# Consultas flexibles sobre grafos de conocimiento

José Félix Yagüe[1], Ignacio Huitzil[12], Carlos Bobed[12], and Fernando Bobillo[12]

[1] University of Zaragoza, Zaragoza, Spain
[2] Aragon Institute of Engineering Research (I3A), Zaragoza, Spain
{ihuitzil,cbobed,fbobillo}@unizar.es

**Resumen** Este artículo corto discute un enfoque basado en lógica difusa para responder consultas flexibles sobre grafos de conocimiento, que incluyen restricciones sobre los valores de las propiedades descritas mediante conjuntos difusos. La principal ventaja es que permite reutilizar grafos RDF y puntos de acceso SPARQL existentes, añadiendo una capa difusa sobre ellos. Además, se permite el uso de tipos de datos difusos descritos mediante Fuzzy OWL 2 para la definición de las funciones de pertenencia difusas. La propuesta se ha implementado en el sistema FUKG y se ha validado en la DBpedia. Este artículo es un resumen de [6].

**Keywords:** Grafos de conocimiento · Lógica difusa · Consultas flexibles.

## 1. Introducción

La Inteligencia Artificial requiere con frecuencia combinar técnicas de representación del conocimiento y razonamiento con técnicas de gestión de la incertidumbre. Los grafos de conocimiento permiten capturar datos del mundo real a gran escala, usando nodos para representar entidades de interés y aristas para representar relaciones entre ellas. El caso más habitual asume el lenguaje estándar RDF (Resource Description Framework), basado en triples de la forma $\langle s,p,o \rangle$, que expresan que la entidad sujeto $s$ está relacionada con la entidad objeto $o$ a través del predicado (o propiedad) $p$. Para consultar grafos de conocimiento, lo más habitual es usar puntos de acceso basados en el lenguaje de consulta estándar SPARQL.

Nuestro objetivo es mejorar los sistemas basados en grafos de conocimiento a través del soporte de imprecisión, mediante lógica difusa. En particular, deseamos puntos de acceso capaces de responder consultas flexibles que incluyan términos imprecisos como, por ejemplo, "rápido" o "ligero" cuando se buscan vehículos.

Una primera aproximación para ello sería proponer un nuevo lenguaje para representar grafos de conocimiento, un nuevo lenguaje para especificar consultas y, por supuesto, un nuevo procedimiento para responder las consultas. En este artículo corto, que es un resumen de un trabajo publicado en junio de 2024 en una revista [6], proponemos un enfoque alternativo que permite reusar los lenguaje estándar RDF y SPARQL, así como los puntos de acceso existentes. Nuestras consultas flexibles se describirán utilizando conjuntos difusos como posibles valores de las propiedades de datos numéricas. Por ejemplo, dada una propiedad "velocidad", se permite como posible valor "alta velocidad". Para responder este tipo de consultas, añadiremos una "capa difusa" por encima de los estándares para manejar los términos flexibles.

## 2. Resolución de consultas flexibles

**Tipo de consultas.** Dado un grafo de conocimiento $\mathcal{K}$, las consultas flexibles que nos planteamos resolver se caracterizan por los siguientes parámetros:

- Conjunto de clases $C_1$,$C_2$,...,$C_N$
- Conjunto de propiedades numéricas funcionales $P_1$,$P_2$,...,$P_n$
- Lista de conjuntos difusos $D_1$,$D_2$,...,$D_n$
- Lista de modificadores difusos (*hedges*) $H_1$,$H_2$,...,$H_n$
- Operador para fusionar clases $\diamond$: intersección $\wedge$ (por defecto) o unión $\vee$.
- Operador para fusionar las propiedades $@:[0,1]^n \to [0,1]$ (una t-norma, una t-conorma, un operador OWA, etc.).
- Número máximo de resultados $k$.

Por ejemplo, si queremos consultar a la DBpedia (http://dbpedia.org) acerca de "motocicletas o automóviles con velocidad máxima muy alta y peso en vacío medio", tendríamos dbo:Motorcycle y dbo:Automobile como clases, dbp:maxSpeedKmh y dbp:emptyWeightKg como propiedades, HighmaxSpeedKmh y NeutralemptyWeightKg como conjuntos difusos, muy como modificador difuso, la unión como operador de fusión para las classes y la t-norma del producto para las propiedades.

