Università di Bologna

Dipartimento di Scienze dell'Informazione

Reperimento delle Informazioni nelle Librerie Digitali Distribuite di Conoscenza Matematica Formale

Ferruccio Guidi

fguidi@cs.unibo.it

INTRODUZIONE (1/2)

La nostra attività di ricerca si situa nel contesto di HELM (la *Hypertextual Electronic Library of Mathematics*), che è sviluppata dal Prof. Asperti e dal suo gruppo presso l'Università di Bologna.

HELM intende integrare i moderni strumenti per l'automazione del ragionamento formale (proof assistants) con le recenti tecnologie per lo sviluppo di applicazioni *Web* e dell'*electronic publishing*.

Scopo ultimo del progetto è lo sviluppo di un'opportuna tecnologia per creare e mantenere una libreria digitale, ipertestuale e distribuita, di conoscenza matematica formale.

Nello spirito del *Semantic Web*, i documenti di HELM sono corredati da metadati RDF (*Resource Description Framework*), che forniscono informazioni orientate alla macchina sulla loro struttura e contenuto.

INTRODUZIONE (2/2)

HELM sfrutta queste informazioni per fornire dei servizi di ricerca, sia interattiva che automatica, che consentono di reperire i documenti in base a richieste che tengano conto della loro natura matematica.

Per sviluppare questi servizi occorre un motore di ricerca che agisca sui metadati di HELM, ma gli attuali linguaggi d'interrogazione non forniscono alcune importanti funzionalità ritenute auspicabili.

In questo contesto, l'obiettivo della nostra ricerca è lo sviluppo di un linguaggio d'interrogazione che abbia i requisiti richiesti dalla comunità RDF e che soddisfi le necessità di HELM.

Gli aspetti peculiari del nostro linguaggio, **MathQL-1**, riguardano le risposte alle interrogazioni, che hanno una struttura 4-dimensionale e possiedono una loro sintassi interpretata da una semantica rigorosa.

IL SEMANTIC WEB E RDF (1/2)

La maggior parte delle informazioni sul *Web* sono orientate alla persona. Il *Semantic Web* è un estensione del *Web* in cui le informazioni hanno una forma che permetta ad una macchina di comprenderle e di trattarle automaticamente.

RDF è parte dell'attività del W3C sui metadati ed è uno standard per descrivere le risorse *Web* a livello di contenuto generale.

RDF incoraggia la visione dei "metadati come dati" usando XML, con URI e Namespace, come tecnologia privilegiata.

I metadati possono essere definiti dall'utente (le parole chiave di un documento) o generati in automatico (le dipendenze fra documenti).

La distinzione fra dati e metadati non è assoluta e la stessa informazione può essere interpretata simultaneamente nei due modi.

IL SEMANTIC WEB E RDF (2/2)

RDF è accuratamente disegnato per avere le seguenti caratteristiche:

- **Indipendenza**: i metadati devono essere descritti senza assunzioni sul loro dominio di applicazione o significato.
- **Interscambio**: i metadati devono essere facili da trasportare.
- **Scalabilità**: dev'essere facile trattare grandi insiemi di metadati.

Un database RDF contiene asserzioni su risorse che hanno proprietà con valori. Risulta che questa struttura descrive in modo naturale la maggioranza dei dati *Web* trattati dalle macchine.

