Valutazione automatica di test con risposta aperta

Un metodo di test volto alla valutazione automatica di soluzioni creative date a problemi non elementari

- **Annalina Fabrizio**, XIV G.i. Accademia Navale
  annalinafab@libero.it
- **Giuseppe Fiorentino**, XIV G.i. Accademia Navale
  fiorent@dm.unipi.it
- **Giuliano Pacini**, XIV G.i. Accademia Navale
  pacini@di.unipi.it

INTRODUZIONE

Le esigenze della didattica moderna impongono tempi precisi di apprendimento con una sostanziale garanzia del risultato qualitativo. I nuovi indirizzi didattici chiedono spesso un cambiamento nella struttura dei corsi che da estensivi diventano intensivi. Gli studenti hanno pertanto un minor tempo per fissare i concetti, tanto che ogni lezione diviene di fatto "essenziale" ai fini della comprensione di quanto segue. Lo studente deve quindi apprendere durante le ore di lezione e consolidare immediatamente nell'esercitazione.

I nuovi indirizzi didattici si prefiggono anche di colmare il divario tra "sapere" e "saper fare", nella prospettiva di una maggiore integrazione degli aspetti teorico/nozionistici con i momenti pratico/applicativi. Diventa così fondamentale la possibilità di verificare giorno per giorno il livello di apprendimento della materia e la capacità di passare dalla nozione all'applicazione [Papert, 1993] [Jonassen e Land, 2000] [Varisco, 2001]. In questo senso, le tecniche per la valutazione da parte del docente e per l'auto-valutazione da parte dello studente diventano strumenti imprescindibili dal successo dell'attività didattica. Da un altro punto di vista, si può dire che l'obiettivo è quello di distribuire le verifiche su tutto il periodo d'insegnamento, riducendo il tal modo sia il livello di drammaticità che la durata dell'esame finale.

Appare evidente che la valutazione continua del livello d'apprendimento è un'attività sostenibile solo se viene (almeno in buona parte) automatizzata con l'ausilio di opportuni sistemi informatici. I sistemi di preparazione e attuazione di test oggi disponibili su calcolatore possono essere già molto utili allo scopo [Logora e Stermini, 2001] [Domenici, 1988]. Un esempio notevole di applicazione dei calcolatori nell'ambito del testing è offerto dal Computer Adaptive Testing (CAT), di cui il tutorial interattivo presente all'indirizzo http://EdRes.org/scripts/cat ne costituisce un'ottima introduzione [Rudner, 1998], insieme all'ampia bibliografia là citata.

Normalmente i sistemi attuali sono basati su test a risposta chiusa. Volendo estendere il concetto, l'ideale sarebbe sottoporre lo studente a test in grado di verificare la capacità di applicare le nozioni, con quesiti aperti all'attitudine creativa, oltre che all'esibizione delle conoscenze acquisite. Essenzialmente, si tratterebbe di valutare la reale capacità di applicare le nozioni per la soluzione di problemi di dimensione non minimale.

È giustamente opinione comune che la valutazione delle soluzioni liberamente date a problemi di una qualche sostanza richiede quella conoscenza della materia e quell'esperienza didattica proprie del docente. Un prodotto informatico con queste capacità di valutazione, se intese in senso compiuto, dovrebbe quindi produrre prove formali di correttezza delle soluzioni e fornire prestazioni di Intelligenza Artificiale di alto livello, probabilmente al di là delle attuali tecnologie.

Fermo restando quanto espresso in linea generale, nel presente articolo si propone

un metodo per valutare automaticamente la bontà di soluzioni date a problemi non elementari, posti in domini di conoscenza prevalentemente procedurali. In che senso, e con quali limitazioni, la nostra proposta raggiunge quello che potrebbe apparire un obiettivo ambizioso sarà chiarito nel seguito. Si focalizzerà l'attenzione su cosa s'intende per test a risposta aperta ed in che senso il metodo che proponiamo rientra a pieno titolo in questa tipologia. Successivamente si chiarirà la metodologia adottata per poter utilizzare una valutazione sostanzialmente black-box per test e quesiti, nel loro complesso, tutt'altro che banali. Inoltre si discuteranno eventuali tecniche di modulazione della difficoltà dei test che il metodo proposto mette a disposizione. In ultimo si discuteranno due complete implementazioni del metodo effettivamente utilizzate per l'insegnamento di Access ed Excel presso L'Accademia Navale di Livorno.