Un aspecto clave de nuestra propuesta es la representación de los conjuntos difusos y los modificadores difusos usando los tipos de datos (*datatypes*) del lenguaje Fuzzy OWL 2 [2], que propone una sintaxis en XML para codificar las funciones trapezoidal, triangular, Z (o *left-shoulder*) y S (o *right-shoulder*) usando anotaciones (*annotation properties*) del lenguaje estándar para ontologías OWL 2 (que, al estilo de los comentarios de un programa, pueden ser ignorados por ciertas aplicaciones para mantener compatibilidad hacia atrás). Por ejemplo, HighmaxSpeedKmh, definido como una función de pertenencia triangular con parámetros (500,1000,1500), se puede codificar mediante el siguiente conjunto de tres triples RDF (obsérvese que el objeto del tercer triple es una cadena de caracteres multilínea en sintaxis XML):

```
ex:fuzzyLabel      rdf:type      owl:AnnotationProperty .
ex:HighmaxSpeedKmh rdf:type      rdfs:Datatype .
ex:HighmaxSpeedKmh ex:fuzzyLabel """<fuzzyOwl2 fuzzyType=\"datatype\">
     <Datatype type=\"triangular\" a=\"500\" b=\"1000\" c=\"1500\" />
  </fuzzyOwl2>""" .
```

**Resolución de consultas.** Para responder a estas consultas flexibles, proponemos un procedimiento ([6, Algoritmo 1]) que se ilustra en la Figura 1:

1. Recuperar las instancias de las clases $x$ y sus valores $y_i^X$ para cada propiedad $P_i$. Esto se resuelve con una consulta SPARQL, donde la primera parte explota los caminos de propiedades rdf:type/rdfs:subClassOf* y, si el operador de fusión es la unión, se combinan subconsultas con UNION. Por ejemplo, se recupera dbr:Fiat_G.91Y, con valor "1110"^^xsd:decimal para la propiedad dbp:maxSpeedKmh.
2. Para cada conjuntos difuso $D_i$ (como ex:HighmaxSpeedKmh), recuperar su función de pertenencia $F_i$, con consultas SPARQL como por ejemplo:

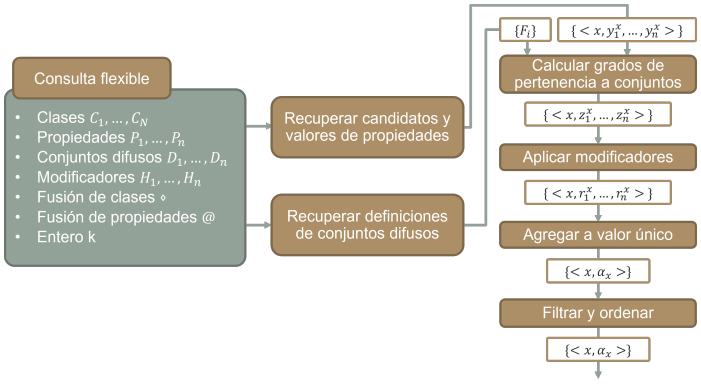



**Figura 1.** Diagrama de flujo del algoritmo del proceso de respuesta a consultas

```
SELECT ?Fi WHERE {
   ex:HighmaxSpeedKmh  rdf:type       rdfs:Datatype .
   ex:HighmaxSpeedKmh  ex:fuzzyLabel  Fi . }
```

3. Calcular, para cada individuo $x$, el grado de pertenencia del valor $y_i^x$ al conjunto difuso $F_i$: $z_i^x = F_i(y_i^x)$. Por ejemplo, para la velocidad máxima del dbr:Fiat_G.91Y, que era 1110, el grado de pertenencia a ex:HighmaxSpeedKmh es $z_i^x = 0{,}78$.
4. Aplicar los modificador difusos a los valores anteriores: $r_i^x = H_i(z_i^x)$. Si muy$(x) = x^2$, en el ejemplo anterior obtenemos $r_i^x = 0{,}78^2 = 0{,}61$.
5. Agregar todos los $r_i^x$ para cada individuo $x$: $\alpha_x = @_{i \in \{1,\ldots,n\}}(r_i^x)$. En nuestro ejemplo, tomando el producto para agregar los valores correspondientes a la velocidad máxima (0,61) y al peso (0,55, no detallado aquí), obtenemos $\alpha_x = 0{,}34$.
6. Por último, se forma una lista de valores $\langle x, \alpha_x \rangle$, como $\langle$dbr:Fiat_G.91Y,0,34$\rangle$, se ordenada decrecientemente y se filtra para quedarse con los $k$ mayores valores.

