visualdataweb.org/FootOntologyPlus/>

SELECT ?player ?playerName (COUNT(?goal) as ?totalGoals)
WHERE {
    ?match :includedInTournament :League1 ;
        :hasGoal ?goal .
    ?goal :scoredByPlayer ?player .
    ?player :FullName ?playerName
}
GROUP BY ?player ?playerName
ORDER BY DESC (?totalGoals)
```

?player	?playerName	?totalGoals
:Player14	David Brown^^xsd:string	1
:Player19	Jan Kowalski^^xsd:string	1
:Player7	João Silva^^xsd:string	1

Figura 6.5: Risultato della query sulla nostra ontologia. Il nome dell'individuo giocatore è incluso in caso di omonimi.

Capitolo 7

Conclusioni

Bibliografia

[1] *Footology*. licensed under CC BY 4.0. URL: https://github.com/arditb1997/footology.