RISPOSTE APERTE E VALUTAZIONE BLACK-BOX

Il nostro metodo intende avvalersi di quesiti che pongono problemi non elementari, aspirando a valutare se lo studente è in grado di darne risoluzione sulla base delle nozioni di cui dispone. Certi problemi ammetteranno soluzioni esprimibili con delle formule, adatte ad esempio a questioni di matematica o di ingegneria. Nel caso di problemi informatici, supponiamo nell'ambito delle basi di dati, la risposta potrà essere data con una query espressa in SQL o con altri formalismi, magari di tipo semigrafico. Nel caso della geometria euclidea la soluzione potrà corrispondere alla descrizione di un procedimento; si pensi, tanto per fissare le idee, alla sequenza di passi necessari per tracciare la bisettrice di un angolo.
Nel presente articolo ci occuperemo di test che propongono problemi con soluzioni di natura operativa, per esempio procedure o formule di calcolo. È chiaro che ogni soluzione procedurale fornisce dei risultati dipendenti dai dati iniziali del problema. Per esempio, ogni angolo avrà la sua bisettrice, sempre determinabile con la stessa sequenza di passi operativi.
Bisogna a questo punto precisare che alla base del nostro metodo c'è l'idea che le risposte ai quesiti sono le soluzioni, e non i risultati che con esse si ottengono. È in questo senso che le risposte sono aperte, in quanto la soluzione può essere liberamente formulata. Si noti la differenza con i test a risposta numerica, dove la risposta è costituita dal risultato e non dal procedimento seguito per ottenerlo.
In generale, l'esame di soluzioni liberamente espresse, per problemi non banali, richiede conoscenza della materia ed esperienza didattica, necessarie per valutare la bontà della soluzione, basandosi sullo studio di come questa è formulata. La valutazione automatica che intendiamo utilizzare fa più semplicemente perno sul binomio soluzione-risultati. In sostanza, assumiamo che una soluzione operativa è funzionalmente corretta se produce risultati esatti su una buona varietà di dati iniziali. Riprendendo l'esempio geometrico, la procedura risolutiva dovrà costruire la bisettrice per un campione significativo di angoli (acuti, ottusi, maggiori di 180 gradi, ecc.).
In accordo a questo punto di vista, per valutare la correttezza funzionale di una soluzione operativa può essere sufficiente la capacità di "eseguirla", indipendentemente da un'analisi statica della sua formulazione. La tecnica che proponiamo trae spunto da quest'ultima osservazione. In sostanza, come già esposto da altri autori, intendiamo valutare automaticamente soluzioni espresse in formalismi che siano eseguibili da parte del calcolatore, confrontando i risultati dell'esecuzione con quelli derivanti da una soluzione di riferimento preparata dal docente [Valenti et al., 2001] [Gaudel et al., 1993].
Riassumendo, la risposta è aperta, poiché la soluzione può essere liberamente formulata; la risposta è automaticamente valutata in base a verifiche black-box sui risultati che essa produce.
È chiaro che un giudizio di correttezza fondato sulla sola verifica dei risultati finali (che chiameremo valutazione *black-box*), se strettamente inteso, è suscettibile di ovvie critiche:

- dato un problema, sono possibili soluzioni che pur fornendo risultati corretti, sono progettate in modo strutturalmente o logicamente opinabile;
- al contrario, si possono avere risultati sistematicamente scorretti per un'imperfezione o incompletezza di una soluzione formulata in modo sostanzialmente giusto.

Il primo punto di critica si riferisce alla convenienza di raffinare il giudizio di correttezza nel caso di una soluzione che produca risultati esatti. Questo può essere almeno in parte soddisfatto affiancando al giudizio black-box un'analisi comparativa rispetto alle caratteristiche strutturali della soluzione del docente [Valenti et al., 2001]. Al

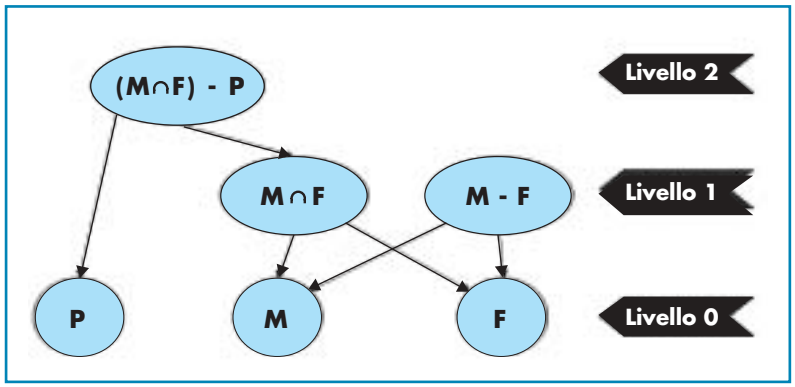

figura 1

Una gerarchia di sottoproblemi.

contrario, nel caso in cui una soluzione produca risultati scorretti, alla luce delle tecnologie attuali, una stima automatica della gravità degli errori commessi dallo studente sembra poco realistica. Questa richiederebbe, infatti, un tipo di analisi semantica non riconducibile ad un mero esame strutturale. Intuitivamente, un giudizio black-box appare applicabile senza riserve quando la complessità del problema è ridotta, in modo tale da rendere poco indicativa una valutazione raffinata di "quanto" una soluzione può essere errata.

UNA GERARCHIA DI SOTTOPROBLEMI

L'idea con la quale si intende superare le perplessità relative all'efficacia delle valutazioni black-box trae spunto dalla metodologia più usata nella pratica per la soluzione di problemi complessi: il "divide et impera". La soluzione di un problema non elementare è scomposta in un'opportuna progressione di passi risolutivi più semplici (sottoproblemi). Risolvendo nell'ordine i sottoproblemi, e sfruttando via via i risultati parziali ottenuti in precedenza, si arriva alla soluzione del problema nella sua interezza. Tale scomposizione di solito si presenta quasi naturalmente nei problemi posti in domini di conoscenza procedurali.

In sostanza, il nostro metodo prevede che il problema originale sia fin dall'inizio suddiviso in una collezione di sottoproblemi, ciascuno di dimensione tale che:

- ogni sottoproblema sia semplice da risolvere, supponendo che quelli che lo precedono nella progressione siano già risolti;
- la bontà della soluzione dei singoli sottoproblemi possa essere soddisfacentemente valutata con un giudizio black-box.

Il test sottoposto allo studente si articolerà in accordo ai sottoproblemi, nel senso che il test descriverà il problema originale, ma proporrà come quesiti specifici anche i singoli sottoproblemi. Lo studente eseguirà il test dando soluzioni coordinate alla collezione di sottoproblemi. Per giudicare l'elaborato nel suo complesso, si valuterà automaticamente in stile black-box la correttezza delle soluzioni date ai singoli sottoproblemi. Una somma pesata delle valutazioni singole fornirà il giudizio complessivo.

