Universitá degli Studi di Salerno

Dipartimento di Studi e Ricerche Aziendali

Dottorato di Ricerca in Scienze e Tecnologogia dell'Informazione,
dei Sistemi Complessi e dell'Ambiente

XII Ciclo

Abstract della tesi di dottorato
in
Software Engineering

Search-Based Software Maintenance and Testing

Annibale Panichella

Coordinatore
Prof. Roberto Scarpa

Tutor
Prof. Andrea De Lucia

Aprile 2014

Abstract

Durante il processo di sviluppo e manutenzione del software, l'ingegnere del software è chiamato ad effettuare un insieme di operazioni complesse e costose. Per tali ragioni, diversi ricercatori hanno suggerito di automatizzare tali processi con lo scopo di ridurre sensibilmente il costo di sviluppo e manutenzione del software, dal momento che l'automazione richiede un minore intervento da parte dell'uomo. Uno dei modi più utilizzati in letteratura per realizzare tale automazione è il *Search-Based Software Engineering* (SBSE), che consiste nel riformulare problemi tradizionali di ingegneria del software come problemi di ottimizzazione. In SBSE l'insieme di tutte le possibili soluzioni ad un dati problema ne definiscono lo spazio di ricerca (o la regione ammissibile) mentre una funzione di fitness è utilizzate per differenziare le varie soluzioni e fornire una guida al processo di ottimizzazione. Una volta riformulato un problema di ingegneria del software come problema di ottimizzazione, algoritmi di ricerca o ottimizzazione possono essere utilizzati per risolvere tale problema. In letteratura sono state proposti ed utilizzati diversi algoritmi di ottimizzazione, come ad esempio algoritmi genetici, genetic programming, simulated annealing, hill climbing (gradient descent), greedy algorithms, particle swarm and ant colony.

La presente tesi investiga e propone l'utilizzo di approcci search-based con lo scopo di ridurre il costo di manutenzione e testing del software, ponendo particolare attenzione a quattro attività principali: (i) comprensione del codice, (ii) predizione dei difetti, (iii) generazione automatica dei casi di test, e (iv) ottimizzazione della test suite per il testing di regressione. Per le prime due attività (comprensione del codice e predizione dei difetti) la presente tesi fornisce una loro prima riformulazione come problemi di ottimizzazione e propone l'utilizzo di Algoritmi Genetici (GA) per risolverli. Nello specifico, la tesi analizza le peculiarità del codice sorgente rispetto a documenti testuali scritti in linguaggio naturale, e propone l'utilizzo di algoritmi genetici (GA) per calibrare in modo automatico le tecniche di Information Retrieval (IR) a supporto di diversi task di manutenzione del software. La tesi, inoltre, analizza e propone l'utilizzo di Algoritmi Genetici Multi-Obiettivo (MOGA) per costruire modelli di predizione dei difetti multi-obiettivo che tengano in considerazione, durante il processo di predizione, delle contrastanti esigenze pratiche dell'ingegnere del software.

Le attività di (i) generazione automatica dei casi di test e (ii) ottimizzazione della test suite sono state ampliamente studiate utilizzando approcci search-based. Tuttavia, nonostante l'ampia letteratura a riguardo, entrambe le attività presentano diverse problematiche e una loro riformulazione come problemi di ottimizzazione non sempre fornisce le prestazioni attese. Il successo di algoritmi di ottimizzazione, e in particolare di algoritmi genetici, dipende da diversi fattori. Uno di questi fattori è il grado di diversità tra i diversi individui (soluzioni) che influisce sulla capacità degli algoritmi genetici nell'esplorare efficacemente lo spazio di ricerca. Ad esempio, la generazione automatica dei casi di test tramite approcci search-based pu essere affetta dalla *genetic drift*, ovvero, la riduzione della diversità tra le soluzioni che pu portare ad una convergenza prematura ad ottimi locali. Per questa ragione, la presente tesi analizza gli effetti di meccanismi volti a preservare la diversità tra gli individui (soluzioni) sulle performance degli algoritmi genetici e propone un nuovo meccanismi di diversità basato sulla Decomposizione a Valori Singolari (SVD) e sull'algebra lineare. Tale meccanismo è stato integrato all'interno dell'algoritmo genetico tradizione ed è stato valutato empiricamente nel contesto dell'attività di generazione automatica dei casi di test. Tale

meccanismo è stato, inoltre, integrato all'interno di un algoritmo genetico multi-obiettivo (MOGA) e valutato empiricamente nel contesto dell'attivitè di ottimizzazione della test suite per il testing di regressione.