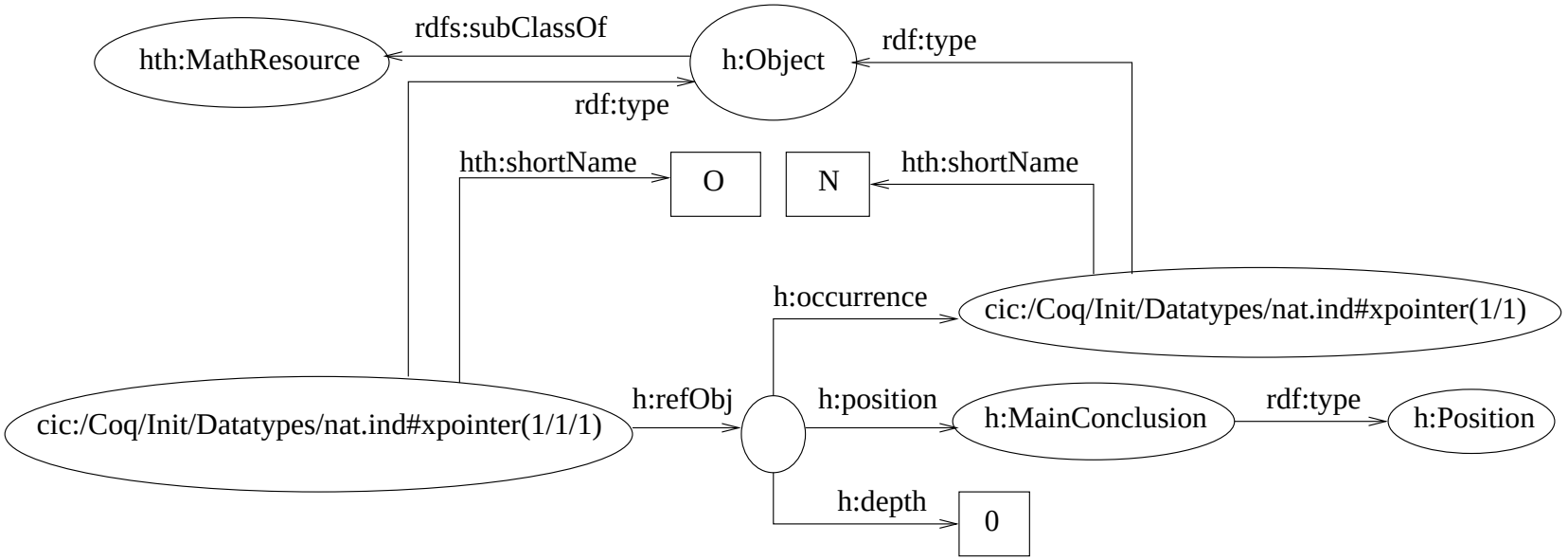
Un database RDF può essere descritto come un grafo orientato etichettato in cui una proprietà corrisponde a un arco fra la risorsa a cui si applica e il valore che ha (può essere una risorsa o un letterale).

UNA PARTE DEL GRAFO RDF DI HELM

Questo grafo RDF rappresenta alcuni metadati sulla dichiarazione del numero naturale zero (chiamato "O"), come appaiono nella libreria.

Questa dichiarazione è della forma: $O : N$.

I nodi ovali sono per le risorse e quelli squadrati per i letterali.



METADATI USATI DA HELM

Gli attuali metadati di HELM (84 Mbyte di file RDF) riguardano informazioni generali sui documenti (inserite a mano) e informazioni sulle dipendenze fra documenti (generate in automatico).

Le informazioni sulle dipendenze descrivono riferimenti a costanti, sorte e variabili includendo altre proprietà quali le posizioni dei riferimenti e il numero di premesse di un'ipotesi o conclusione.

Si noti che queste descrizioni si avvalgono di metadati strutturati.

I metadati sui riferimenti sono pensati per catturare gli invarianti di alcune importanti operazioni di confronto fra termini e consentono di:

- trovare i teoremi di una certa forma;
- trovare i teoremi che possono dimostrare un dato asserto.

IL SEMANTIC WEB E RDF (3/2)

Le risorse possono avere un nome o essere anonime. La proprietà *type* è tale che l'asserzione $(r, type, r')$ modella "r ha tipo r'".

Il valore di una proprietà può essere semplice o strutturato (una risorsa anonima con proprietà che descrivono le componenti).