Supponiamo, ad esempio, di avere una base di dati per la gestione di un istituto didattico, e immaginiamo di aver bisogno dell'elenco dei docenti che insegnano Matematica e Fisica e che non abitano a Pisa. Si tratterà allora di determinare l'elenco dei docenti che insegnano Matematica (**M**) e quello dei docenti che insegnano Fisica (**F**), dai quali derivare l'elenco **M«F** dei docenti che insegnano entrambe le materie. Dalla lista **M«F** dovremo poi eliminare l'elenco dei docenti che abitano a Pisa (**P**), trovando il risultato finale che possiamo indicare con **(M«F)-P**.

Come si vede, già nel caso di un problema relativamente semplice, si individua la collezione di sottoproblemi gerarchicamente organizzata descritta in figura 1, che tra l'altro mostra che la soluzione di un sottoproblema può essere sfruttata più volte. Infatti, volendo determinare anche la lista dei docenti che insegnano Matematica, ma non Fisica, si potrà effettuare direttamente la differenza (**M-F**) tra la lista **M** e la lista **F**, già disponibili. Il test proporrà i due problemi finali **(M«F)-P** e **M-F**, ma proporrà come quesiti anche i quattro sottoproblemi **P**, **M**, **F** e **M«F**, con i quali lo studente potrà separatamente confrontarsi.

SOTTOPROBLEMI OPERATIVAMENTE SVINCOLATI

L'applicazione del metodo delineato nella sezione precedente sembra porre una difficoltà concettuale e pratica piuttosto imbarazzante. L'incapacità di risolvere un sottoproblema della progressione, apparentemente, impedisce la risoluzione dei successivi. Inoltre, sembra inevitabile che l'eventuale soluzione errata di un sottoproblema si propaghi ai successivi invalidandone i risultati. Il frazionamento del problema originale sarebbe quindi vanificato dalla dipendenza gerarchica dei passi risolutivi.

In realtà, per *eseguire* la soluzione di uno specifico problema ottenendo risultati corretti, c'è bisogno di soluzioni corrette per tutti i sottoproblemi che lo precedono nella gerarchia. Tuttavia, ciò che serve effettivamente è la possibilità di *eseguire* le soluzioni dei sottoproblemi logicamente precedenti, utilizzandone i corrispondenti risul-

tati. Si noti che questo non comporta la necessità di conoscere la formulazione delle soluzioni stesse. Ne segue che la soluzione dei sottoproblemi, gerarchicamente precedenti, non deve essere necessariamente data dallo studente!
Possiamo, infatti, immaginare che lo studente abbia a disposizione le soluzioni di tutti i sottoproblemi, come preventivamente elaborate dal docente. La disponibilità si limiterà però alla possibilità di eseguire queste soluzioni, senza ovviamente poter prendere visione della loro formulazione. Nel caso di programmi Pascal, questo potrebbe essere banalmente realizzato mettendo a disposizione l'eseguibile dei sottoprogrammi, e non il loro codice sorgente.
Si propone pertanto un ambiente di lavoro in cui lo studente possa risolvere uno qualsiasi dei sottoproblemi ed eseguire la sua soluzione, indipendentemente da quali altri sottoproblemi egli abbia già risolto in precedenza. Intuitivamente, affrontando un problema, si può assumere di avere le soluzioni *eseguibili* di tutti gli altri. Lo studente potrà per esempio percorrere la gerarchia in figura 1 partendo dall'alto (top-down) piuttosto che dal basso (bottom-up), come sembrerebbe a prima vista necessario.

LEGAMI GERARCHICI DA RICONOSCERE

Abbiamo detto che un test si articola in accordo ai sottoproblemi in cui è scomposto. Si potrebbe allora obiettare che l'eventuale complessità del problema originale resta alla fine banalizzata, poiché la gerarchia dei passi risolutivi è fornita dal docente. Fermo restando che la collezione dei sottoproblemi è offerta allo studente come un insieme di quesiti distinti, questo non vuol dire che la gerarchia stessa sia resa altrettanto nota nella formulazione del test.
In sostanza, il test potrà descrivere il problema complessivo e la collezione dei sottoproblemi, lasciando allo studente il compito di riconoscere le dipendenze logiche, oltre all'incarico di risolvere i singoli sottoproblemi.
È questo il compromesso con il quale si riesce a valutare automaticamente, con giudizi di tipo black-box, le soluzioni date a problemi non elementari. Lo studente, con la risoluzione della collezione dei sottoproblemi, dimostra la sua conoscenza della materia e la padronanza degli strumenti di lavoro ad essa connessi. Lo studente dimostra inoltre la capacità di inquadrare il problema nella sua interezza riconoscendo le relazioni logiche-gerarchiche tra i diversi sottoproblemi. Stimolare il riconoscimento delle dipendenze logiche, essendo già identificati i passi risolutivi, è un'efficace impostazione didattica per sviluppare le capacità di problem-solving intese come l'attitudine all'analisi, alla sintesi e all'organizzazione. Si noti, d'altra parte, che questo riconoscimento delle dipendenze logiche è un caso elegante di test a corrispondenza.
È interessante osservare che il grado di riconoscimento della gerarchia può essere esaminato senza che lo studente espliciti le connessioni individuate. Quest'idea, apparentemente bizzarra, può essere chiarita con due diversi argomenti che presentiamo di seguito.
Supponiamo che i sottoproblemi siano molto interdipendenti. Il successo nello svolgimento dell'intero test sarà indice del fatto che lo studente vi ha riconosciuto un'organizzazione gerarchica, tale da consentire un'efficace riutilizzazione dei risultati dei sottoproblemi già svolti. Se il test è opportunamente calibrato, risolverlo, ripartendo per ogni sottoproblema da zero, richiederebbe semplicemente troppo tempo! Pertanto, verificando le soluzioni date ai singoli sottoproblemi, si ricava una prova indiretta del riconoscimento di un'efficace gerarchia, magari diversa da quella individuata dal docente.
Qualora interessi, si può anche stimare quanto la gerarchia utilizzata dallo studente aderisca a quella ipotizzata dal docente. Infatti, ritornando ad una generica struttura gerarchica, si consideri un sottoproblema sp e si supponga che esso sia stato correttamente risolto dallo studente, sfruttando le soluzioni che il docente ha dato ai quesiti $sp_1, \ldots, sp_m$, dai quali sp dipende. Se, in fase di correzione, noi "guastiamo" la soluzione del docente di un qualsiasi sp_i, allora anche quella di sp data dallo studente cesserà di funzionare. Possiamo così accertare la reale dipendenza della soluzione di sp da quelle di $sp_1, \ldots, sp_m$. Il che fornisce una prova, diretta questa volta, del ricorso alla gerarchia suggerita dal docente.

