

# Aggiornamento progressi tesi

Quantum computing e inferenza logica

Davide Camino

November 29, 2025

## Abstract

### Ad alto livello

La tesi vuole esplorare le possibilità offerte dal quantum computing per fare inferenza su basi di conoscenza (*Knowledge Base* KB).

Le KB su cui ci concentriamo sono le ontologie espresse in OWL 2, un linguaggio formale per il *semantic web* sviluppato dal W3C Web Ontology Working Group. La versione più espressiva di questo linguaggio è OWL 2 DL che ha una semantica compatibile con la description logic *SRQIQ*.

I problemi di reasoning su *SRQIQ* DL sono N2ExpTime-complete, di conseguenza anche fare inferenza su ontologie espresse in OWL 2 DL ha la stessa complessità. Per evitare questo costo computazionale sono stati proposti dei sotto-linguaggi di OWL 2 DL che consentono di sviluppare algoritmi polinomiali per fare reasoning. In questo lavoro non ci occupiamo di queste varianti del linguaggio.

Sono stati sviluppati numerosi reasoner che hanno ottime performance anche quando operano su KB di grandi dimensioni. Anche se questi strumenti si comportano bene nella maggior parte dei casi esistono ontologie abbastanza grandi e complesse da mettere in crisi questi sistemi.

Il quantum computing sembra promettere di poter esplorare degli spazi esponenzialmente vasti in tempo polinomiale. Questo lavoro vuole proporre degli strumenti alternativi alla computazione classica che sfruttando quantum computing permettano fare inferenza su ontologie con complessità temporale inferiore a quella attuale.

### Tesi

#### Idea iniziale

Inizialmente l'idea era di esplorare a fondo il funzionamento di uno dei reasoner attualmente usati per fare inferenza su ontologie e valutare la possibilità di demandare una parte dell'elaborazione a un computer quantistico.

Questa idea per quanto interessante è stata scartata perché la mia competenza nell'ambito di questi reasoner è troppo limitata e non sarei riuscito in tempo utile a comprenderne appieno il funzionamento per poi poter implementare un pezzo dell'algoritmo su macchine quantistiche.

### **Pubblicazione di riferimento**

Il professor Roversi mi ha consigliato un lavoro che esplora la possibilità di mappare un programma e una query Prolog in una funzione di energia. Se si fa questa traduzione in modo appropriato è possibile fare in modo che la configurazione a minore energia corrisponda alla risposta della query.

Questa funzione di energia può essere riscritta come Hamiltoniana e il minimo può essere trovato attraverso quantum annealing.

L'articolo presenta e implementa QA-Prolog (Quantum Annealing Prolog): la pipeline sviluppata per poter eseguire un programma prolog (è concesso utilizzare solo un sottoinsieme dei comandi prolog) su un dispositivo della D-Wave. Ad alto livello le trasformazioni sono:

1. Prolog;
2. Circuito digitale codificato in Verilog;
3. Ottimizzazione e sintesi del circuito attraverso Yosys;
4. Dalle porte logiche a un'Hamiltoniana simbolica;
5. Dall'Hamiltoniana simbolica a quella fisica;
6. Esecuzione su quantum annealer o simulatore.

### **Idea Attuale**

È possibile usare prolog per esprimere delle basi di conoscenza, possiamo quindi sfruttare QA-Prolog per trasformare queste KB più una query associata in problemi risolvibili nativamente su macchine quantistiche (adiabatic quantum computing).

Attorno a questa idea centrale si possono costruire altre parti della pipeline sia a monte che a valle. Ne parliamo nella sezione sotto.

### **Stato attuale**

#### **QA-Prolog**

**Compatibilità:** La pipeline si compone di diversi strumenti che integrano pezzi software sviluppati apposta per QA-Prolog e librerie esterne. Il progetto ha ricevuto l'ultimo aggiornamento circa 6 anni fa. In questi giorni ho quindi lavorato per rendere nuovamente compatibili i vari pezzi della pipeline aggiornando le librerie usate e risolvendo eventuali problemi dovuti a istruzioni deprecate.