Altre rappresentazioni di un database RDF includono:

- **triple RDF**: un insieme di asserzioni descritte come triple;
- **sintassi RDF**: RDF usa XML come sintassi d'interscambio. Le risorse e le proprietà predefinite hanno il Namespace "RDF:".

RDF fornisce un linguaggio per la descrizione di vocabolari (*RDF Schema*), pensato per scambiare i metadati fra differenti comunità, che permette di imporre gerarchie di proprietà e insiemi di valori.

IL SEMANTIC WEB E RDF (4/2)

RDF Schema (o RDFS) è specificato in termini di RDF usando un sistema gerarchico di classi. Qui una classe è un insieme di valori.

Le classi RDFS sono simili a quelle dei linguaggi orientati agli oggetti, ma le proprietà sono definite in termini delle classi a cui si applicano.

RDFS definisce le proprietà *domain* e *range*, che si applicano alle proprietà stesse e descrivono il loro dominio e codominio.

Altre classi e proprietà predefinite da RDF Schema sono:

- *Class*: la classe delle classi;
- *Resource*: la classe delle risorse;
- *subClassOf*: permette di costruire gerarchie di classi;
- *subPropertyOf*: permette di costruire gerarchie di proprietà.

LINGUAGGI DI INTERROGAZIONE (1/1)

Un linguaggio di interrogazione per RDF è una sintassi per esprimere richieste ad un database RDF, visto come insieme di asserzioni RDF.

Abbiamo analizzato le principali proposte e implementazioni:

- *rdfDB*, *Algae*, *SquishQL*, *DAML+OIL* e *RDF Query*: solo richieste congiuntive e nessun supporto alle richieste basate sulla gerarchia inferita da RDF Schema.
- *RDFQL*, *RDQL*, *RQL*, *RuleML* e *XDD*: hanno una sintassi testuale o una sintassi XML ma non entrambe.
- *RDF Database Access Protocol*, *RDFPath*, *TRIPLE* e *Metalog*: non c'è ancora una descrizione completa della loro sintassi.

Nessuno di questi linguaggi ha una semantica formale ben definita.

UNA PANORAMICA DI MathQL-1 (1/3)

Il nostro linguaggio ha le seguenti caratteristiche principali:

- Nelle richieste: capacità di imporre vincoli basati su RDF Schema e di attraversare i valori composti delle proprietà.
- Nelle richieste: diversi operatori logici per comporre i vincoli e capacità di selezionare le risorse usando espressioni regolari.
- Capacità di elaborare le risposte che hanno 4 dimensioni.
- Una sintassi XML (orientata alla macchina) e una sintassi testuale (orientata alla persona) per le richieste e per le risposte.
- una semantica ben definita, per le richieste e per le risposte, basata su un modello astratto che cattura il concetto più rilevante di RDF: le relazioni fra risorse, cioè le proprietà.

UNA PANORAMICA DI MathQL-1 (2/3)

Le risposte sono insiemi di *valori attribuiti* (v.a.) ovvero stringhe *di testa* con un insieme di *attributi* il cui *valore* è una stringa multipla.

Gli attributi hanno per nome una lista di stringhe e gli insiemi di attributi possono essere partizionati in *gruppi* (cioè sottoinsiemi).

Gli insiemi di v.a. hanno una geometria a 4 dimensioni: si possono estendere indipendentemente le stringhe di testa, gli attributi di ciascun gruppo, i diversi gruppi e i valori di ciascun attributo.

Gli insiemi di v.a. possono contenere molti tipi di dati, in particolare:

- insiemi di triple RDF;
- insiemi di valori di proprietà RDF (anche strutturate);
- ogni altra informazione necessaria ad elaborare le risposte.

UNA PANORAMICA DI MathQL-1 (3/3)

- Accesso al database RDF: *property*.
- Gestione degli insiemi di v.a.: *add*, *ex*.
- Supporto per le variabili e l'iterazione: *let*, *for*, *select*.
- Accesso ad un generatore di richieste esterno: *gen*.
- Supporto per le funzioni di libreria (esterne al nucleo).