MODULAZIONE DELLE DIFFICOLTÀ

È difficile costruire un test ugualmente significativo per lo studente più esperto e per quello meno preparato. In effetti, un test calibrato sul livello medio rischia di essere banale per i bravi o impossibile per i meno preparati. A tal proposito, l'idea fondamentale del CAT si basa sul fatto che, partendo

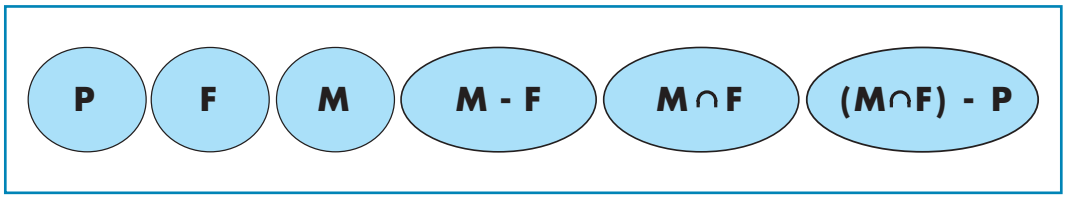

figura 2
Sottoproblemi parzialmente ordinati.

da un'adeguata (ampia) collezione di domande opportunamente classificate, è possibile valutare il livello di competenza dello studente, *mentre esegue il test*, consentendo al sistema di selezionare la domanda successiva nel modo più adeguato sulla base delle risposte precedentemente fornite. In tal modo, evitando le domande troppo semplici o troppo complesse per ciascuno studente, il test risulterà individualizzato, consentendo un giudizio più corretto e più rapido rispetto ai rigidi test tradizionali.

Nell'ambito del problem-solving in cui ci siamo posti, un approccio in stile CAT va opportunamente adattato. È infatti irrealistico ipotizzare l'esistenza di una collezione di problemi che siano tutti una variante l'uno dell'altro e che presentino un sostanziale continuum di difficoltà. Per adattare la difficoltà, bisognerà agire modificando e dettagliando opportunamente la formulazione del problema, mentre lo studente lo risolve.

Un possibile approccio per conseguire tale obiettivo consiste nel formulare un test "difficile all'origine" in cui molte informazioni e suggerimenti utili per lo svolgimento della prova sono nascosti. Per rendere la difficoltà modulabile, basta fare in modo che il sistema fornisca le informazioni aggiuntive solo quando lo studente le richiede, naturalmente registrando il fatto, al fine di adeguare la valutazione.

Vediamo come l'idea può essere applicata al caso della gerarchia di sottoproblemi. I sottoproblemi possono essere ad esempio elencati indipendentemente dalla gerarchia o addirittura in un ordine arbitrario. Per ogni sottoproblema lo studente può chiedere al sistema da quali altri dipende e per ogni richiesta esaudita il sistema registra una penalità.

L'approccio appare ancora più versatile osservando che la gerarchia può essere palesata con diversi gradi d'incompletezza. Infatti, la collezione dei sottoproblemi illustrata in figura 1 può essere ad esempio proposta in più modi:

- esplicitando solo la livellazione, ovvero come in figura 1, ma senza mostrare gli archi tra i nodi;
- presentando la collezione di sottoproblemi in modo parzialmente ordinato rispetto alle dipendenze gerarchiche, come ad esempio in figura 2, dove il quesito *p* precede *q* nell'elenco, se per risolvere *q* ci si può avvalere della soluzione di *p*;
- rinunciando a qualsiasi ordine nella presentazione dei sottoproblemi.

Come vedremo, questa modulazione della difficoltà si può applicare anche in altre occasioni, non esclusa la completezza con la quale la collezione dei sottoproblemi è definita, si veda a tal proposito il paragrafo "Un'esperienza con Excel". Come si vede, alla teorica ricchezza di varianti richiesta da un approccio di tipo CAT, qui si sopperisce con un unico test, in cui la difficoltà non è modulata dal sistema, ma dallo studente stesso, che adatta il problema proposto al suo livello di competenza.

Nel seguito, presenteremo due casi concreti in cui il metodo di test già delineato è stato effettivamente impiegato per l'insegnamento di Informatica Generale, presso l'Accademia Navale di Livorno.

UN'ESPERIENZA CON ACCESS

Prima di descrivere l'implementazione del metodo di test, è utile richiamare, alcuni tratti essenziali di Access. Risalterà così con maggiore evidenza che quanto si propone di valutare automaticamente è un oggetto molto "creativo", prossimo ad un linguaggio naturale, sia pure posto sotto una veste grafica.

Access è un sistema di gestione di basi di dati (DBMS) relazionale in cui le informazioni sono rappresentate sotto forma tabellare. Ogni oggetto da rappresentare, come un cliente nell'esempio in figura 3, è memorizzato come una riga (*record*) in una tabella. I record, omogenei tra loro, costituiscono unità di informazione organizzate per *campi* eterogenei. Le naturali relazioni tra i dati sono realizzate in una forma semplice ed ef-

figura 3
Una tabella di Access.

