Resumen

Esta tesis presenta una extensión del popular lenguaje XPath, que ofrece una lista de respuestas ordenadas a una consulta flexible aprovechando las variantes difusas de los operadores *and*, *or* y *avg* para las condiciones XPath, así como dos restricciones estructurales, llamadas *down* y *deep*, para el que se asocia un cierto grado de relevancia. En la práctica, este grado es muy bajo para algunas respuestas obtenidas con la consulta original, y por lo tanto, no deberían ser calculadas, con el fin de aliviar la complejidad computacional del proceso de recuperación de información. Con el fin de mejorar la escalabilidad de nuestro intérprete para hacer frente a archivos XML grandes, hacemos uso de la capacidad de la programación lógica difusa para descartar de forma anticipada los cálculos que conducen a soluciones poco significativas (es decir, con un pobre grado de relevancia según las preferencias expresadas por los usuarios cuando usan el nuevo comando FILTER). Nuestra propuesta se ha implementado en un lenguaje lógico difuso, aprovechando los altos recursos expresivos de este paradigma declarativo para la gestión de "umbrales dinámicos" de una manera natural y eficiente. Además de utilizar nuestro entorno FLOPER para desarrollar el intérprete, también proponemos su implementación con el lenguaje estándar XQuery. Básicamente, definimos una biblioteca XQuery capaz de gestionar de forma difusa expresiones XPath, de tal manera que nuestro FuzzyXPath puede ser codificado como expresiones XQuery. Las ventajas de nuestro enfoque es que cualquier interprete XQuery puede manipular una versión borrosa de XPath mediante el uso de la biblioteca que hemos implementado.

Por otro lado, se presenta un método para depurar consultas XPath, describiendo cómo las expresiones XPath puede manipularse para obtener un conjunto de consultas alternativas que coincidan con un documento XML determinado. Para cada nueva consulta, damos un "chance degree" que representa una estimación de su desviación con respecto a la expresión inicial. Nuestro trabajo se centra en pro-

porcionar a los programadores un repertorio de alternativas (que contienen nuevos comandos como las etiquetas "JUMP/DELETE/SWAP") que se pueden utilizar para obtener mas respuestas. Nuestro depurador es capaz, de la misma manera que el intérprete, de gestionar grandes documentos XML haciendo uso del comando FILTER que ignora de forma anticipada cálculos que conducen a soluciones no significativas (es decir, con un "chance degree" muy rebajado, según las preferencias del usuario). El punto clave, nuevamente, es la capacidad natural para realizar "umbralizacion dinámica" que ofrece el lenguaje lógico difuso usado para implementar la herramienta, conectando así de alguna manera con el llamado «top-k answering problem» muy conocido en la lógica difusa y el soft-computing (o computación flexible).

En cuanto a nuevas aplicaciones no estándares, en el último bloque de esta tesis reforzamos las sinergias bilaterales entre FuzzyXPath y FLOPER. En particular, nos ocupamos de fórmulas proposicionales difusas que contienen varios símbolos proposicionales vinculados con conectivos definidos en un retículo de grados de verdad más complejos que *Bool*. En primer lugar, recordamos un método basado en SMT (*Satisfiability Modulo Theories*) difuso para demostrar automáticamente teoremas en relevantes lógicas con infinitos valores (incluyendo a las de *Łukasiewicz* y *Gödel*). A continuación, en lugar de centrarnos en cuestiones de satisfactibilidad (es decir, demostrar la existencia de al menos un modelo) como normalmente se hace en un entorno SAT/SMT, nuestro interés se traslada al problema de encontrar un conjunto de modelos (sobre un dominio finito) para una fórmula difusa dada. Reutilizaremos un método anterior basado en la programación lógica difusa donde la fórmula se concibe como un objetivo de un árbol de derivación, proporcionado por nuestra herramienta FLOPER, que contiene en sus ramas todos los modelos de la fórmula original, junto con otras interpretaciones (obtenidas tras interpretar de forma exhaustiva cada símbolo proposicional de todas las formas posibles con respecto a un conjunto de valores recogidos en un retículo subyacente de grados-de-verdad). A continuación utilizamos la capacidad de la herramienta FuzzyXPath para explorar estos árboles de derivación una vez exportados a formato XML, con el fin de detectar automáticamente si la fórmula es una tautología, satisfactible o una contradicción.