**Implementación.** FUKG (*FUzzy Knowledge Graphs*) es una aplicación Java que implementa el procedimiento anterior y está disponible en http://webdiis.unizar.es/~fbobillo/fukg. Su interfaz de usuario permite proporcionar la información relevante (punto de acceso del grafo de conocimiento y parámetros de la consulta) de una manera cómoda. Para interactuar con el punto de acceso, usa Apache Jena y Jena Java API. La herramienta incorpora funcionalidad como autocompletado de entidades u ocultamiento de IRIs. Dado que uno de los objetivos es permitir responder consultas flexibles sobre grafos de conocimiento existentes que no incluyen las definiciones de los conjuntos difusos, FUKG permite proporcionarlas a través de un fichero local independiente al grafo.

**Validación.** FUKG se ha validado a través del consultas a la DBpedia, que tiene un tamaño muy considerable, y a pesar de ello el tiempo de consulta es inferior a un segundo, si bien la inicialización (que requiere recuperar una única vez la lista completa de clases y propiedades) requirió cerca de un minuto.

## 3. Discusión

**Trabajo relacionado.** La literatura incluye varias extensiones difusas de los grafos de conocimiento (en particular, de grafos RDF). Principalmente, se definen axiomas difusos: por ejemplo, U. Straccia considera cuádruplas de la forma $\langle s,p,o,\alpha\rangle$, con $\alpha\in(0,1]$, que expresan que el triple RDF $\langle s,p,o\rangle$ es cierto con grado de verdad $\alpha$ [5]. Este tipo de trabajos no permiten reutilizar directamente el lenguaje estándar RDF, ni tampoco prestan atención a los conjuntos difusos descritos mediante funciones de pertenencia. Con respecto a la consulta de grafos RDF difusos, se han propuesto extensiones de los lenguajes de consulta. Por ejemplo, J. Z. Pan et al. propusieron Fuzzy SPARQL [4], que utiliza comentarios para codificar la parte difusa y hace el lenguaje compatible hacia atrás. Sin embargo, Fuzzy SPARQL no permite funciones de pertenencia difusas en las consultas. Por otro lado, en la propuesta de U. Straccia se extiende la noción de consultas conjuntivas al caso difuso, permitiendo la aparición de funciones de pertenencia, aunque no se propone un lenguaje de consulta concreto [5].

**Trabajo futuro.** Existen varias líneas de trabajo futuro definidas. Por un lado, se pretende extender la propuesta a más tipos de consultas, teniendo algunos resultados para consultas cuantificadas difusas de la forma "Q de X son F" [3]. Por otro lado, se pretende poder gestionar los valores faltantes, puesto que actualmente un único valor desconocido para cierta propiedad descarta completamente al individuo como candidato a ser recuperado. En este sentido, se tienen algunos resultados, donde una métrica inspirada en la entropía permite cuantificar la importancia de las distintas propiedades, lo que se tiene en cuenta para la resolución de consultas [1]. Finalmente, se pretende aplicar la herramienta a casos prácticos concretos para evaluar el desempeño de la herramienta al manejar grandes cantidades de datos.

**Agradecimientos.** Trabajo financiado por los proyectos de I+D+i PID2024-159530OB-I00 (MCIN/AEI 10.13039/501100011033) y UZ2024-IyA-02 (Universidad de Zaragoza).

## Referencias

1. Bobillo, F., Bobed, C., Mena, E., Straccia, U.: A fuzzy logic-based approach to semantic query answering with missing values. In: Proceedings of the 33rd IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2024). pp. 1–8. IEEE (2024)
2. Bobillo, F., Straccia, U.: Fuzzy ontology representation using OWL 2. International Journal of Approximate Reasoning **52**(7), 1073–1094 (2011)
3. Palacín, E., Huitzil, I., Lisi, F.A., Straccia, U., Bobillo, F.: Fuzzy quantification over OWL ontologies and knowledge graphs (2026). https://doi.org/10.48550/arXiv.2606.25778
4. Pan, J.Z., Stamou, G., Stoilos, G., Thomas, E., Taylor, S.: Scalable querying service over fuzzy ontologies. In: Proceedings of WWW 2008. pp. 575–584 (2008)
5. Straccia, U.: A minimal deductive system for general fuzzy RDF. In: Proceedings of the 3rd International Conference on Web Reasoning and Rule Systems (RR 2009). Lecture Notes in Computer Science, vol. 5837, pp. 166–181. Springer (2009)
6. Yagüe, J.F., Huitzil, I., Bobed, C., Bobillo, F.: FUKG: Answering flexible queries over knowledge graphs. The Electronic Library **42**(3), 368–392 (2023). https://doi.org/10.1108/EL-02-2023-0052