Clienti : Table

	Codice Cliente	Ragione Sociale	Città	Indirizzo
+	A10	Alfa S.a.S.	Livorno	Viale Italia 60
+	A12	Eterea Inc.	Livorno	Via mazzini 22
+	A20	Stella chiara	Pisa	Via Crispi 81
+	B22	Gli Strozzi	Firenze	Lungarno Lungo 33
+	B31	Bel Fiume	Pisa	Lungarno Pacinotti 78
+	N20	Nicoletti	Pisa	Corso Italia 80

ficace, vale a dire: la coincidenza di valori posti in campi omologhi di tabelle distinte.
La figura 4 illustra un esempio in cui la naturale relazione uno a molti tra i clienti e gli ordini (un cliente può effettuare più ordini, ma ogni ordine è relativo ad un solo cliente) è realizzata duplicando il campo Codice Cliente nelle tabelle dei Clienti e degli Ordini, e considerando in relazione i record delle due tabelle in cui tale informazione coincide.
La parte più interessante dello strumento consiste nel fatto che è possibile estrarre informazioni da un database siffatto in modo alquanto raffinato, utilizzando un vero e proprio linguaggio di interrogazione: lo SQL o *Structured Query Language*. Mediante tale linguaggio è possibile ricavare informazioni analitiche e di sintesi, anche relative a "fatti", non esplicitamente registrati nel database. Vediamone un esempio.
La query SQL in figura 5 genera il risultato (*recordset*) in figura 6, contenente il numero degli ordini inevasi di ciascun cliente.
Tale query, pur modesta nelle sue dimensioni, esegue un certo numero di operazioni sulle informazioni memorizzate nel database: estrae dei campi dalle tabelle esistenti (SELECT-FROM), utilizzando una delle relazioni impostate sul database (INNER JOIN-ON) quindi conteggiando (COUNT) il numero dei record che soddisfano una data condizione (WHERE).
È importante, ai nostri scopi, richiamare che il recordset prodotto dall'esecuzione di una query è del tutto omogeneo ad una tabella. In questo senso, un recordset può essere a sua volta utilizzato come fonte di dati per una query successiva, consentendo la creazione di gerarchie di query in grado di rispondere ad interrogazioni tutt'altro che banali. È evidente che la possibilità di riusare query esistenti corrisponde esattamente all'idea di risolvere un problema per sottoproblemi gerarchicamente organizzati.
Va detto infine che Access mette a disposizione dell'utente uno strumento per la composizione di query che nulla toglie all'espressività dell'SQL, rappresentando solo un modo agevole di soddisfarne la sintassi. Si tratta della *Query By Example* o QBE. In tale ambiente la query è assemblata graficamente utilizzando un layout predefinito.
La figura 7 mostra la QBE equivalente a quella data in figura 5, dove il tratto tra le tabelle corrisponde alla JOIN dell'SQL.
Usando la terminologia già introdotta, diremo quindi che le query QBE sono le soluzioni che lo studente dà ai sottoproblemi,

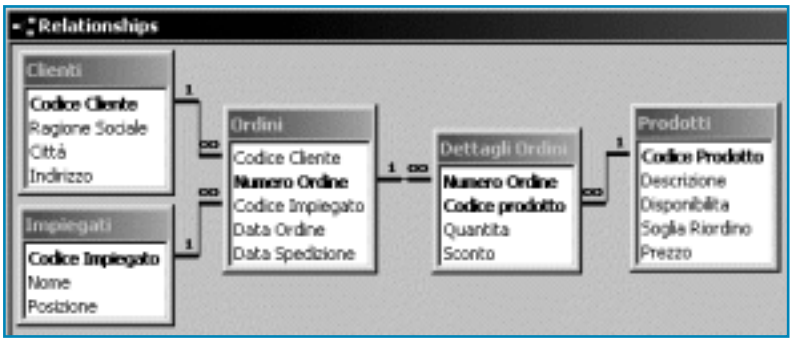

figura 4
Relazioni di Access.

mentre i recordset costituiscono i risultati ottenuti eseguendole.

Un esempio di gerarchia di quesiti
Quello che segue è un semplice esempio di collezione di quesiti variamente legati da dipendenze gerarchiche. Con riferimento alla figura 8, si tratta di sottoproblemi relativi ad una possibile indagine sull'andamento del servizio che gli impiegati della ditta hanno fornito ai clienti. Ciascun quesito può essere facilmente risolto con una sola query, purché si sfrutti con proprietà la gerarchia. In accordo al nostro metodo di test, il riconoscimento dei legami gerarchici fa implicitamente parte dei compiti dello studente. In questo caso specifico lo studente è agevolato, poiché la collezione di quesiti è fornita in modo semi-ordinato come quelli in figura 2.
Considerando ad esempio il quesito 5, è intuitivo che la sua soluzione si può avvalere di quelle dei quesiti 3 e 4. Per il quesito 6 si può sfruttare la soluzione del quesito 3, e per quest'ultimo quella del quesito 1.

figura 5
Una query SQL.

```
SELECT    Clienti.[Codice Cliente], Clienti.[Ragione Sociale],
          Count(Ordini.[Numero Ordine]) AS [Ordini Inevasi]
FROM      Clienti INNER JOIN Ordini ON
          Clienti.[Codice Cliente] = Ordini.[Codice Cliente]
WHERE     Ordini.[Data Spedizione] Is Null
GROUP BY  Clienti.[Codice Cliente], Clienti.[Ragione Sociale];
```

Implementazione del metodo
Il sistema, realizzato in *Visual Basic for Applications* (VBA) all'interno di Access stes-

figura 6
Un recordset in Access.