Ogni tipo di dato è rappresentato usando gli insiemi di v.a..

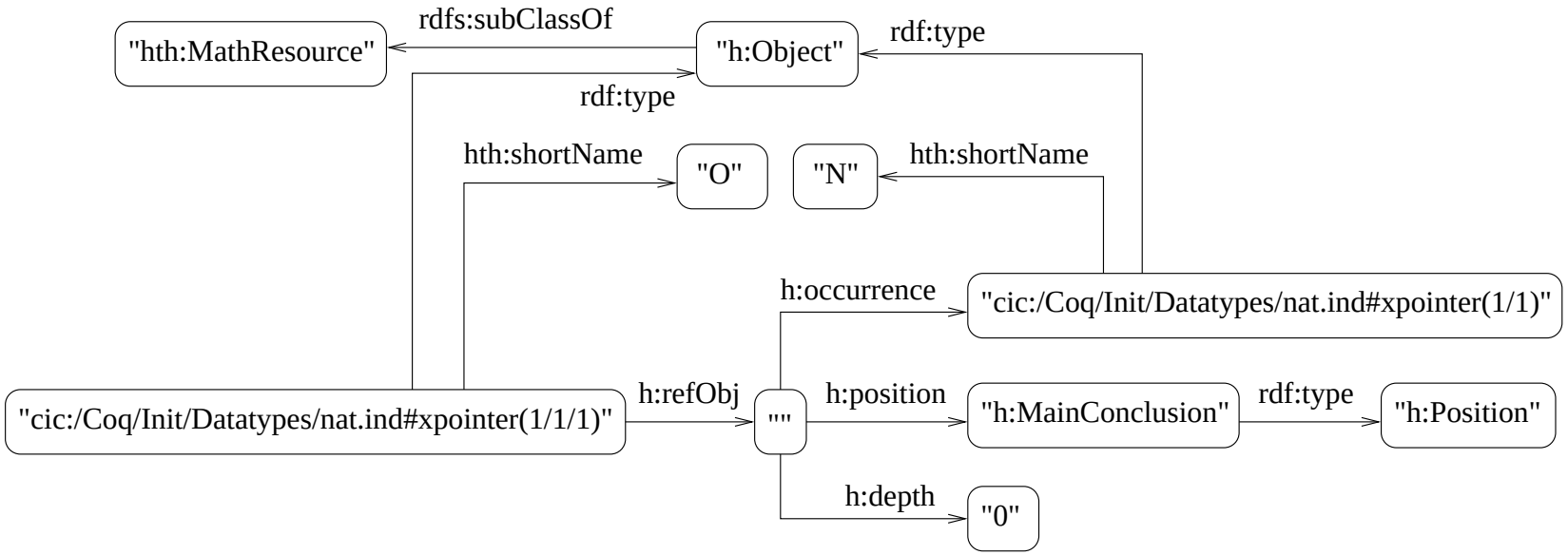
Le funzioni di libreria includono operazioni insiemistiche, operatore condizionale, operatori logici e numerici, composizione sequenziale, lettori e linearizzatori testuali e XML, utilità di debugging.

HELM usa un generatore di richieste che le costruisce a partire da una specifica ad alto livello dei documenti che si vogliono reperire.

UN GRAFO RDF ASTRATTO

Questo è lo stesso grafo di prima, mostrato però dal punto di vista dell'operatore *property*. I nodi sono di un sol tipo e le risorse anonime hanno un contenuto: la stringa vuota, che non è un URI.

Le radici del grafo sono le istanze della classe *h:Object*.



UNA PANORAMICA DI MathQL-1 (4/3)

property accede al grafo astratto del database RDF e ritorna le istanze di un albero descritto da un insieme di persorsi.

I percorsi iniziano dalle risorse a cui i metadati si riferiscono e contengono una sequenza (anche vuota) di archi contigui che denotano proprietà RDF (anche composte).

