Relazione di "Sistemi complessi: modelli e simulazione"

Simon Vocella
Matricola: 718289

19 luglio 2012

1 Il problema

Al giorno d'oggi esistono molti sistemi di catalogazione dei paper scientifici (ex. DBLP), il nostro programma si propone di fare scraping in due o più di questi sistemi, con ovviamente dello scraping ad-hoc dipendente da come è strutturato il sito, e si propone di risolvere due problemi:

- Calcolare un indice di similarità tra i vari siti
- Filtrare i risultati di Google Scholar tramite DBLP

2 Sorgenti informative utilizzate

In questo progetto di è deciso di utilizzare i seguenti sistemi di catalogazione:

- DBLP (http://www.informatik.uni-trier.de/~ley/db/)
- Google scholar (http://scholar.google.it/)

3 Approccio e metriche adottate

Esiste un sistema centrale che comunica ai vari agenti cosa bisogna cercare. Ogni agente è specializzato in un sistema di catalogazione diverso. Nel nostro caso avremo due agenti. L'agente farà scraping nel proprio sito e ritornerà i risultati al sistema centrale. Man mano che ogni agente consegna i propri risultati, viene creata una lista $< Key, Value >$ in cui la Key sarà il titolo del paper j-esimo e $Value$ sarà uguale a $\sum_{i=0}^{n} i(j)/(n * m)$ dove $i(j)$ sarà uguale a 1 nel caso i-esimo agente al j-esimo paper, altrimenti 0 e m e n sono rispettivamente il numero dei paper trovati in totale e il numero degli agenti.
L'indice di similarità sarà la sommatoria dei vari $Value$ $\sum_{j=0}^{m} Value(j) = \sum_{j=0}^{m} \sum_{i=0}^{n} i(j)/(n * m)$, in questo caso l'indice ci indica quanto le n liste raccolte dagli n agenti siano simili.

Nel caso in cui $\sum_{i=0}^{n} i(j)/(n * m) = 1/m$ allora qualsiasi agente ha trovato il paper j-esimo.

Nel caso in cui l'indice di similarità sia $\sum_{j=0}^{m} Value(j) = \sum_{j=0}^{m} \sum_{i=0}^{n} i(j)/(n * m) = 1$, ci indica che le n liste di paper raccolte dagli n agenti sono identiche.

4 Architettura del sistema

DOMANDA: descrivo che ho eliminato il livello di python?
DOMANDA2: devo listare i programmi?

Il sistema è basato su quattro livelli diversi. Inizialmente abbiamo un piano di controllo basato su Jason che chiameremo Librarian. Librarian si prende l'onere di chiedere all'utente cosa vuole cercare, tramite una GUI in Swing e poi comunica il termine da ricercare ai due agenti Dblp e Scholar. Il nome è abbastanza esplicativo sul loro tipo di specializzazione. Ogni agente comunica un azione interna di $scraping.get\_information$ che comunica ad un interfaccia ad hoc di scraping chiamata Session. Session si preoccupa, una volta istanziata, di far partire un webkit modificato che rimane in ascolto su una porta specificata o di default che chiameremo per semplicità Webkit. Tramite Session comunichiamo le nostre azioni di visita al Webkit e tramite xpath manipoliamo il DOM della pagina e tramite funzioni di QT o Javascript manipoliamo funzioni come il submit, click o history, etc.
Ogni agente esegue il proprio scraping e manipola le pagine che visita per raccogliere informazioni (nel nostro caso i titoli dei paper), lo fa su una sola pagina nel caso di DBLP o su più pagine nel caso dii Google Scholar.
Una volta ricevuti i risultati, l'azione interna restituisce o meglio unifica (visto che parliamo di linguaggio funzionale) il risultato come una stringa e poi viene restituito all'oggetto Librarian.
Librarian si occuperà di raccogliere i paper in un HashMap e di calcolare incrementalmente l'indice di similarità.
Una volta che tutti gli agenti hanno restituito i loro risultati e Librarian ha calcolato completamente l'indice di similarità, Librarian mostra i risulati e chiude il programma.

5 Descrizione dei risultati

DOMANDA: devo fare qualche screenshot?

Risultati dell'agente Dblp:

- An analysis of different types and effects of asynchronicity in cellular automata update schemes
- Towards an agent-based proxemic model for pedestrian and group dynamics: motivations and first experiments
- An Agent Model of Pedestrian and Group Dynamics: Experiments on Group Cohesion
- An Agent-Based Proxemic Model for Pedestrian and Group Dynamics: Motivations and First Experiments
- A Cellular Automata Based Model for Pedestrian and Group Dynamics: Motivations and First Experiments
- ... etc.

Risultati dell'agente Scholar:

- Modeling dynamic environments in multi-agent simulation
- Situated cellular agents: A model to simulate crowding dynamics
- Situated cellular agents approach to crowd modeling and simulation

- Awareness in collaborative ubiquitous environments: The multilayered multi-agent situated system approach
- Toward a platform for multi-layered multi-agent situated system (MMASS)-based simulations: focusing on field diffusion
- ... etc.

Hashmap risultante:

- key: Web Intelligence and Intelligent Agent Technology. Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence, Worshops, value: 0.0017857142857142857
- key: Visualization of Discrete Crowd Dynamics in a 3D Environment, value: 0.0017857142857142857
- key: PASSIONE STORICA E STORIA CIVICA NELLA CALABRIA NORDOCCIDENTALE. RASSEGNA BIBLIOGRAFICA E RIFLESSIONI STORIOGRAFICHE, value: 0.0017857142857142857
- key: Le memorie del vecchio maresciallo, value: 0.0017857142857142857
- key: Bio-ICT Convergence: Filling the Gap Between Computer Science and Biology, value: 0.0017857142857142857
- ... etc.

Indice di similarità risultante: 0.5464285714285698.

6 Conclusioni

Come abbiamo visto le due liste di risultati differiscono per un buon numero di paper, in questo caso è colpa di Scholar che mette molti altri risultati oltre agli articoli scientifici (un es. è PASSIONE STORICA E STORIA CIVICA NELLA CALABRIA NORDOCCIDENTALE. RASSEGNA BIBLIOGRAFICA E RIFLESSIONI STORIOGRAFICHE, value: 0.0017857142857142857, che non mi sembra molto ad avere a che fare con l'informatica). Si potrebbe migliorare il risultato, aumentando il numero dei siti di catalogazione da visitare, così riducendo il peso di Scholar da 1/2 a 1/n dove n è il numero dei siti scelti.