ClientiConOrdiniInevasi : Select Query

Codice Cliente	Ragione Sociale	Ordini Inevasi
B22	Gli Strozzi	3
B31	Bel Fiume	1
N20	Nicoletti	1

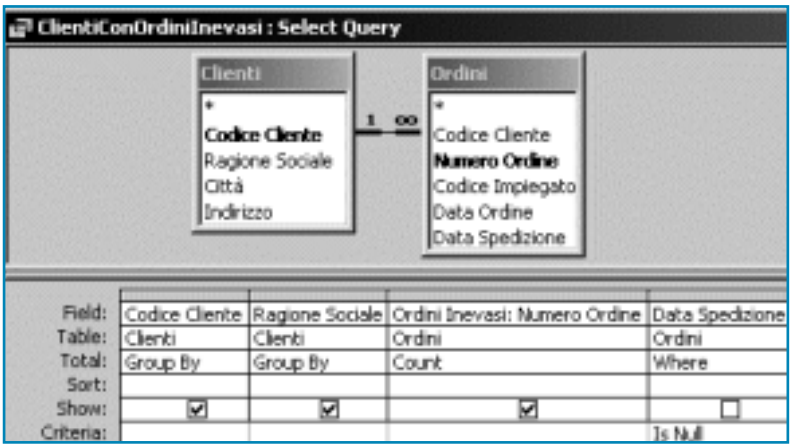

figura 7

Una query by example (QBE).

so, implementa tutti gli ingredienti essenziali del metodo didattico esposto, fornendo strumenti efficaci per ogni fase di vita del test: preparazione, svolgimento e correzione.

La preparazione del test. Scelto un database, il docente costruisce la collezione di sottoproblemi e li inserisce come quesiti nel testo della prova. Il docente provvede quindi a risolvere effettivamente i sottoproblemi realizzando tutte le query in accordo alla gerarchia; a questo punto il sistema genera il database da fornire allo studente come parte integrante del test.

Lo svolgimento del test. Nel database per lo studente, i recordset prodotti dalle soluzioni del docente sono riportati sotto forma di tabelle; in questo modo lo studente ha a disposizione i risultati, ma non può prendere visione delle corrispondenti query. È ovvio che tramite queste *tabelle-risultato* lo studente può validare per confronto le sue soluzioni. Tuttavia, l'aspetto più caratterizzante è che lo studente può utilizzare le tabelle-risultato ogni qualvolta riconosce e vuole sfruttare una dipendenza gerarchica tra sottoproblemi. In tal modo si realizza la possibilità di dare una soluzione corretta ad uno qualsiasi dei quesiti proposti, indipendentemente da quelli non ancora svolti o svolti in maniera errata. Si garantisce così l'indipendenza delle soluzioni e la non propagazione a cascata degli errori.
Il sistema permette inoltre allo studente di modificare liberamente il database per verificare il corretto funzionamento delle sue soluzioni al variare dei dati. Infatti, il sistema provvede su richiesta a rigenerare le tabelle-risultato.

La valutazione del test. Al termine dello svolgimento del test, il sistema dà una valutazione automatica della bontà delle soluzioni formulate dagli studenti. Per ogni studente, il sistema confronta i recordset prodotti dalle sue query con quelli derivanti dalle soluzioni del docente. In questa fase le soluzioni del docente continuano ad essere operanti ad eccetto di quella relativa al quesito di volta in volta esaminato. Questo confronto può essere ripetuto su più database.
In linea di principio una query è considerata corretta se fornisce risultati corrispondenti a quelli attesi per tutti i database campione. Ciò non toglie che si può graduare il giudizio, per esempio sulla base del numero o del tipo dei database campione per i quali il funzionamento delle soluzioni è ritenuto soddisfacente.

UN'ESPERIENZA CON EXCEL

Excel è un foglio di calcolo dove lo spazio è suddiviso in celle indirizzabili mediante una coordinata di colonna, indicata da una lettera o una sequenza di lettere, e una coordinata di riga, che è un numero intero. Ogni cella può contenere un valore di vari tipi (numeri, testo, date, ecc.) o una formula di calcolo in cui possono comparire sia dei valori costanti che dei riferimenti al contenuto di altre celle. Quando una cella contiene una formula, ciò che normalmente si vede sul foglio non è la formula stessa, ma il risultato che si ottiene calcolandola.

figura 8

Esempio di gerarchia di quesiti in Access.

1. *ImpiegatiPerCliente:* per ogni cliente, mostrare quali sono gli impiegati che lo hanno servito.
2. *QuantiOrdiniPerCliente:* per ogni cliente, determinare quanti ordini egli ha emesso.
3. *QuantiImpiegatiPerCliente:* per ogni cliente, determinare quanti sono gli impiegati che lo hanno servito.
4. *ClientiServitiDaUnSoloImpiegato:* elencare i clienti che sono stati serviti da un solo impiegato.
5. *ClientiConPiùOrdiniServitiDaUnSoloImpiegato:* elencare i clienti che hanno emesso più di un ordine, ma sono stati serviti da un solo impiegato.
6. *ClientiServitiDaMaxNumDiImpiegati*: ciascun cliente è stato servito da un certo numero di impiegati distinti; elencare i clienti che sono stati serviti dal massimo numero di impiegati.

	A	B	C	D	E	F	G	H
1	**1) Determinare la Penalità Complessiva del Nuovo Capolista** **2) Determinare le Nuove Posizioni in classifica** **3) Determinare i Nuovi Distacchi** **4) Determinare il Nuovo Capolista**							
2		Posizioni	Distacchi	Pen. Ult. Gara	X1	Nuove Posizioni (2)	Nuovi Distacchi (3)	Nuovo Capolista (4)
3	Gemini	**1**	0	4				
4	Aldebaran	**2**	2	1				
5	Gran carro	**3**	5	11				
6	Stella Polare	**4**	9	6				
7	Orsa Maggiore	**5**	12	8				
8			P.d.C.				P.d.N.C. (1)	
9			8					
10								
11		Posizioni	Distacchi	Pen. Ult. Gara	X1	Nuove Posizioni (2)	Nuovi Distacchi (3)	Nuovo Capolista (4)
12	Gemini	**1**	0	4	12	2	1	
13	Aldebaran	**2**	2	1	11	1	0	Aldebaran
14	Gran carro	**3**	5	11	24	4	13	
15	Stella Polare	**4**	9	6	23	3	12	
16	Orsa Maggiore	**5**	12	8	28	5	17	
17			P.d.C.				P.d.N.C. (1)	
18			8				11	
19								
20		P.d.C. = Pen. Complessiva del Capolista						
21		P.d.N.C. = Pen. Complessiva del Nuovo Capolista						
22		P.C.A. = Pen. Complessiva Aggiornata						
23								

figura 9

Foglio di lavoro per lo studente.