Le istanze ritornate vengono codificate in un insieme di v.a. usando un'opportuna strategia, che dipende dalla forma dell'albero in ingresso, per creare gli attributi e per partizionarli nei gruppi.

L'insieme delle istanze ritornate può essere ristretto vincolando dei nodi ad avere (o non avere) valori dati. I vincoli prevedono anche il confronto con espressioni regolari POSIX 1003.2-1992.

Con certe opzioni, *property* può accedere alla gerarchia di propietà.

L'USO DI MathQL-1 IN HELM (1/3)

HELM è implementato in Caml ad oggetti e la *Suite* MathQL-1, che è scritta e mantenuta da noi, include i moduli Caml per usare MathQL-1 nel contesto di HELM. Questa *Suite* include:

- Il componente Caml di base per MathQL-1: è indipendente da HELM e fornisce la rappresentazione Caml delle richieste e delle risposte più alcune funzioni di utilità generale.
- L'interprete MathQL-1: è indipendente da HELM e può lavorare in due modalità di accesso ai metadati: Postgres e Galax.
- Il generatore di richieste per HELM: genera alcuni tipi predefiniti di richieste, significative per i metadati di HELM, partendo da una descrizione ad alto livello dei documenti voluti.
- Le utilità di controllo: servono per verificare gli altri componenti.

L'USO DI MathQL-1 IN HELM (2/3)

Il modo più rapido per accedere ai metadati è di leggere i file RDF che li descrivono, costruire un database relazionale con le informazioni raccolte e permettere all'interprete di accedervi.

Questa tecnica, usata da molti motori di ricerca RDF, ha il vantaggio che i file RDF sono letti solo quando il database è costruito per cui l'interprete accede a informazioni già scandite.

Lo svantaggio è che il database deve essere aggiornato ogni volta che sono disponibili nuovi metadati (nel caso di HELM questo accade ogni volta che gli utenti aggiungono dei documenti alla libreria).

HELM mantiene un database PostgreSQL con i suoi metadati, con una tabella per ogni proprietà RDF (semplice o strutturata). La dimensione del database si aggira attorno ai 200 Mbyte.

L'USO DI MathQL-1 IN HELM (3/3)

Un altro modo di accedere ai metadati è di permettere all'interprete di leggere i file RDF durante l'esecuzione, quando un'informazione è richiesta (in questo caso i file possono essere letti più volte).