**Utilizzo:** Attualmente (con qualche difficoltà affrontata nel paragrafo successivo) riesco a percorrere interamente la pipeline e ottenere la risposta a una semplice query presente nella documentazione stessa di QA-Prolog.

Questo è già un risultato interessante dato che il progetto era stato abbandonato anche perché 6 anni fa si era in grado di risolvere problemi molto piccoli. Oggi le macchine della D-Wave sono decisamente più grandi di quelle usate nei test di allora e avere uno strumento del genere può rendere nettamente più alla portata di tutti la computazione su hardware quantistico.

**Problemi:** Nonostante io riesca a percorrere completamente la pipeline non riesco ancora a eseguire tutto lo stack attraverso un unico comando. In particolare la primissima traduzione aggiunge dei caratteri speciali al nome dei moduli (che sono la traduzione delle regole prolog) che vengono interpretati male dall'ultimo pezzo della pipeline (prima dell'esecuzione) questo impedisce la costruzione dell'Hamiltoniana fisica e il processo si interrompe.

Modificando manualmente il file Verilog è possibile rimuovere questi caratteri e eseguire tutti gli step successivi correttamente.

La spiegazione che mi sono dato è che vengono utilizzate delle librerie per fare il parsing e la riscrittura dei documenti, probabilmente una di queste librerie funziona diversamente (rispetto a quando il programma è stato scritto) e produce questi caratteri speciali a monte oppure li tratta in modo diverso a valle.

### Quantum Gate: QAUA

**Ragioni:** Il QAOA può essere considerato come una versione discreta di quello che avviene nelle macchine della D-Wave con l'adiabatic quantum computing. L'esecuzione di un singolo passo di QAUA è più rapida dell'evoluzione (quasi statica) che avviene dentro alle QPU di D-Wave, però fornisce risultati approssimati. Per ottenere risultati simili a quelli dell'adiabatic quantum computing è necessario rendere il circuito parametrico e iterare l'esecuzione per ottenere la stima migliore di questi parametri. Una volta trovati i parametri migliori si può eseguire nuovamente il circuito e leggere i risultati.

Nel nostro lavoro vogliamo quindi usare architetture basate su quantum gate e sfruttando l'algoritmo QAUA come alternativa al quantum annealing.

**Implementazione:** Una volta ottenuta una rappresentazione del problema come Hamiltoniana è facile esprimere il problema in forma QUBO o Ising, quest'ultima è facile da riscrivere come operatori di Pauli in modo da codificare la funzione energia per architettura a quantum gate.

**Difficoltà:** Mentre è facile imporre che una certa variabile booleana valga uno o zero usando il paradigma del quantum annealing, farlo per l'algoritmo QAUA è più complesso. Nel nostro lavoro è necessario poter imporre un certo valore alle variabili perché ci serve per valutare le query prolog.

Non è possibile imporre semplicemente il valore di una variabile, si deve fare in modo indiretto attraverso delle penalità imposte alla funzione energia. Stimare quanto grandi devono essere queste penalità potrebbe non essere semplice.

## **Lavoro futuro**

**Updated QA-Prolog:** Il primo obbiettivo che ho è far funzionare correttamente QA-Prolog, in modo che si possa eseguire un singolo comando per far partire la traduzione e leggere i risultati, senza che si producano errori o file incompatibili tra i vari step della pipeline.

**Q-Prolog:** potrebbe essere interessante ampliare le possibilità di QA-Prolog per farlo diventare Quantum Prolog In grado di eseguire il programma sia sfruttando il paradigma del quantum adiabatic computing sia architetture quantum gate based attraverso l'algoritmo QAUA.

**OWL to Prolog:** Esistono delle pubblicazioni che mostrano o a livello teorico o attraverso implementazioni come sia possibile tradurre un'ontologia in OWL in un programma Prolog. Quest'ultimo step chiuderebbe il cerchio fornendo davvero un set di strumenti per fare inferenza su ontologie attraverso hardware quantistico.