Ai valori così ottenuti si può fare riferimento in altre formule, consentendo in tal modo di organizzare elaborazioni in cascata di notevole complessità. Il sistema di riferimenti, unito all'appropriato meccanismo di copiatura delle formule, consente infine di effettuare elaborazioni di grandi quantità di dati digitando un numero di formule relativamente esiguo.

Il sistema di test per l'insegnamento di Excel, realizzato in Visual Basic, fornisce una completa implementazione del metodo delineato nelle sezioni precedenti, offrendo utili strumenti a tutte le fasi di vita di un test. Secondo la terminologia adottata, in questo caso la soluzione di un sottoproblema consiste in formule Excel, eventualmente da ricopiare in opportuni intervalli di celle.

La preparazione del test

Il docente prepara il foglio di calcolo con i dati iniziali e risolve mediante formule opportune la collezione dei sottoproblemi che intende proporre. Riporta sul foglio stesso il testo della prova, quindi invoca il generatore automatico del test per gli studenti. Il foglio che si ottiene è del tipo illustrato in figura 9, dove la collezione dei quesiti che formulano il test è riportata nella riga 1.

Si tratta di passi operativi per l'aggiornamento della classifica di una manifestazione sportiva articolata su più prove. Si suppone che ad ogni concorrente in ogni prova siano attribuite delle penalità. Il vincitore finale è chi accumula al termine delle prove il minimo numero di penalità. Come per Access, la gerarchia tra i quesiti è comunicata a livello di semi-ordinamento.

Tra le righe 3 e 9, le colonne **B** e **C** descrivono la classifica comprensiva di un certo numero di prove già effettuate. In particolare, le celle da **C3** a **C7** forniscono i distacchi, mentre la cella **C9** completa

l'informazione precisando le penalità attualmente totalizzate dal capolista. La colonna **D** riporta le penalità attribuite durante un'ulteriore prova, rispetto alla quale aggiornare la classifica.
Nelle stesse righe, le rimanenti colonne costituiscono lo spazio in cui lo studente scriverà le sue formule risolutive. Tale zona è suddivisa in aree colorate, una per ogni quesito.
La zona inferiore, tra le righe 11 e 18, ha struttura identica a quella già descritta. Qui si trova una replica dei dati iniziali e le formule del docente. Lo studente può vedere i valori di questa zona ed utilizzarli, ma non può prendere visione delle formule che li producono (tecnicamente, sono *nascoste*).
La presenza di un quesito **X1** non esplicitamente definito, è legata a possibili modulazioni della difficoltà che discuteremo al termine di questa sezione.
Se le dipendenze gerarchiche sono riconosciute e sfruttate opportunamente, ogni quesito è risolvibile con una formula da ricopiare lungo la relativa colonna.

Lo svolgimento del test

Nello svolgimento del test, lo studente inserisce le formule risolutive nello spazio a lui riservato, ottenendo un immediato riscontro della loro bontà. Infatti, i dati nelle celle della parte inferiore mutano di colore da rosso in nero se nelle corrispondenti celle dello studente sono presenti valori coincidenti. Val la pena di notare che lo studente, quando riconosce e vuole sfruttare una dipendenza gerarchica, farà riferimento ai risultati presenti nella parte inferiore del foglio, dove si trovano le soluzioni del docente. Così, nello spirito del metodo proposto, lo studente può affrontare i quesiti indipendentemente da quelli che ha già risolto, bloccando in ogni caso la propagazione a cascata degli eventuali errori nelle sue formule. Infine, poiché sia le formule dello studente sia quelle nascoste del docente sono immediatamente ricalcolate da Excel, lo studente può modificare i dati iniziali a suo piacimento, al fine di controllare la robustezza delle soluzioni date. Le due copie dei dati iniziali sono collegate, nel senso che le variazioni effettuate nella parte superiore si ritrovano automaticamente nell'altra.

La valutazione del test

Il sistema mette a disposizione del docente degli strumenti per la correzione automatica degli elaborati. La valutazione si basa sul confronto dei risultati prodotti dalle formule dello studente rispetto a quelle del docente, nelle varie zone del foglio. Il confronto si ripete su diversi campioni di dati iniziali. Anche in questo caso, le soluzioni del docente continuano ad essere operanti ad eccetto di quella relativa al quesito di volta in volta esaminato.

Il sottoproblema implicito

Riprendiamo infine brevemente il tema della modulazione della difficoltà del test, già trattato in "Modulazione delle difficoltà", dove abbiamo visto come si può agire sul modo in cui la gerarchia è palesata. Si consideri la figura 9. Nel gruppo di celle da E3 a E7 è contenuto un *quesito implicito*, la cui definizione è omessa. I risultati prodotti dalla soluzione del docente sono tuttavia visibili nelle celle da E12 a E16. In sostanza, si fornisce la soluzione eseguibile del sottoproblema, ma non la sua formulazione, che lo studente è stimolato ad immaginarsi esaminando i risultati. Nel nostro caso la formulazione del quesito implicito X1 sarebbe: calcolare le penalità complessive dei diversi concorrenti dopo l'ultima gara.
Si osservi che in questo modo si modula la completezza con la quale la collezione di sottoproblemi è comunicata. In Access la tecnica può essere applicata lasciando la tabella-risultato in assenza della descrizione del corrispondente quesito. Come già detto, lo studente potrà adattare la difficoltà del test alle proprie capacità, chiedendo suggerimenti sul quesito implicito, pagandone ogni volta "il prezzo" in termini di valutazione.