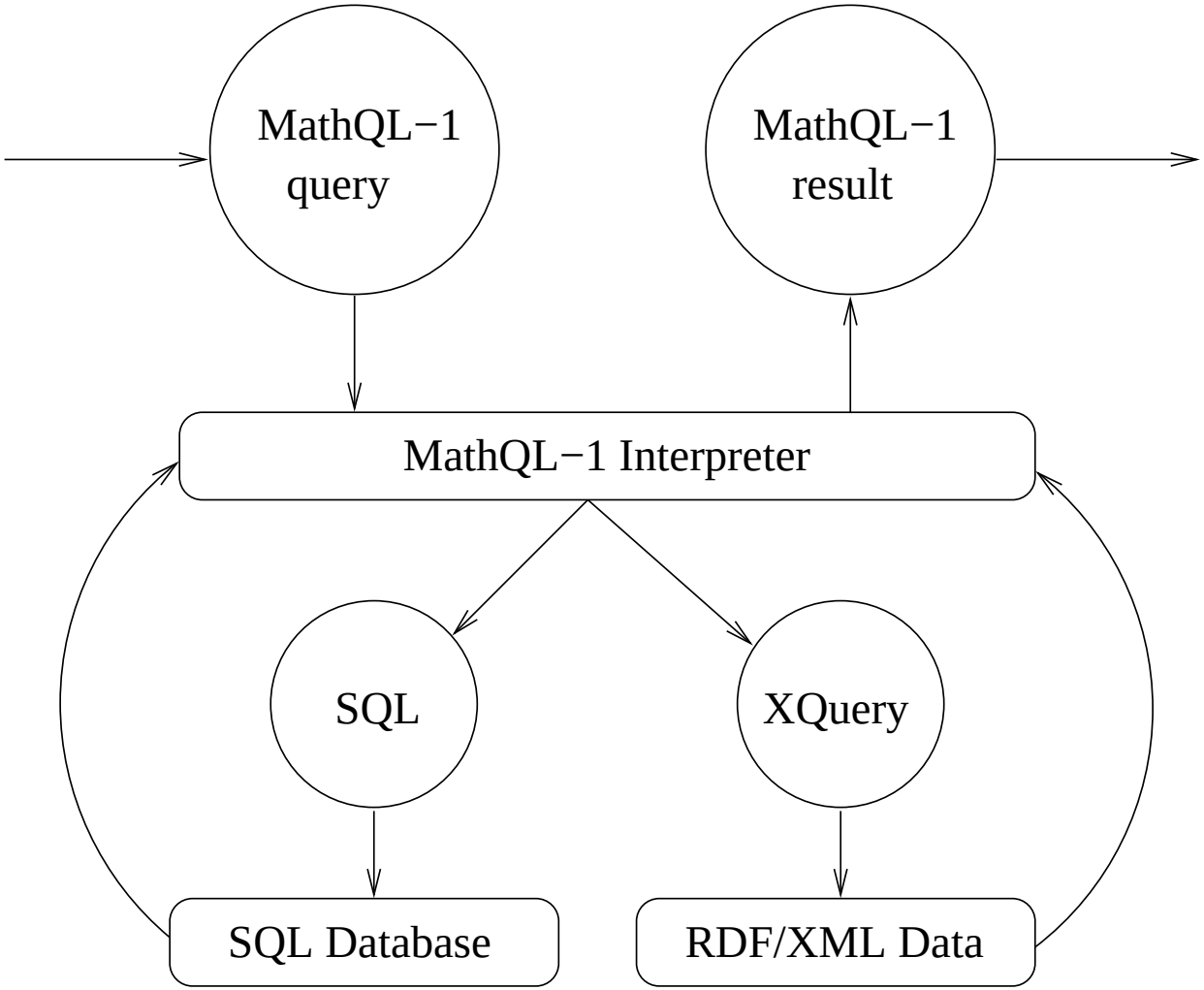
L'interprete può anche leggere i file RDF di HELM usando Galax, che è un'implementazione Caml di un motore di ricerca basato su XQuery: un linguaggio di interrogazione per database XML.

Dalle nostre misure risulta che in modo Galax le richieste vengono risolte 10 volte più piano rispetto alle stesse in modo Postgres.

La nostra implementazione dell'interprete è studiata per ottenere le migliori prestazioni ed è 100 volte più veloce delle implementazioni precedenti sviluppate all'interno del progetto HELM.

Galax non è ancora supportato nell'ultima implementazione.

LA STRUTTURA DELL'INTERPRETE



L'USO DI MathQL-1 IN HELM (4/3)

L'interprete valuta l'operatore *property* inoltrando una o più richieste a basso livello in SQL o XQuery, ed elaborando i loro risultati.

L'uso di un linguaggio di interrogazione orientato ai grafi al posto di SQL o XQuery permette di nascondere la rappresentazione concreta dei metadati RDF (tabelle di database o file XML) all'utente finale.

All'utente basta conoscere la struttura del grafo RDF e l'interprete può accedere a metadati memorizzati in formati diversi.

Questo approccio è impossibile usando solo SQL o XQuery, anche se questi linguaggi sono più espressivi di MathQL-1 in alcuni aspetti.

Si noti che una richiesta MathQL-1 non troppo complessa, come quella che fornisce le leggi transitive presenti in HELM, può richiedere molte migliaia di accessi alla base di metadati sottostante.