CONCLUSIONI E SVILUPPI

Il metodo presentato appare come un buon compromesso per gestire test diretti a valutare automaticamente la capacità di applicare nozioni e strumenti per la soluzione di problemi non elementari, posti in domini di conoscenza procedurali. L'idea iniziale del metodo prende le mosse dal procedimento più usato per la soluzione di problemi complessi, vale a dire la scomposizione in una gerarchia di sottoproblemi tutti relativamente semplici. Le valutazioni sono effettuate tramite confronti di risultati (valutazioni black-box), cosa facile da realizzare quando le soluzioni risultano eseguibili da parte di un calcolatore.
Il metodo è stato sperimentato con soddisfazione all'Accademia Navale di Livorno per l'insegnamento di Access e Excel. Gli autori sono convinti che l'idea possa essere

applicata anche in altri settori. Ciò che il metodo richiede è che la materia di insegnamento suggerisca dei problemi naturalmente scomponibili in sottoproblemi più semplici, e che la soluzione dei sottoproblemi sia data in un linguaggio o formalismo che un calcolatore possa eseguire. Si pensi per esempio alla cinematica in cui un simulatore, magari numerico, svolge il ruolo dell'esecutore; oppure alla chimica di cui le formule sono il suo naturale "linguaggio".
In domini di conoscenza procedurali, il metodo delineato si propone quale strumento per superare i limiti che approcci come il CAT, pur notevoli nel loro ambito, presentano (si veda a tal proposito l'URL http://www.fairtest.org/facts/computer.htm).
Riportiamo di seguito quelle che, a nostro avviso, sono le principali differenze del metodo proposto rispetto al CAT.

- Sono trattabili problemi sostanzialmente aperti invece di limitarsi a domande con risposte più o meno chiuse.
- La personalizzazione che il CAT realizza adattando il test al singolo studente (a costo, però, della conseguente sensazione di differenza di trattamento) è superata dall'avere un unico test che lo studente può adattare alle proprie competenze attraverso il ricorso ai suggerimenti.
- Infine, ma non meno importante in un ambito di valutazione continua, il metodo proposto non richiede a priori una gran mole di lavoro, come creare e tarare un'ampia suite di domande in quanto, individuato e scomposto il problema da proporre, molti dei suggerimenti in grado di semplificare quasi "ad libitum" il test possono essere generati automaticamente dal sistema mediante un'analisi statica delle soluzioni del docente.

Uno dei punti di merito del metodo è che l'interazione con il calcolatore, al fine di stimare già durante l'effettuazione del test la bontà delle soluzioni, stimola l'interesse e la creatività dello studente, il quale veramente impara facendo. Inoltre, l'idea di proporre un problema non elementare, opportunamente scomposto in una collezione di sottoproblemi con relazioni logico-operative da riconoscere, va nella direzione di sposare il momento didattico con quello di valutazione [Varisco, 2001]. La scomposizione a priori (più o meno dettagliata) del problema orienta lo studente verso una soluzione ben fatta nel suo complesso, convogliando aspetti didattici e formativi. D'altra parte, la scomposizione a priori non esaurisce il compito. Infatti, il riconoscimento dei legami logici e gerarchici tra i sottoproblemi, se lasciata allo studente, contribuisce alla formazione delle abilità di problem-solving. In pratica, ogni test è anche un esempio di problem-solving, possibilmente ricco di suggerimenti metodologici.
Riguardo agli sviluppi del metodo, il meccanismo di valutazione non prevede per adesso il ricorso ad analisi "esperta" del modo in cui le soluzioni sono formulate. È evidente che alle valutazioni black-box potrebbe essere già da subito affiancata un'analisi strutturale di raffinamento. Più in generale, il tipo di interazione con il calcolatore, che noi proponiamo, sembra conservare la sua validità anche nel caso di futuri potenziamenti delle tecniche di analisi statica delle soluzioni. Anzi, l'impressione è che dalla sinergia possano scaturire ottimi risultati.

riferimenti bibliografici

Domenici G. (1988), Le prove strutturate di conoscenza, *Corso di perfezionamento in Metodo della Valutazione scolastica*, Giunti e Lisciani Editori.

Gaudel M.C., Bernot G., Le Gall P., Marre B. (1993), Experience with Black Box Testing from Formal Specifications, in *Proc. of the* 2nd *Int. Conf. on Achieving Quality in Software*, AQuis'93.

Jonassen D.H., Land S.M. (2000), *Theoretical Foundation of Learning Environments*, Erlbaum, Mahwah, NJ.

Papert S. (1993), *The Children's machine-Rethinking in the age the computer*, Basic Books, NewYork.

Rogora E., Sterbini A. (2001), Multiple Choice Quiz in a Web Assisted Teaching Environment, in A. Andronico et al. (eds), *Proc. Didamatica 2001 Informatica per la Didattica*, Bari, Laterza.

Rudner L. M. (1998), *An On-Line, Interactive, Computer Adaptive Testing Tutorial*. http://edres.org/scripts/cat.

Varisco B.M. (2001), Tecnologie didattiche, apprendimento e valutazione, in D. Persico (ed) *Atti del convegno TED*, Genova.

Valenti S., Cucchiarelli A., Panti M. (2001), Assessing Programming Skills Over The Web in A. Andronico et al. (eds), *Proc. Didamatica 2001 Informatica per la Didattica*, Bari, Laterza.